Internet Engineering Task Force	T. Hansen	
Internet-Draft	AT&T Laboratories	
Intended status: Standards Track	C. Daboo	
Expires: January 14, 2010	Apple Inc.	
	July 13, 2009	

Sieve Email Filtering: MIME part Tests, Iteration, Extraction, Replacement and Enclosure
draft-ietf-sieve-mime-loop-09

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79. This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

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 January 14, 2010.

Copyright Notice

Abstract

This document defines extensions to the Sieve email filtering language
to permit analysis and manipulation of the MIME body parts of an email
message.

Note

This document is being discussed on the MTA-FILTERS mailing list, ietf-
mta-filters@imc.org.

Table of Contents

---

1. Introduction   TOC

MIME messages ([RFC2045] (Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies," November 1996.)) are often complex objects, consisting of many parts and sub-parts. This extension defines mechanisms for performing tests on MIME body parts, looping through the MIME body parts, extracting information from a MIME body part, changing the contents of a MIME body part, and enclosing the entire message within a wrapper.

---

2. Conventions Used in This Document   TOC

Conventions for notations are as in [RFC5228] (Guenther, P. and T. Showalter, "Sieve: An Email Filtering Language," January 2008.) section 1.1.
The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] (Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," March 1997.).

---

3. Sieve Loops: Actions "foreverypart" and "break"   TOC

The base Sieve language has no looping mechanism. Given that messages may contain multiple parts, in order to support filters that apply to any and all parts, we introduce a new control command: "foreverypart",

which is an iterator that walks though every MIME part of a message, including nested parts, depth first, and applies the commands in the specified block to each of them. The iterator will start with the first MIME part (as its current context) and will execute a command block (Sieve commands enclosed by {...}). Upon completion of this command block, the iterator advances to the next MIME part (as its current context) and executes the same command block again.
The iterator can be terminated prematurely by a new Sieve command, "break".

> **Usage:** foreverypart [":name" string] block

> **Usage:** break [":name" string];

"foreverypart" commands can be nested inside other "foreverypart" commands. When this occurs, the nested "foreverypart" iterates over the MIME parts contained within the MIME part currently being targeted by the nearest enclosing "foreverypart" command. (That is, the inner loop only operates on children of the bodypart currently accessed by the outer loop.) If that MIME part is a terminal MIME part (i.e. does not contain other MIME parts) then the nested "foreverypart" loop is simply ignored.
Sieve implementations MAY limit the number of nested loops that occur within one another, however they MUST support at least one nested loop inside another loop.
If a name is given to a "break" command, it terminates the closest enclosing loop with the identical matching name. (If a nested "foreverypart" name is the same as a "foreverpart" name in an outer level, the outer level name is hidden.) It is an error if there is no enclosing loop with that name.

---

4. Changes to Sieve Tests

This specification extends the base Sieve "header", "address" and "exists" tests to support targeting those tests at a specific MIME part or at all MIME parts in the enclosing scope.

4.1. Test "header"

The "header" test is extended with the addition of new ":mime" and ":anychild" tagged arguments and their associated options.

> **Usage:**

```
   header [":mime"] [":anychild"] [MIMEOPTS] [COMPARATOR] [MATCH-
   TYPE]
   <header-names: string-list> <key-list: string-list>
```

Usage: The definition of [MIMEOPTS] is:

```
   Syntax: ":type" / ":subtype" / ":contenttype" / ":param" <param-
   list: string-list>
```

