              Sieve Email Filtering -- Regular Expression Extension
                      draft-ietf-sieve-regex-00.txt

Status of this Memo

Copyright Notice

Abstract

   In some cases, it is desirable to have a string matching mechanism
   which is more powerful than a simple exact match, a substring match
   or a glob-style wildcard match. The regular expression matching
   mechanism defined in this draft should allow users to isolate just
   about any string or address in a message header or envelope.

Meta-information (to be removed prior to publication as an RFC)

   This information is intended to facilitate discussion.

This document is intended to be an extension to the Sieve mail
filtering language, available from the RFC repository as
<ftp://ftp.isi.edu/internet-drafts/draft-ietf-sieve-3028bis-05.txt>.

This document and the Sieve language itself are being discussed on
the MTA Filters mailing list at <mailto:ietf-mta-filters@imc.org>.
Subscription requests can be sent to
<mailto:ietf-mta-filters-request@imc.org?body=subscribe> (send an
email message with the word "subscribe" in the body). More
information on the mailing list along with an archive of back
messages is available at <http://www.imc.org/ietf-mta-filters/>.

Change History (to be removed prior to publication as an RFC)

Changes from draft-murchison-sieve-regex-08:

o Updated to XML source.

o Documented interaction with variables.

Open Issues (to be removed prior to publication as an RFC)

o The major open issue with this draft is what to do, if anything,
   about localization/internationalization. Are [IEEE.1003-2.1992]
   collating sequences and character equivalents sufficient? Should
   we reference the unicode technical specification? Should we punt
   and publish the document as experimental?

o Should we allow shorthands such as \\b (word boundary) and \\w
   (word character)?

o Should we allow backreferences (useful for matching double words,
   etc.)?

Table of Contents

1. Introduction

   This document describes an extension to the Sieve language defined by
   [I-D.ietf-sieve-3028bis] for comparing strings to regular
   expressions.

   Conventions for notations are as in [I-D.ietf-sieve-3028bis] section
   1.1, including use of [RFC2119].

[2](#). Capability Identifier

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

[3](#). **Regex Match Type**

   Commands that support matching may take the optional tagged argument
   ":regex" to specify that a regular expression match should be
   performed. The ":regex" match type is subject to the same rules and
   restrictions as the standard match types defined in [I-D.ietf-sieve-
   3028bis].

   For convenience, the "MATCH-TYPE" syntax element defined in
   [[I-D.ietf-sieve-3028bis](#)] is augmented here as follows:

   MATCH-TYPE =/ ":regex"

   Example:

   require "regex";

   # Try to catch unsolicited email.
   if anyof (
     # if a message is not to me (with optional +detail),
     not address :regex ["to", "cc", "bcc"]
       "me(\\\\+.*)?@company\\\\.com",

     # or the subject is all uppercase (no lowercase)
     header :regex :comparator "i;octet" "subject"
       "^[^[:lower:]]+$") {

     discard;   # junk it
   }

   The ":regex" match type is compatible with both the "i;octet" and
   "i;ascii-casemap" comparators and may be used with them.

   Implementations MUST support extended regular expressions (EREs) as
   defined by [[IEEE.1003-2.1992](#)]. Any regular expression not defined by
   [[IEEE.1003-2.1992](#)], as well as [[IEEE.1003-2.1992](#)] basic regular
   expressions, word boundaries and backreferences are not supported by
   this extension. Implementations SHOULD reject regular expressions
   that are unsupported by this specification as a syntax error.

   The following tables provide a brief summary of the regular
   expressions that MUST be supported. This table is presented here
   only as a guideline. [[IEEE.1003-2.1992](#)] should be used as the
   definitive reference.

```
+------------+-------------------------------------------------------+
| Expression | Pattern                                               |
+------------+-------------------------------------------------------+
|     .      | Match any single character except newline.            |
|            |                                                       |
|    [ ]     | Bracket expression.  Match any one of the enclosed    |
|            | characters.  A hypen (-) indicates a range of         |
|            | consecutive characters.                               |
|            |                                                       |
|    [^ ]    | Negated bracket expression.  Match any one character  |
|            | NOT in the enclosed list.  A hypen (-) indicates a    |
|            | range of consecutive characters.                      |
|            |                                                       |
|    \\      | Escape the following special character (match the     |
|            | literal character).  Undefined for other characters.  |
|            | NOTE: Unlike [IEEE.1003-2.1992], a double-backslash   |
|            | is required as per section 2.4.2 of                   |
|            | [I-D.ietf-sieve-3028bis].                             |
+------------+-------------------------------------------------------+
```

