

Internet Engineering Task Force  
Internet-Draft  
Expires: August 31, 2006

T. Hansen  
AT&T Laboratories  
C. Daboo  
Cyrusoft International, Inc.  
February 26, 2006

Sieve Extensions: MIME Tests, MIME Bodypart Iteration, Replacement and  
Enclosure  
[draft-ietf-sieve-mime-loop-00.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/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 August 31, 2006.

#### Copyright Notice

Copyright (C) The Internet Society (2006).

#### Abstract

The current Sieve language has no way to look at individual MIME parts, looping mechanism, or any way to manipulate those individual parts. This document defines extensions for each of these needs.

#### Note

This document is being discussed in the MTA-FILTERS mailing lists, [ietf-mta-filters@imc.org](mailto:ietf-mta-filters@imc.org).

## 1. Introduction

Sieve scripts are used to make decisions about the disposition of a mail message. The original Sieve spec, [5], defined operators for looking at the message headers, such as addresses and the subject. Other extensions provide access to the body of the message, or allow you to manipulate the header of the message. But none of these extensions take into account that MIME messages ([1]) 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, changing the contents of a MIME body part, and enclosing the message with a wrapper.

## 2. Sieve Loops

The current Sieve language has no looping mechanism. Given that messages may contain multiple attachments, in order to support filters that apply to any and all attachments, we introduce a new control command: "for\_every\_part", which is an iterator that walks through every MIME part of a message, including nested parts, and applies 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.

NOTE: Need to deal with this comment

(<http://www.imc.org/ietf-mta-filters/mail-archive/msg02707.html>):

What will this do: `for.every.part { if (some test here) { for.every.part { ... } } }` i.e. will the internal "for" be anchored at the focus established by the enclosing "for" ? I'd hope so, but this should be stated.

NOTE: Need to deal with this comment

(<http://www.imc.org/ietf-mta-filters/mail-archive/msg02707.html>):

BTW, shouldn't there be a "mime" shorthand for testing the type/subtype, without requiring `if allof (mime :type "multipart", mime :subtype "alternative")` ? Especially for any recursive version of "mime" (since the elements inside of "allof" can be matching different mime parts).

The iterator can be terminated prematurely by a new sieve command, "break".



Usage: for\_every\_part block

Usage: break;

### 3. Test "mime"

Usage: mime [:anychild] [COMPARATOR] [MATCH-TYPE] <header-names: string-list> [<parameter-names: string-list>] [<key-list: string-list>]

Usage: mime [:filename] ...

Usage: mime [:type] ...

Usage: mime [:subtype] ...

For Sieve tests on MIME parts, a new Sieve test ("mime") is defined. Similar in concept to the Sieve "header" test, it will parse the MIME header lines so that tests can be performed on specific elements.

If :anychild is NOT specified:

If used within the context of a "for\_every\_part" iterator, the "mime" test will examine the headers associated with the current MIME part context.

If used outside the context of a "for\_every\_part" iterator, the "mime" test will examine only the outer headers of the message.

If :anychild IS specified, the "mime" test will examine all MIME body parts and return true if any of them satisfies the test.

The "mime" test has all of the options available from the header test, [5] [section 5.7](#). In addition, these options are available:

:filename examines the "Content-Disposition:" header field for its "filename" parameter. If there is no "Content-Disposition:" header field, then it will look at the "Content-Type:" header field for the "name" parameter.

:type examines the "Content-Type:" header field type parameter.

:subtype examines the "Content-Type:" header field subtype parameter.

NOTE: We need to add some way to look at parameter lists, ala <http://www.imc.org/ietf-mta-filters/mail-archive/msg01655.html> and related messages: Content-Type: text/plain; charset="foo"



#### **4. Action Replace**

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

A new sieve action command is defined to allow the MIME part to be replaced by a text message. The "replace" command causes a MIME part to be removed and replaced with the text supplied by the command.

When used in the context of a "for\_every\_part" loop, 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 "for\_every\_part" loop context does not alter the overall message structure.)

When used outside the context of a "for\_every\_part" 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.

If the :mime parameter is specified, then the replacement string is, in fact, a MIME entity as defined in [1] [section 2.4](#), including both MIME headers and content. If the optional :mime parameter is not supplied, the reason string is considered to be a UTF-8 string.

If the entire message is being replaced, a ":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 [2] encoded words if and only if non-ASCII characters are present. Implementations MUST preserve the previous Subject header as an Original-Subject header.

If the entire message is being replaced, a ":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 [4] 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 which fail such a validity check simply be ignored rather than causing the vacation action to fail. Implementations MUST preserve the previous From header as an Original-From header.



## 5. 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. This `enclose` action takes precedence over all other message modifications, such as `"replace"`. If multiple `"enclose"` actions are executed by a script, only the text specified on the last one is used when creating the enclosed message. 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 Subject: header is specified by the `:subject` argument. Any headers specified by `:headers` are copied from the old message into the new message.

## 6. Sieve Capability Strings

A Sieve implementation that defines the `"for_every_part"` and `"break"` actions will advertise the capability string `"for_every_part"`.

A Sieve implementation that defines the `"mime"` test 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"`.

## 7. Examples

### 7.1. Example 1

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

```
require [ "for_every_part", "mime", "replace" ];
for_every_part {
    if ( anyof ( mime :subtype :is "exe", mime :filename :matches "*.com" )
{
    replace "Executable attachment removed by user filter";
}
```





```
}
```

## **7.2. Example 2**

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

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

for_every_part {
    if mime :filename :matches [ "*.com", "*.exe", "*.vbs", "*.scr",
                                "*.pif", "*.hta", "*.bat", "*.zip" ] {
        # these attachment types are executable
        enclose :subject "Warning" "
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;
    }
}
```

## **8. Acknowledgements**

Comments from members of the MTA Filters Working Group, in particular Ned Freed, Nigel Swinson and Mark Mallett, are gratefully acknowledged.

## **9. Security Considerations**

To be provided

## **10. IANA Considerations**

To be provided

## **11. Change History**

### **11.1. [draft-ietf-sieve-mime-00](#)**

Changed title 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.

### **11.2. draft-hansen-sieve-loop-01**

Merged with [draft-daboo-sieve-mime-00.txt](#).

## **12. Normative References**

- [1] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", [RFC 2045](#), November 1996.
- [2] Moore, K., "MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text", [RFC 2047](#), November 1996.
- [3] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [4] Resnick, P., "Internet Message Format", [RFC 2822](#), April 2001.
- [5] Showalter, T., "Sieve: A Mail Filtering Language", [RFC 3028](#), January 2001.



Authors' Addresses

Tony Hansen  
AT&T Laboratories  
200 Laurel Ave.  
Middletown, NJ 07748  
USA

Email: [tony+sieve@loop.maillennium.att.com](mailto:tony+sieve@loop.maillennium.att.com)

Cyrus Daboo  
Cyrusoft International, Inc.  
5001 Baum Blvd.  
Suite 780  
Pittsburgh, PA 15213  
USA

Email: [daboo@cyrusoft.com](mailto:daboo@cyrusoft.com)



## Intellectual Property Statement

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](mailto:ietf-ipr@ietf.org).

## Disclaimer of Validity

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.

## Copyright Statement

Copyright (C) The Internet Society (2006). 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.

## Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.