When the ":mime" tagged argument is present in the "header" test, it
will parse the MIME header lines in the message so that tests can be
performed on specific elements.
When used outside the context of a "foreverypart" iterator, and without
an ":anychild" tagged argument, the "header" test will examine only the
outer top-level [RFC5322] (Resnick, P., Ed., "Internet Message Format,"
October 2008.) headers of the message.
When used inside the context of a "foreverypart" iterator, and without
an ":anychild" tagged argument, the "header" test will examine the
headers associated with the current MIME part context from the loop.
When used outside the context of a "foreverypart" iterator, and with an
":anychild" tagged argument, the "header" test will examine all MIME
body parts and return true if any of them satisfies the test.
When used inside the context of a "foreverypart" iterator, and with an
":anychild" tagged argument, the "header" test will examine the current
MIME part context and all its nested MIME body parts, returning true if
any of them satisfies the test.
The "header" test with the ":mime" tagged argument can test various
aspects of certain structured MIME headers. Implementations SHOULD
support desegmentation, decoding, and charset translation of parameter
values encoded according to [RFC2231] (Freed, N. and K. Moore, "MIME
Parameter Value and Encoded Word Extensions:
Character Sets, Languages, and Continuations," November 1997.) as part
of this test. Additionally, [RFC2047] (Moore, K., "MIME (Multipurpose
Internet Mail Extensions) Part Three: Message Header Extensions for
Non-ASCII Text," November 1996.) describes a process whereby [RFC5322]
(Resnick, P., Ed., "Internet Message Format," October 2008.) headers
can be encoded in various ways. That encoding is not strictly allowed
in MIME parameters, however in practice it has been used in many email
implementations. So, SIEVE implementations MAY decode [RFC2047] (Moore,
K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message
Header Extensions for Non-ASCII Text," November 1996.) encoded words in
parameter values as part of this test.
These options are available:

:type for a "Content-Type" MIME header field, parses and tests the
   value of the MIME type specified in the header; for a "Content-
   Disposition" MIME header field, parses and tests the value of the
   disposition specified in the header; for other MIME headers, uses
   a blank string for the test

:subtype
for a "Content-Type" MIME header field, parses and tests the value of the MIME subtype specified in the header; for a "Content-Disposition" MIME header field, uses a blank string for the test; for other MIME headers, uses a blank string for the test

:contenttype for a "Content-Type" MIME header field, parses and tests the combined value of the MIME type and subtype specified in the header; for a "Content-Disposition" MIME header field, behaves the same as the :type option; for other MIME headers, uses a blank string for the test

:param parses the header looking for MIME parameters in the header. The supplied string-list lists the names of any parameters to be tested. If any one named parameter value matches any of the test string values, the test will return true.

When the :count option from [RFC5231] (Segmuller, W. and B. Leiba, "Sieve Email Filtering: Relational Extension," January 2008.) is used, the following applies:

a. for :type, :subtype or :contenttype, return a count of the number of headers that parsed successfully

b. for :param, return a count of the number of parameters with the given name that were found

Example:

```
require ["mime", "fileinto"];

if header :mime :type "Content-Type" "image"
{
    fileinto "INBOX.images";
}
```

In this example, any message that contains a MIME image type part at the top-level is saved to the mailbox "INBOX.images".
Example:

```
require ["mime", "fileinto"];

if header :mime :anychild :contenttype
        "Content-Type" "text/html"
{
    fileinto "INBOX.html";
}
```

In this example, any message that contains any MIME part with a content-type of "text/html" is saved to the mailbox "INBOX.html". Example:

```
require ["mime", "foreverypart", "fileinto"];

foreverypart
{
    if allof (
      header :mime :param "filename" :contains
          "Content-Disposition" "important",
      header :mime :subtype "Content-Type" "pdf",
      size :over "100K")
    {
        fileinto "INBOX.important";
        break;
    }
}
```

In this example, any message that contains a MIME part that has a content-disposition with a filename parameter containing the text "important", has a content-subtype of "pdf" and is bigger than 100 Kb is saved to the mailbox "INBOX.important".

4.2. Test "address"

The "address" test is extended with the addition of new ":mime" and ":anychild" tagged arguments and their associated options.

