Workgroup: Network Working Group
Internet-Draft:
draft-happel-sieve-filter-rule-metadata-00
Published: 13 March 2023
Intended Status: Informational
Expires: 14 September 2023
Authors: H.-J. Happel
audriga GmbH

Sieve Filter Rule Metadata

Abstract

This document describes current practices in managing Sieve scripts and proposes a standardized way for storing filter rule metadata in Sieve comments.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on 14 September 2023.

Table of Contents

1. Introduction

Sieve is a formal language for email filtering specified in (RFC5228). Filters are stored in so called "scripts", which are text files containing Sieve filtering expressions.

Users can edit Sieve scripts directly, in case they have access to the script files, or by using the ManageSieve protocol (RFC5804), if it is made available by their email service.

While experienced users may follow this approach, (RFC5228, section 1) anticipates that "GUI-based editors will be the preferred way of editing filters for a large number of users."

This likely describes current practice, given that typical end users rarely have direct access to their Sieve scripts, and given that most email services do not offer ManageSieve access. Furthermore, even for a popular email client such as Thunderbird, only very rough Script editors seem to exist [TBSieve].

1.1. GUI-based email filter editors

Many Webmail and Groupware software features GUI-based email filter editors, which use Sieve as the underlying script language. When creating filters, the editor will create or update the user's Sieve script. When opening the editor, the script will be parsed and presented in a constrained set of UI widgets (such as dropdown-lists or input fields).

The vast majority (if not all) GUI-based editors follow a similar interaction paradigm, which is designed along the Sieve language design concepts. A user first needs to define a "test" (RFC5228, section 2.5), which defines filtering criteria (e.g., based on sender or size of a message). The second major component is the definition of one or more "actions" (RFC5228, section 4), such as to move an email matching the filtering criteria into a certain mailbox.

The design of existing editors affords users to *modularize* their email filters, so that multiple test/action style "fiter rules" are created. The "else" and "elseif" keywords, which are common in free-form Sieve scripts (RFC5228, section 9), are typically not used in those modularized filter rules.

Modularized filter rules often contain a "stop;" commend as their final action by default, which makes them widely independent of each other. GUI-based email filter editors typically allow users to label and to (re-)order filter rules.

1.2. Indirect creation of email filters

Besides a dedicated GUI-based editor for email filters, many Webmail and Groupware software include additional ways to modify the Sieve script of a user. There are *contextual* interfaces, which create rules in the background or which launch a prefilled filter rule creation dialog.

Examples are:

* Creating a filter rule based on a sender's email address ("Move all emails from "John Doe" to ...")
* Adding a sender to a block- or allowlist

The latter case is also an example for special purpose filters as described in the following.

1.3. Special purpose email filters

Email filters are often used as a technology for realizing other features, such as blocklists, vacation messages [RFC5230], or email

forwarding. In most of these cases, there will be a *dedicated user interface* specific to the feature. This user interface will still write to the user's Sieve script, but the corresponding email filter will either not be editable or hidden in the regular email filter editor.

This special handling in the filter editor (i.e., the recognition of a special purpose filter rule) is typically based on metadata which is captured in Sieve comments (see also Section 4).

1.4. Other kinds of Sieve scripts

Besides capturing rules based on user input, Sieve scripts are sometimes also used in other ways. E.g., some SPAM filtering systems use machine-generated Sieve scripts for their operations. Such scripts are typically hidden from end users and hence cannot be edited. They are out of the scope of this draft.

1.5. Conventions Used in This Document

2. Modular Sieve scripts in practice

2.1. Rule comments

In order to manage modularized Sieve scripts, GUI-based script editors need to capture the following information.

* The *type* of a rule, distinguishing user-defined rules from certain special-purpose rules. Examples for the latter are: VACTION, SPAM, ALLOWED/BLOCKED SENDER, AUTOFORWARD.
* The user-defined name of a rules, as shown in the rule management UI. Some systems will use predefined rule names instead of rule types.

In addition to this, some systems store further information:

* Rule Id: an id for internal reference and/or ordering of rules in processing and list presentation.
* Rule position: a numberic value to maintain the ordering of filter rules in a separate field.
* Rule description: additional notes complementing the rule label

Since none of this information is natively supported by the Sieve rule language, systems fall back to Sieve comments for storing. Various vendor-specific practices have emerged, as seen in Section 4.

