Sieve Mail Filtering Language: IMAP flag Extension

Status of this memo

Copyright

Abstract

   Recent discussions have shown that it is desirable to set different
   [IMAP] flags on message delivery. This can be done, for example,
   by a Sieve interpreter that works as a part of a Mail Delivery
   Agent.

   This document describes an extension to the Sieve mail filtering
   language for setting [IMAP] flags. The extension allows to set both
   [IMAP] system flags and [IMAP] keywords.

[0](#). Meta-information on this draft

This information is intended to facilitate discussion. It will be
removed when this document leaves the Internet-Draft stage.

Editorial comments are marked with << and >>.

[0.1](#). Discussion

This draft defines an extension to the Sieve mail filtering
language ([RFC 3028](#)) being 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/>.

[0.2](#). Changes from the version submitted to the Sieve mailing list

1. Added addflag and removeflag actions

2. Changed the semantics of setflag (setflag is not additive any
   more)

3. Corrected section "Interaction with Other Sieve Actions".
   Removed incorrect reference to the forward action as to an
   action that prohibits setflag.

4. Added paragraph about the mutual order of "fileinto"/"keep" and
   "setflag"/"addflag"/"removeflag" actions.

[0.3](#). Changes from the revision 00

1. Corrected Capability Identifier section ([Section 2](#))

2. Corrected "Interaction with Other Sieve Actions" section
   ([Section 4](#))

3. Examples were updated to be compatible with Sieve-07 draft

4. Added "mark" and "unmark" actions

[0.4](#). Changes from the revision 01

1. Some language fixes based on Tony Hansen comments

2. Clarified that the extension allows to set both IMAP System
   Flags and Keywords

[0.5](#). Changes from the revision 02

   1. BugFix: all backslashes must be escaped

   2. Added extended example and more detailed description of
      "addflag"/"removeflag" additivity.

   3. Minor example bugfixes

[0.6](#). Changes from the revision 03

   1. Added second way to specify flags to be set (via optional tagged
      arguments). [Tim Showalter]

   2. Rules for using Reject with imapflags relaxed. [Randall Gellens]

   3. Removed ABNF section completely, added syntax description to
      action definition. [Tim Showalter]

   4. Cleaned up the example. [Ken Murchison]

   5. Added FM (Flag Manupulation) acronym.

   6. Clarified "mark"/"unmark" bahavior. [Randall Gellens]

[0.7](#). Changes from the revision 04

   1. "Interaction with Other Sieve Actions" was simplified based on
      comments from Tim Showalter. Added sentence saying that
      imapflags doesn't change an implicit keep.

   2. Several editorial comments from Tim Showalter.

[0.8](#). Changes from the revision 05

   1. Updated copyright, author address, section numbers and references.

   2. Added dependency on Sieve "variables" extension.

   3. Several editorial comments from Matthew Elvey.

   4. Removed "mark" and "unmark" actions.

   5. Added "hasflag" test.

   6. Dropped ":globalflags"

   7. An invalid flag name doesn't cause a script execution failure anymore,
     as imapflags now depends on variables and a variable can have an
arbitrary

value.

0.9. Changes from the revision 06 (now draft-ietf-sieve-imapflags-00.txt)

1. Updated boilerplate and references.

2. Fixed capability string in the extended example.

3. Improved implementation using macros (section 6).

0.10. Changes from draft-ietf-sieve-imapflags-00.txt

1. Added back the internal variable, made the variable parameter to all actions optional.

2. Some editorial suggestions from Mark E. Mallett.

3. Updated boilerplate once again.

1. Introduction

This is an extension to the Sieve language defined by [SIEVE] for setting [IMAP] flags. It adds a new tagged argument to "keep" and "fileinto" that describes the list of flags that have to be set when the message is delivered to the specified mailbox. It also adds several actions to help manipulate list of flags and a test to check if a flag belongs to a list.

Sieve interpreters that don't support integration with IMAP SHOULD ignore this extension.

The capability string associated with extension defined in this document is "imap4flags".

