Network Working Group
Request for Comments: 5490
Category: Standards Track

A. Melnikov
Isode Limited

March 2009

The Sieve Mail-Filtering Language -- Extensions for Checking Mailbox Status and Accessing Mailbox Metadata

Status of This Memo

Copyright Notice

Abstract

This memo defines an extension to the Sieve mail filtering language (RFC 5228) for accessing mailbox and server annotations, checking for mailbox existence, and controlling mailbox creation on "fileinto" action.

Table of Contents

1. Introduction

This memo defines an extension to the Sieve mail filtering language [SIEVE] for accessing mailbox and server annotations. This allows for customization of the Sieve engine behaviour based on variables set using [METADATA].

This document also defines an extension for checking for mailbox existence and for controlling mailbox creation on "fileinto" action.

2. Conventions Used in This Document

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 [KEYWORDS].

Conventions for notations are as in [SIEVE] Section 1.1, including the use of [ABNF].

This document is written with an assumption that readers are familiar with the data model and terms defined in Section 3 of [METADATA].

3. "mailbox" and "mboxmetadata" Extensions

3.1. Test "mailboxexists"

Usage: mailboxexists <mailbox-names: string-list>

The "mailboxexists" test is true if all mailboxes listed in the "mailbox-names" argument exist in the mailstore, and each allows the user in whose context the Sieve script runs to "deliver" messages into it. When the mailstore is an IMAP server, "delivery" of messages is possible if:

a. the READ-WRITE response code is present for the mailbox (see Section 7.1 of [IMAP]), if IMAP Access Control List (ACL) [IMAPACL] is not supported by the server, or

b. the user has ’p’ or ’i’ rights for the mailbox (see Section 5.2 of [IMAPACL]).

Note that a successful "mailboxexists" test for a mailbox doesn’t necessarily mean that a "fileinto" action on this mailbox would succeed. For example, the "fileinto" action might put user over quota. The "mailboxexists" only verifies existence of the mailbox and whether the user in whose context the Sieve script runs has permissions to execute "fileinto" on it.

The capability string for use with the require command is "mailbox".

Example: The following example assumes that the Sieve engine also supports "reject" [REJECT] and "fileinto" [SIEVE]. However, these extensions are not required in order to implement the "mailbox" extension.

```
require ["fileinto", "reject", "mailbox"];
if mailboxexists "Partners" {
   fileinto "Partners";
} else {
   reject "This message was not accepted by the Mailstore";
}
```

3.2. ":create" Argument to "fileinto" Command

Usage: `fileinto [:create] <mailbox: string>`

If the optional ":create" argument is specified with "fileinto", it instructs the Sieve interpreter to create the specified mailbox, if needed, before attempting to deliver the message into the specified mailbox. If the mailbox already exists, this argument is ignored. Failure to create the specified mailbox is considered to be an error.

The capability string for use with the ":create" parameter is "mailbox".

3.3. Test "metadata"

```
Usage:  metadata [MATCH-TYPE] [COMPARATOR]
        <mailbox: string>
        <annotation-name: string> <key-list: string-list>
```

This test retrieves the value of the mailbox annotation "annotation-name" for the mailbox "mailbox" [METADATA]. The retrieved value is compared to the "key-list". The test returns true if the annotation exists and its value matches any of the keys.

The default match type is ":is" [SIEVE]. The default comparator is "i;ascii-casemap" [SIEVE].

The capability string for use with the require command is "mboxmetadata".

Annotations MUST be accessed with the permissions of the user in whose context the Sieve script runs, and annotations starting with the "/private" prefix MUST be those of the user in whose context the Sieve script runs.

Example: The following example assumes that the Sieve engine also supports the "vacation" [VACATION] extension. However, this extension is not required in order to implement the "mboxmetadata" extension.

```
require ["mboxmetadata", "vacation"];

if metadata :is "INBOX"
   "/private/vendor/vendor.isode/auto-replies" "on" {

    vacation text:
I’m away on holidays till March 2009.
Expect a delay.
.
}
```

3.4. Test "metadataexists"

```
Usage:  metadataexists <mailbox: string> <annotation-names: string-
        list>
```

The "metadataexists" test is true if all of the annotations listed in the "annotation-names" argument exist (i.e., have non-NIL values) for the specified mailbox.

The capability string for use with the require command is "mboxmetadata".

4. "servermetadata" Extension

4.1. Test "servermetadata"

```
Usage:  servermetadata [MATCH-TYPE] [COMPARATOR]
        <annotation-name: string> <key-list: string-list>
```

This test retrieves the value of the server annotation "annotation-name" [METADATA]. The retrieved value is compared to the "key-list". The test returns true if the annotation exists and its value matches any of the keys.

The default match type is ":is". The default comparator is "i;ascii-casemap".

The capability string for use with the require command is "servermetadata".

Annotations MUST be accessed with the permissions of the user in whose context the Sieve script runs, and annotations starting with the "/private" prefix MUST be those of the user in whose context the Sieve script runs.