2.2. Deactivation of filtering rules

Another aspect of informal standardization has been the deacativation of certain Sieve code. Based on the RFC, Sieve offers two means of deactivation:

- *Commenting out Sieve code
- *Keeping one active and multiple deativated scripts (as per ManageSieve (RFC5804, section 1.4))

While some vendors using type (b) scripts rely on commenting out for disabling individual modular rules, others came up with more formalized approaches such as wrapping disabled rules using a simple "if (false)" clause.

2.2.1. Example: Commenting out

The following filtering rule is take from an OpenXChange-generated Sieve script:

```
Flag: |UniqueId:15|Rulename: testRuleCreate
if true
{
    discard ;
}
```

Once "deactivated" in the user interface, this filtering rule will be represented in the Sieve script as follows:

```
## Flag: |UniqueId:15|Rulename: testRuleCreate
#<!-->if true
#<!-->{
#<!-->    discard ;
#<!-->}
```

2.2.2. Example: Wrapping

The following filtering rule is take from an CriticalPath-generated Sieve script:

```
if allof (true, header :contains "X-Priority" "5")
{
        discard;
}
```

Once "deactivated" in the user interface, this filtering rule will be represented in the Sieve script as follows:

```
if allof (false, header :contains "X-Priority" "5")
{
        discard;
}
```

2.3. Issues

While the described informal workarounds helped vendors to cope within the boundaries of the Sieve language, we see a number of issues with this status quo:

- Interoperability: Vendor-specific workarounds tend to work only with the vendors own user interfaces. Any attempt to edit the resulting Sieve scripts can essentially cause trouble displaying these scripts respectively their contained rules within the vendors UI. Furthermore, editing special-purpose rules might even lead to more severe side effects. The lack of interoperability might be considered a barrier for the implementation of useful Sieve clients, which in turn hinders more widespread adoption of Sieve.
- Portability: similar to the way Sieve clients should be able to interoperate with the Sieve usage of servers, Sieve scripts should be portable between different Sieve server vendors. Otherwise, users will not be able to transfer their data between different providers.

3. Proposed conventions

The purpose of this section is to propose a normative reference for filtering rule comments and deactivation.

3.1. Script comments

(TBD)

3.2. Rule comments

(TBD)

3.3. Deactivation of filtering rules

For the purpose of deactivating rules, the "wrapping" approach described in the previous section should be applied.

(TBD)

4. Examples

In this section, we will document Sieve script patterns used by the software of different vendors.

"(...)" stands for omitted code in example scripts.

4.1. CriticalPath

```
# cp_filter_name: Test filter
# cp_filter_description:
```

4.2. Horde (Ingo)

```
# Sieve Filter
# Generated by Ingo (http://www.horde.org/apps/ingo/) (09/03/2021, 05:37

require ["vacation", "regex", "fileinto", "imapflags", "body", "reject",

# Whitelisted Addresses
if address :all :comparator "i;ascii-casemap" :is ["From", "Sender", "Re
    keep;
    stop;
}

# Vacation
(...)

# Blacklisted Addresses
if address :all :comparator "i;ascii-casemap" :is ["From", "Sender", "Re
    discard;
    stop;
}

# Spamfilter
if header :comparator "i;ascii-casemap" :matches "X-Spam-Status" "yes*"
    fileinto "spamblock";
    stop;
}

# Forwards
if true {
    redirect "fws@foo.com";
}

# userDefined Rule (Name: fileInto userDefinedFolder)
if address :all :comparator "i;ascii-casemap" :contains "To" "x"  {
    fileinto "userDefinedFolder";
    stop;
}
```

4.3. Oracle Communication Messaging Server

```
#RULE: $Name="Blocked senders" $Type="BLOCKED_ADDRESSES" $Version=1 $Lz=
#BEGINFILTER
(...)
#ENDFILTER

#RULE: $Name="Test Folder Bulkmail" $Type="DEFAULT_TYPE" $Version=1 $Lz=
#require "fileinto";
(...)
#ENDRULE
```

4.4. OpenXchange

```
## Flag: vacation|UniqueId:1|Rulename: vacation notice

## Flag: autoforward|UniqueId:2|Rulename: autoforward

## Flag: |UniqueId:0|Rulename: MyTestFiler
```

4.5. OpenWave/InterMail

5. Security considerations

TBD

6. IANA Considerations

This document has no IANA actions at this time.

7. Informative References

[TBSieve] Schmid, T., "Thunderbird Sieve Add-On", <https://addons.thunderbird.net/de/thunderbird/addon/sieve/>.

Author's Address

Hans-Joerg Happel
audriga GmbH

Email: happel@audriga.com
URI: https://www.audriga.com