<<Extension name has been changed as revision 03 of this document is widely deployed in CMU Cyrus server>>

The "imap4flags" extension can be used with or without the "variables" extension [Variables]. When the "variables" extension is also present a script can use explicit variable names in setflag/addflag/removeflag actions. See also section 3 for more details.
When the "variables" extension is not present the explicit variable name parameter to setflag/addflag/removeflag MUST NOT be used.

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.

[2.1](). General requirements for flag handling

The following notes apply to processing of Addflag and Removeflag actions,
hasflag test and :flags tagged argument.

A Sieve interpreter MUST ignore empty strings (i.e. "") in a list-of-flags
parameter.

The Sieve interpreter SHOULD check the list of flags for validity as
described by [[IMAP]()] ABNF. In particular non-ASCII characters are not
allowed in flag names. However spaces MUST be always allowed and multiple
spaces between flag names MUST be treated as a single space character.
A string containing a space-separated list of flags is equivalent to a
string list consisting of the flags. The last requirement is to simplify
amalgamation of multiple flag lists.

If a flag validity check fails the flag should be silently
ignored, but a warning message SHOULD be logged by the Sieve interpreter.

[3](). Actions

All actions described in this specification (setflag, addflag, removeflag)
operate on string variables that contain a set of [[IMAP]()] flags.
On variable substitution a set of flags is represented as a string
containing space-separated list of flag names.
<<Should this cause an automatic duplicate elimination?>>

Note that the parameter specifying a variable name to setflag/addflag/
removeflag actions is optional. If the parameter is not specified
the actions operate on the internal variable, which has the empty
value when the script starts execution. If the SIEVE interpreter doesn't
support the "variables" extension [[Variables]()], the present variable
name parameter MUST cause runtime error.

The "addflag" action adds flags to an existing set. The "removeflag"
action removes flags from an existing set. The "setflag" action
replaces an existing set of flags with a new set. The "set" action
defined in [[Variables]()] can be used to replace an existing set of flags
with a new set as well.

The :flags tagged argument to "keep" and "fileinto"
actions is used to associate a set of flags referenced by a variable
with the current message. If the :flags tagged argument is not specified
with those 2 actions, the current value of the internal variable is used
instead. The value also applies to the implicit keep.

<<When keep/fileinto is used multiple times and duplicate elimination
is used, the last flag value wins.>>

[3.1](). Setflag Action

```
Syntax:   setflag [<variablename: string>] <list-of-flags: string-list>

Setflag is used for setting [IMAP] system flags or keywords.
Setflag replaces any previously set flags.


Example:  if size :over 500K {
              setflag "flags" "\\Deleted";
          }

A more substantial example is:

Example:
    if header :contains "from" "boss@frobnitzm.example.edu" {
        setflag "flagvar" "\\Flagged";
        fileinto :flags "${flagvar}" "INBOX.From Boss";
    }
```

3.2. Addflag action

```
Syntax:   addflag [<variablename: string>] <list-of-flags: string-list>
```

Addflag is used to add flags to a list of [IMAP] flags. It doesn't
replace any previously set flags. This means that multiple occurrences
of addflag are treated additively. The order of the flags MAY NOT
be preserved and duplicates are allowed.

The following examples demonstrate requirements described in 2.1.
The following two actions

```
    addflag "flagvar" "\\Deleted";
    addflag "flagvar" "\\Answered";
```

produce the same result as the single action

```
    addflag "flagvar" ["\\Deleted", "\\Answered"];
```

or

```
    addflag "flagvar" "\\Deleted \\Answered";
```

or

```
    addflag "flagvar" "\\Answered   \\Deleted";
```

3.3. Removeflag Action

```
Syntax:   removeflag [<variablename: string>] <list-of-flags: string-list>
```

Removeflag is used to remove flags from a list of [IMAP] flags.