> **Usage:** address [":mime"] [":anychild"] [COMPARATOR] [ADDRESS-PART]
>   [MATCH-TYPE]
>   <header-list: string-list> <key-list: string-list>

When the ":mime" tagged argument is present in the "address" test, it will parse the MIME header lines as if they were standard address header lines in a message so that tests can be performed on specific elements.
The behavior of the ":anychild" tagged argument and the interaction with the "foreverypart" iterator is the same as for the extended "header" test Section 4.1 (Test "header").
That is,

> the use of "address" with no ":mime" and ":anychild" tagged argument is the test defined in [RFC5228] (Guenther, P. and T. Showalter, "Sieve: An Email Filtering Language," January 2008.), i.e. it will *only* operate on top level header fields, whether it is inside "foreverypart" or not.

the use of "address" with ":mime" and no ":anychild" operates on the
current MIME part only (or on the top level header fields, if
outside "foreverypart")

the use of "address" with ":mime" and ":anychild" operates on the
current MIME part and all of its descendants

Example:

```
require ["mime", "fileinto"];

if address :mime :is :all "content-from" "tim@example.com"
{
    fileinto "INBOX.part-from-tim";
}
```

In this example, any message that contains a MIME Content-From header
at the top-level matching the text "tim@example.com" is saved to the
mailbox "INBOX.part-from-tim".

4.3. Test "exists"

The "exists" test is extended with the addition of the new ":mime" and
":anychild" tagged arguments and their associated options.

Usage: exists [":mime"] [":anychild"] <header-names: string-list>

When the ":mime" tagged argument is present in the "exists" test, the
test is extended to check for the existence of MIME headers in MIME
parts.
The behavior of the ":anychild" tagged argument and the interaction
with the "foreverypart" iterator is the same as for the extended
"header" test Section 4.1 (Test "header").
That is,

the use of "exists" with no ":mime" and ":anychild" tagged argument
is the test defined in [RFC5228] (Guenther, P. and T. Showalter,
"Sieve: An Email Filtering Language," January 2008.), i.e. it will
only operate on top level header fields, whether it is inside
"foreverypart" or not.

the use of "exists" with ":mime" and no ":anychild" operates on the
current MIME part only (or on the top level header fields, if
outside "foreverypart")

the use of "exists" with ":mime" and ":anychild" operates on the
current MIME part and all of its descendants

Example:

```
require ["mime", "fileinto"];

if exists :mime :anychild "content-md5"
{
    fileinto "INBOX.md5";
}
```

In this example, any message that contains a MIME Content-MD5 header in any MIME part is saved to the mailbox "INBOX.md5".

---

5. Action "replace"

> **Usage:** replace [":mime"] [":subject" string] [":from" string]
>   <replacement: string>

The "replace" command is defined to allow a MIME part to be replaced with the text supplied in the command.
When used in the context of a "foreverypart" iterator, the MIME part to be replaced is the "current" MIME part. If the current MIME context is a multipart MIME part, the entire multipart MIME part is replaced, which would alter the MIME structure of the message by eliminating all of the children of the multipart part. (Replacing a non-multipart MIME part within a "foreverypart" loop context does not alter the overall message structure.) If the MIME structure is altered, the change takes effect immediately: the "foreverypart" iterator that is executing does not go into the no-longer existing body parts, and subsequent "foreverypart" iterators would use the new message structure.
When used outside the context of a "foreverypart" loop, the MIME part to be replaced is the entire message.
If the :mime parameter is not specified, the replacement string is a text/plain part in UTF-8 [RFC3629] (Yergeau, F., "UTF-8, a transformation format of ISO 10646," November 2003.).
If the :mime parameter is specified, then the replacement string is, in fact, a MIME entity as defined in [RFC2045] (Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies," November 1996.) section 2.4, including both MIME headers and content.
If the entire message is being replaced, the optional ":subject" parameter specifies a subject line to attach to the message that is generated. UTF-8 characters can be used in the string argument; implementations MUST convert the string to [RFC2047] (Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text," November 1996.) encoded words if and only if non-ASCII characters are present. Implementations MUST preserve the previous Subject header as an Original-Subject header.