Example: The following example assumes that the Sieve engine also supports "variables" [VARIABLES], "enotify" [NOTIFY], and "envelope" [SIEVE] extensions. However, these extensions are not required in order to implement the "servermetadata" extension.

```
require ["enotify", "servermetadata", "variables", "envelope"];

if servermetadata :matches
   "/private/vendor/vendor.isode/notification-uri" "*" {
    set "notif_uri" "${0}";
}

if not string :is "${notif_uri}" "none" {
    # :matches is used to get the MAIL FROM address
    if envelope :all :matches "from" "*" {
        set "env_from" " [really: ${1}]";
    }

    # :matches is used to get the value of the Subject header
    if header :matches "Subject" "*" {
        set "subject" "${1}";
    }
```

```
           # :matches is used to get the address from the From header
           if address :matches :all "from" "*" {
               set "from_addr" "${1}";
           }

           notify :message "${from_addr}${env_from}: ${subject}"
                  "${notif_uri}";
       }
```

4.2. Test "servermetadataexists"

```
   Usage:  servermetadataexists
           <annotation-names: string-list>
```

The "servermetadataexists" test is true if all of the server annotations listed in the "annotation-names" argument exist (i.e., have non-NIL values).

The capability string for use with the require command is "servermetadata".

5. Security Considerations

Extensions defined in this document deliberately don’t provide a way to modify annotations.

A failure to retrieve data due to the server storing the annotations being down or otherwise inaccessible may alter the result of Sieve processing. So implementations SHOULD treat a temporary failure to retrieve annotations in the same manner as a temporary failure to retrieve a Sieve script. For example, if the Sieve script is stored in the Lightweight Directory Access Protocol (LDAP) and the script can’t be retrieved when a message is processed, then the agent performing Sieve processing can, for example, assume that the script doesn’t exist or delay message delivery until the script can be retrieved successfully. Annotations should be treated as if they are a part of the script itself, so a temporary failure to retrieve them should be handled in the same way as a temporary failure to retrieve the Sieve script itself.

Protocols/APIs used to retrieve annotations MUST provide at least the same level of confidentiality as protocols/APIs used to retrieve Sieve scripts. For example, if Sieve scripts are retrieved using LDAP secured with Transport Layer Security (TLS) encryption, then the protocol used to retrieve annotations must use a comparable mechanism for providing connection confidentiality. In particular, the protocol used to retrieve annotations must not be lacking encryption.

6. IANA Considerations

IANA has added the following registrations to the list of Sieve extensions:

```
To: iana@iana.org
Subject: Registration of new Sieve extension
Capability name: mailbox
Description: adds test for checking for mailbox existence and a new
             optional argument to fileinto for creating a mailbox
             before attempting mail delivery.
RFC number: this RFC
Contact address:
    The Sieve discussion list <ietf-mta-filters@imc.org>

Capability name: mboxmetadata
Description: adds tests for checking for mailbox metadata item
             existence and for retrieving of a mailbox metadata
             value.
RFC number: this RFC
Contact address:
    The Sieve discussion list <ietf-mta-filters@imc.org>

Capability name: servermetadata
Description: adds tests for checking for server metadata item
             existence and for retrieving of a server metadata
             value.
RFC number: this RFC
Contact address:
    The Sieve discussion list <ietf-mta-filters@imc.org>
```

7. Acknowledgements

Thanks to Cyrus Daboo for initial motivation for this document.

Thanks to Barry Leiba, Randall Gellens, and Aaron Stone for helpful comments on this document.

The author also thanks the Open Mobile Alliance’s Mobile Email working group for providing a set of requirements for mobile devices, guiding some of the extensions in this document.

8. References

8.1. Normative References

[ABNF]	Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.
[IMAP]	Crispin, M., "Internet Message Access Protocol - Version 4rev1", RFC 3501, March 2003.
[IMAPACL]	Melnikov, A., "IMAP4 Access Control List (ACL) Extension", RFC 4314, December 2005.
[KEYWORDS]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
[METADATA]	Daboo, C., "The IMAP METADATA Extension", RFC 5464, February 2009.
[SIEVE]	Guenther, P. and T. Showalter, "Sieve: An Email Filtering Language", RFC 5228, January 2008.

8.2. Informative References

[NOTIFY]	Melnikov, A., Leiba, B., Segmuller, W., and T. Martin, "Sieve Email Filtering: Extension for Notifications", RFC 5435, January 2009.
[REJECT]	Stone, A., "Sieve Email Filtering: Reject and Extended Reject Extensions", RFC 5429, March 2009.
[VACATION]	Showalter, T. and N. Freed, "Sieve Email Filtering: Vacation Extension", RFC 5230, January 2008.
[VARIABLES]	Homme, K., "Sieve Email Filtering: Variables Extension", RFC 5229, January 2008.

Author's Address

Alexey Melnikov
Isode Limited
5 Castle Business Village
36 Station Road
Hampton, Middlesex TW12 2BX
UK

EMail: Alexey.Melnikov@isode.com