Removeflag clears flags previously set by "set"/"addflag". Calling
removeflag with a flag that wasn't set before is not an error and
is ignored. Note, that if an implementation doesn't perform automatic
duplicate elimination, it MUST remove all occurences of the flags
specified in the second parameter to removeflag. Empty strings in the
list-of-flags MUST be ignored. Also note, that flag names are
case-insensitive, as described in [IMAP].
Multiple removeflag actions are treated additively.

```
   Example:
     if header :contains "Disposition-Notification-To" "mel@example.com" {
         addflag "flagvar" "$MDNRequired";
     }
     if header :contains "from" "imap@cac.washington.example.edu" {
         removeflag "flagvar" "$MDNRequired";
         fileinto :flags "${flagvar}" "INBOX.imap-list";
     }
```

4. **Test hasflag**

   Syntax: hasflag [MATCH-TYPE]
           <variable-list: string-list> <list-of-flags: string-list>

   The "hasflag" test evaluates to true if any of the variables matches any
   flag name. The type of match defaults to ":is".

   Flagname comparisons is always done with the "i;ascii-casemap" operator,
   i.e., case-insensitive comparisons, as defined in [IMAP].

   Note, that if an implementation automatically performs flags reordering
   and/or duplicate elimination, it MUST perform it on both variable-list
   values and flag-list values. This is required so that, when the variable
   "MyFlags" has the value "A B", the following test

    hasflag :is "MyFlags" "b A"

   evaluates to true as expected. The above test can be also written as

    hasflag "MyFlags" ["b","A"]

5. **Tagged argument :flags**

   This specification adds a new optional tagged argument ":flags" that
   alter the behavior of actions "keep" and "fileinto".

   The :flags tagged argument specifies that the flags provided in the
   subsequent argument should be set when fileinto/keep deliver the message
   to the target mailbox/user's main mailbox. If the :flags tagged argument
   is not specified, "keep" or "fileinto" will not set any flag when they
   deliver the message to the mailbox.

```
Syntax:    ":flags" <list-of-flags: string-list>
```

The copy of the message filed into mailbox will have only flags
listed after ":flags".

The Sieve interpreter MUST ignore all flags that it can't store
permanently. This means that the interpreter must not treat failure
to store any flag as a runtime failure to execute the Sieve
script. For example, if the mailbox "INBOX.From Boss" can't store any
flags, then

```
  fileinto :flags "\\Deleted" "INBOX.From Boss";
```

and

```
  fileinto "INBOX.From Boss";
```

are equivalent.

This document doesn't dictate how the Sieve interpreter will set
the [IMAP] flags. In particular, the Sieve interpreter may work as
an IMAP client, or may have direct access to the mailstore.

6. **Implementation Notes**

"Addflag <variable> <flaglist>" can be implemented as several actions
"set <variable> "${variable} <flag>", where <flag> is a flag in flaglist.

A script interpreter MAY reorder flags and remove duplicates from the list.
Thus a SIEVE script writer MUST NOT assume that the order or duplicates
will be preserved.

A "hasflag <variable> <flag>" test can be implemented as follows:

```
 string :matches :comparator "i;ascii-casemap"
        " ${<variable>} " "* <flag> *"
```


"setflag <variable> [<flag1>,<flag2>,...]" can be implemented as
follows:

```
 set <variable> "<flag1> <flag2> ... <flagN>"
```


"addflag <variable> <flag>" can be implemented as:

```
 if not string :matches :comparator "i;ascii-casemap"
    " ${<variable>} " "* <flag> *" {
     set <variable> "${<variable>} <flag>"
 }
```

```
"removeflag <variable> <flag>" can be implemented as:

 if string :matches :comparator "i;ascii-casemap" " ${<variable>} "
    "* <flag> *" {
     set <variable> "${1} ${2}";
 }
 /* Remove any leading space */
 if string :matches :comparator "i;ascii-casemap" "${<variable>}"
    " *" {
     set <variable> "${1}";
 }
 /* Remove any trailing space */
 if string :matches :comparator "i;ascii-casemap" "${<variable>}"
    "* " {
     set <variable> "${1}";
 }
```

7. Interaction with Other Sieve Actions

This extension work only on the message that is currently being
processed by Sieve, it doesn't affect another message generated
as a side affect of any action.