Implementations MUST preserve all other header fields from the original message with the exception of those relating to the MIME structure that is being replaced.

If the entire message is being replaced, as an indication that the message is no longer as created by the original author of the message, the optional ":from" parameter may be used to specify an alternate address to use in the From field of the message that is generated. The string must specify a valid [RFC5322] (Resnick, P., Ed., "Internet Message Format," October 2008.) mailbox-list. Implementations SHOULD check the syntax and generate an error when a syntactically invalid ":from" parameter is specified. Implementations MAY also impose restrictions on what addresses can be specified in a ":from" parameter; it is suggested that values that fail such a validity check simply be ignored rather than causing the replace action to fail. Implementations MUST preserve the previous From header as an Original-From header. Implementations that support the "editheader" extension [RFC5293] (Degener, J. and P. Guenther, "Sieve Email Filtering: Editheader Extension," August 2008.) MUST ensure that any Original-Subject or Original-From headers added by the system cannot be modified or removed. Implementations MAY prevent the addition of Original-Subject and Orignal-From headers via the "editheader" extension.

If ":mime" is specified and either ":subject" or ":from" is specified, the ":subject:" or ":from" parameter MUST be ignored. This SHOULD be flagged as a compilation error.

---

6. Action "enclose"

>   **Usage:**   enclose <:subject string> <:headers string-list> string

A new Sieve action command is defined to allow an entire message to be enclosed as an attachment to a new message. After enclosure, subsequent actions affecting the message header or content, as well as tests operating on the MIME structure or accessing MIME header fields, use the newly created message instead of the original message; this means that any use of a "replace" action or other similar actions should be executed before the "enclose" action.

If multiple "enclose" actions are executed by a script, the message is enclosed multiple times. (If a Sieve script desires to choose between different enclosures, or wants to delay the enclosure to the end of the script, it can use variables with appropriate tests [RFC5229] (Homme, K., "Sieve Email Filtering: Variables Extension," January 2008.).)

This action does not affect messages that are forwarded via a "redirect" action.

Specifically, the original message becomes a multipart/mixed message with two parts: a text/plain portion with the string argument as its body, and a message/rfc822 portion with the original message enclosed.

The Content-Type: header field becomes multipart/mixed. The optional
Subject: header is specified by the :subject argument; if not present
the subject will be taken from the enclosed message. Any headers
specified by :headers are copied from the old message into the new
message. If not specified by :headers, Date: and From: headers should
be synthesized to reflect the current date and the user running the
Sieve action.

---

7. Action "extracttext"

> **Usage:** extracttext [MODIFIER] [":first" number] <varname: string>

The "extracttext" action may be used within the context of a
"foreverypart" loop and is used to store text into a variable as
defined by [RFC5229] (Homme, K., "Sieve Email Filtering: Variables
Extension," January 2008.). Servers MUST support transcoding of any
textual body part into UTF-8 for use with this action. This requires
decoding any transfer encoding as well as transcoding from the
indicated character set into UTF-8. It stores at most :first characters
of the transcoded content of the current MIME body part in the variable
identified by varname. If the :first parameter is not present, the
whole content of the current MIME body part is stored. In either case
the actually stored data MAY be truncated to conform to implementation
specific limit on variable length and/or on MIME body part length. If
the transfer encoding or character set is unrecognized by the
implementation or recognized but invalid, an empty string will result.
If "extracttext" is used outside the context of a "foreverypart" loop,
the action will set the variable identified by varname to the empty
string. This SHOULD be flagged as a compilation error.
Modifiers are applied on the extracted text before it is stored in the
variable.

---

8. Sieve Capability Strings

A Sieve implementation that defines the "foreverypart" and "break"
actions will advertise the capability string "foreverypart".
A Sieve implementation that defines the ":mime" and ":anychild" tagged
arguments to the "header", "address" and "exists" commands will
advertise the capability string "mime".
A Sieve implementation that defines the "replace" action will advertise
the capability string "replace".
A Sieve implementation that defines the "enclose" action will advertise
the capability string "enclose".