The extension decribed in this document doesn't change the implicit
keep (see section 2.10.2 of [SIEVE]).

8. Other Considerations

This extension intentionally doesn't allow setting [IMAP] flags on
an arbitrary message in the [IMAP] message store.

9. Security Considerations

Security considerations are discussed in the [IMAP], [SIEVE] and
[Variables].
It is belived that this extension doesn't introduce any additional
security concerns.

10. Extended example

```
#
# Example Sieve Filter
# Declare any optional features or extension used by the script
#
require ["fileinto", "imap4flags", "variables"];


#
```

```
# Move large messages to special mailbox
#
if size :over 1M
        {
        addflag "MyFlags" "Big";
        if header :is "From" "boss@company.example.com"
                   {
# The message will be marked as "\Flagged Big" when filed into
# mailbox "Big messages"
                   addflag "MyFlags" "\\Flagged";
                   }
        fileinto :flags "${MyFlags}" "Big messages";
        }

if header :is "From" "grandma@example.net"
        {
        addflag "MyFlags" ["\\Answered", "$MDNSent"];
# If the message is bigger than 1Mb it will be marked as
# "Big \Answered $MDNSent" when filed into mailbox "grandma".
# If the message is shorter than 1Mb it will be marked as
# "\Answered $MDNSent"
        fileinto :flags "${MyFlags}" "GrandMa";  # move to "GrandMa" folder
        }

#
# Handle messages from known mailing lists
# Move messages from IETF filter discussion list to filter folder
#
if header :is "Sender" "owner-ietf-mta-filters@example.org"
        {
        set "MyFlags" "\\Flagged $Work";
# Message will have both "\Flagged" and $Work flags
        keep :flags "${MyFlags}";
        }

#
# Keep all messages to or from people in my company
#
elsif anyof address :domain :is ["From", "To"] "company.example.com"
        {
        keep :flags "${MyFlags}";                # keep in "Inbox" folder
        }
#
# Try and catch unsolicited email.  If a message is not to me,
# or it contains a subject known to be spam, file it away.
#
elsif anyof (not address :all :contains
                 ["To", "Cc"] "me@company.example.com",
             header :matches "subject"
                ["*make*money*fast*", "*university*dipl*mas*"])
        {
```

```
        remove "MyFlags" "\\Flagged";
        # If message header does not contain my address,
        # it's from a list.
        fileinto :flags "${MyFlags}" "spam";   # move to "spam" folder
        }
    else
        {
        # Move all other external mail to "personal"
        # folder.
        fileinto :flags "${MyFlags}" "personal";
        }
```

11. Acknowledgments

This document has been revised in part based on comments and
discussions which took place on and off the Sieve mailing list.

The help of those who took the time to review the draft and make
suggestions is appreciated, especially that of Tim Showalter,
Barry Leiba, Randall Gellens, Ken Murchison, Cyrus Daboo,
Matthew Elvey, Jutta Degener, Ned Freed, Marc Mutz, Nigel Swinson,
Kjetil Torgrim Homme and Mark E. Mallett.

Special thanks to Tony Hansen and David Lamb for helping me
better explain the concept.

12. Author's Address

Alexey Melnikov
Isode Limited

5 Castle Business Village
Hampton, Middlesex
United Kingdom, TW12 2BX

Email: alexey.melnikov@isode.com

13. Normative References

[SIEVE] Showalter, T., "Sieve: A Mail Filtering Language", Mirapoint,
RFC 3028, January 2001.

[ABNF] Crocker, D., "Augmented BNF for Syntax Specifications: ABNF",
Internet Mail Consortium, RFC 2234, November, 1997.

[KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate
Requirement Levels", Harvard University, RFC 2119, March 1997.

[IMAP] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1",

University of Washington, RFC 3501, March 2003.

[Variables] Homme, K. T., "Sieve -- Variables Extension", University of Oslo, work in progress, draft-ietf-sieve-variables-XX.txt

14. 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.

15. Full Copyright Statement

Acknowledgement