A Sieve implementation that defines the "extracttext" action will advertise the capability string "extracttext". Note that to be useful, the "extracttext" action also requires the "variables" [RFC5229] (Homme, K., "Sieve Email Filtering: Variables Extension," January 2008.) and "foreverypart" capabilities.

---

9. Examples

9.1. Example 1

A Sieve script to replace all the Windows executable attachments in a message would be:

```
require [ "foreverypart", "mime", "replace" ];
foreverypart
{
  if anyof (
          header :mime :contenttype :is
            "Content-Type" "application/exe",
          header :mime :param "filename"
            :matches ["Content-Type", "Content-Disposition"] "*.com" )
  {
    replace "Executable attachment removed by user filter";
  }
}
```

9.2. Example 2

A Sieve script to warn the user about executable attachment types would be:

```
require [ "foreverypart", "mime", "enclose" ];

foreverypart
{
  if header :mime :param "filename"
     :matches ["Content-Type", "Content-Disposition"]
       ["*.com", "*.exe", "*.vbs", "*.scr",
        "*.pif", "*.hta", "*.bat", "*.zip" ]
  {
    # these attachment types are executable
    enclose :subject "Warning" :text
WARNING! The enclosed message contains executable attachments.
These attachments types may contain a computer virus program
that can infect your computer and potentially damage your data.

Before clicking on these message attachments, you should verify
with the sender that this message was sent by them and not a
computer virus.
.
;
    break;
  }
}
```

9.3. Example 3

A Sieve script to extract subject and text out of messages from the
boss:

```
require ["mime", "variables", "extracttext"];

if header :contains "from" "boss@example.org"
{
  # :matches is used to get the value of the Subject header
  if header :matches "Subject" "*"
  {
    set "subject" "${1}";
  }

  # extract the first 100 characters of the first text/* part
  foreverypart
  {
    if header :mime :type :is "Content-Type" "text"
    {
      extracttext :first 100 "msgcontent";
      break;
    }
  }

  # if it's not a 'for your information' message
  if not header :contains "subject" "FYI:"
  {
    # do something using ${subject} and ${msgcontent}
    # such as sending a notification using a
    # notification extension
  }
}
```

---

10. Acknowledgements

---

11. Security Considerations

The "enclose" action creates an entirely new message, as compared to
just redirecting or forwarding the existing message. Therefore, any
site policies applicable to message submission should be enforced.
The looping specification specified here provides easier access to
information about the message contents, which may also be achieved

through other sieve tests. This is not believed to raise any additional security issues beyond those for the Sieve "envelope" and ["body" (Degener, J. and P. Guenther, "Sieve Email Filtering: Body Extension," April 2008.)](#) [RFC5173] tests.

Any change in message content may interfere with digital signature mechanisms that include that content in the signed material. In particular, using "replace" makes direct changes to the body content and will affect the body hash included in DKIM signatures [[RFC4871] (Allman, E., Callas, J., Delany, M., Libbey, M., Fenton, J., and M. Thomas, "DomainKeys Identified Mail (DKIM) Signatures," May 2007.)](#), or the message signature used for S/MIME [[RFC3851] (Ramsdell, B., "Secure/ Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification," July 2004.)](#), PGP [[RFC1991] (Atkins, D., Stallings, W., and P. Zimmermann, "PGP Message Exchange Formats," August 1996.)](#) or OpenPGP [[RFC4880] (Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format," November 2007.)](#).

It is not possible to examine the MIME structure of decrypted content in a multipart/encrypted MIME part.

When "enclose" is used on a message containing a multipart/signed MIME part, the SIEVE implementation MUST ensure that the original message is copied octet-for-octet to maintain the validity of the digital signature.

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

All of the security considerations given in the base Sieve specification also apply to these extensions.

---

12. IANA Considerations

The Original-Subject: and Original-From: headers are to be registered in the Permanent Message Header Fields registry.

The following templates specify the IANA registrations of the Sieve extensions specified in this document. This information should be added to the IANA registry of Sieve Extensions (currently found at http://www.iana.org/assignments/sieve-extensions).

[[RFC Editor Note: replace RFC XXXX with a reference to this RFC.]]

12.1. foreverypart capability

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

Capability name: foreverypart

Description: adds the "foreverypart" and "break" actions for iterating
through MIME parts of a message.

RFC number: RFC XXXX
Contact address: The Sieve discussion list <ietf-mta-filters@imc.org>.

12.2. mime capability

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

Capability name: mime
Description: adds the ":mime" and ":anychild" tagged arguments to the
"header", "address" and "exists" tests. Adds the
:type, :subtype, :contenttype and :param options when ":mime" is used
with the "header" test.

RFC number: RFC XXXX
Contact address: The Sieve discussion list <ietf-mta-filters@imc.org>.

12.3. replace capability

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

Capability name: replace
Description: adds the "replace" action for replacing a MIME body part
of a message.

RFC number: RFC XXXX
Contact address: The Sieve discussion list <ietf-mta-filters@imc.org>.

12.4. enclose capability

To: iana@iana.org
Subject: Registration of new Sieve extension
Capability name: enclose
Description: adds the "enclose" action for enclosing a message with a
wrapper.

RFC number: RFC XXXX
Contact address: The Sieve discussion list <ietf-mta-filters@imc.org>.

12.5. extracttext capability

To: iana@iana.org
Subject: Registration of new Sieve extension
Capability name: extracttext
Description: adds the "extracttext" action for extracting text from a
MIME body part.

RFC number: RFC XXXX
Contact address: The Sieve discussion list <ietf-mta-filters@imc.org>.

13. Change History

[[RFC Editor NOTE: This section is to be removed prior to publication
as an RFC.]]

13.1. draft-ietf-sieve-mime-09

Allow for 2231 processing.
Optional 2047 decoding.
Explain what test options do against a Content-Disposition header.
Explain interaction with :count.
Updated to SIEVE body reference.
Updated IANA section for :mime to also indicate the four options are
added.
Added comments on the interaction of editheader and replace action.
Made variables normative as extracttext requires it.
Fixed some examples.

13.2. draft-ietf-sieve-mime-08

enhance description of enclose and multiple enclose.
Minor nits

13.3. draft-ietf-sieve-mime-07

List :anychild parameter next to :mime, where it was added.
Expand description of "address" and "exists".
In replace, discuss interaction of :mime with :subject/:from.
In enclose, expand discussion o fmultiple enclosures.
Mention compilation error if extracttext is used outside of a
foreverypart loop.

13.4. draft-ietf-sieve-mime-06

Added note to foreverypart about nested identical names hiding outer
names.
Added notes to Security Considerations section about it not working on
multipart/signed sections, and how replace/enclose may affect
signatures.

13.5. draft-ietf-sieve-mime-05

Changed for_every_part to foreverypart, and extract_text to
extracttext.

Add option :name parameter to foreverypart and break. break :name
"string" will break out of closest enclosing foreverypart loop with
that name.

Clarify nesting a bit more.

Minor consistency nit picking.

13.6. draft-ietf-sieve-mime-04

loops are depth first

:anychild clarifications

update examples

grammar nits

transcoding for extract_text

13.7. draft-ietf-sieve-mime-03

add extraction

add security considerations

fill in iana considerations

13.8. draft-ietf-sieve-mime-02

minor syntax glitches in examples

Add clarification on "replace" affecting subsequent for_every_part
loops?

Add IANA considerations for Original-Subject: and Original-From:.

Add note on "enclose" creating From: and Date: headers.

13.9. draft-ietf-sieve-mime-01

what happens when nested for_every_part loop's

a "mime" shorthand for testing the type/subtype, without requiring

interactions with variables
notifications
notifications to calendar service
address tests, exists tests
mimeheader, mimeparameter tests

13.10. draft-ietf-sieve-mime-00

Changed title and text to emphasize MIME Tests.

Changed for.every.part to for_every_part.

Added :anychild to mime test. Default is to use the current context
or outer envelope; specifying :anychild will look at all children.

Added clarifications to replacing parts affecting the structure.

Added :mime option to replace, ala draft-ietf-sieve-vacation-06.

Various other minor nit fixes.

13.11. draft-sieve-mime-loop-04 [TOC]

update reference for recent published rfcs

extract-text now required to do decode transfer encoding and
transcode to UTF-8

removed editheader reference since its not actually used

several text changes as suggested by Nigel Swinson, including re-
writes to abstract and introduction

13.12. draft-hansen-sieve-loop-03 [TOC]

after enclosure, subsequent actions affect newly created message

synthesis of Date/From headers by the enclose action is no longer
controversial

Filled in Security Considerations

Picked up extract_text action from draft-ietf-sieve-notify

Expanded the IANA considerations section

13.13. draft-hansen-sieve-loop-02 [TOC]

Update to 3028bis reference.

Added 2119 conventions section.

Terminology/title tweaks.

Added informative references to body and editheader extensions.

Added description of nested loops.

Replaced mime test by extensions to header, address and exists tests.

13.14. draft-hansen-sieve-loop-01   [TOC]

Merged with draft-daboo-sieve-mime-00.txt.

---

14. References   [TOC]

14.1. Normative References

[TOC]

[RFC2045]	Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies," RFC 2045, November 1996 (TXT).
[RFC2047]	Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text," RFC 2047, November 1996 (TXT, HTML, XML).
[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).
[RFC2231]	Freed, N. and K. Moore, "MIME Parameter Value and Encoded Word Extensions: Character Sets, Languages, and Continuations," RFC 2231, November 1997 (TXT, HTML, XML).
[RFC3629]	Yergeau, F., "UTF-8, a transformation format of ISO 10646," STD 63, RFC 3629, November 2003 (TXT).
[RFC5173]	Degener, J. and P. Guenther, "Sieve Email Filtering: Body Extension," RFC 5173, April 2008 (TXT).
[RFC5228]	Guenther, P. and T. Showalter, "Sieve: An Email Filtering Language," RFC 5228, January 2008 (TXT).
[RFC5229]	Homme, K., "Sieve Email Filtering: Variables Extension," RFC 5229, January 2008 (TXT).
[RFC5231]	

	Segmuller, W. and B. Leiba, "Sieve Email Filtering: Relational Extension," RFC 5231, January 2008 (TXT).
[RFC5293]	Degener, J. and P. Guenther, "Sieve Email Filtering: Editheader Extension," RFC 5293, August 2008 (TXT).
[RFC5322]	Resnick, P., Ed., "Internet Message Format," RFC 5322, October 2008 (TXT, HTML, XML).

14.2. Informative References

[RFC1991]	Atkins, D., Stallings, W., and P. Zimmermann, "PGP Message Exchange Formats," RFC 1991, August 1996 (TXT).
[RFC3851]	Ramsdell, B., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification," RFC 3851, July 2004 (TXT).
[RFC4871]	Allman, E., Callas, J., Delany, M., Libbey, M., Fenton, J., and M. Thomas, "DomainKeys Identified Mail (DKIM) Signatures," RFC 4871, May 2007 (TXT).
[RFC4880]	Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format," RFC 4880, November 2007 (TXT).

Authors' Addresses

	Tony Hansen
	AT&T Laboratories
	200 Laurel Ave.
	Middletown, NJ 07748
	USA
Email:	tony+sieveloop@maillennium.att.com
	Cyrus Daboo
	Apple Inc.
	1 Infinite Loop
	Cupertino, CA 95014
	USA
Email:	cyrus@daboo.name
URI:	http://www.apple.com/