Table 1: Items to match a single character

```
+------------+-------------------------------------------------------+
| Expression | Pattern                                               |
+------------+-------------------------------------------------------+
|   [: :]    | Character class (alnum, alpha, blank, cntrl, digit,   |
|            | graph, lower, print, punct, space, upper, xdigit).    |
|            |                                                       |
|   [= =]    | Character equivalents.                                |
|            |                                                       |
|   [. .]    | Collating sequence.                                   |
+------------+-------------------------------------------------------+
```

Table 2: Items to be used within a bracket expression (localization)

```
+------------+----------------------------------------------------------+
| Expression | Pattern                                                  |
+------------+----------------------------------------------------------+
|     ?      | Match zero or one instances.                             |
|            |                                                          |
|     *      | Match zero or more instances.                            |
|            |                                                          |
|     +      | Match one or more instances.                             |
|            |                                                          |
|   {n,m}    | Match any number of instances between n and m            |
|            | (inclusive). {n} matches exactly n instances. {n,}      |
|            | matches n or more instances.                             |
+------------+----------------------------------------------------------+
```

Table 3: Quantifiers - Items to count the preceding regular
expression

```
+------------+---------------------------------------------+
| Expression | Pattern                                     |
+------------+---------------------------------------------+
|     ^      | Match the beginning of the line or string.  |
|            |                                             |
|     $      | Match the end of the line or string.        |
+------------+---------------------------------------------+
```

Table 4: Anchoring - Items to match positions

```
+------------+----------------------------------------------------------+
| Expression | Pattern                                                  |
+------------+----------------------------------------------------------+
|     |      | Alternation.  Match either of the separated regular      |
|            | expressions.                                             |
|            |                                                          |
|    ( )     | Group the enclosed regular expression(s).                |
+------------+----------------------------------------------------------+
```

Table 5: Other constructs

4. Interaction with Sieve Variables

This extension is compatible with, and may be used in conjunction
with the Sieve Variables extension [I-D.ietf-sieve-variables].

4.1. Match variables

A sieve interpreter which supports both "regex" and "variables", MUST
set "match variables" (as defined by [I-D.ietf-sieve-variables]
section 3.2) whenever the ":regex" match type is used. The list of
match variables will contain the strings corresponding to the group
operators in the regular expression. The groups are ordered by the
position of the opening parenthesis, from left to right. Note that
in regular expressions, expansions match as much as possible (greedy
matching).

Example:

```
require ["fileinto", "regex", "variables"];

if header :regex "List-ID" "<(.*)@" {
    fileinto "lists.${1}"; stop;
}

# Imagine the header
# Subject: [acme-users] [fwd] version 1.0 is out
if header :regex "Subject" "^[(.*)] (.*)$" {
    # ${1} will hold "acme-users] [fwd"
    stop;
}
```

4.2. Set modifier :quoteregex

A sieve interpreter which supports both "regex" and "variables", MUST
support the optional tagged argument ":quoteregex" for use with the
"set" action. The ":quoteregex" modifier is subject to the same
rules and restrictions as the standard modifiers defined in
[I-D.ietf-sieve-variables] section 4.

For convenience, the "MODIFIER" syntax element defined in [I-D.ietf-
sieve-variables] is augmented here as follows:

```
MODIFIER  =/  ":quoteregex"
```

This modifier adds the necessary quoting to ensure that the expanded
text will only match a literal occurrence if used as a parameter to
:regex. Every character with special meaning (".", "*", "?", etc.)
is prefixed with "\" in the expansion. This modifier has a

precedence value of 20 when used with other modifiers.

[5](#). IANA Considerations

The following template specifies the IANA registration of the "regex"
Sieve extension specified in this document:

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

Capability name: regex
Capability keyword: regex
Capability arguments: N/A
Standards Track/IESG-approved experimental RFC number: this RFC
Person and email address to contact for further information:
    Kenneth Murchison
    E-Mail: murch@andrew.cmu.edu

This information should be added to the list of Sieve extensions
given on http://www.iana.org/assignments/sieve-extensions.

6. Security Considerations

Security considerations are discussed in [I-D.ietf-sieve-3028bis].
It is believed that this extension does not introduce any additional
security concerns.

However, a poor implementation COULD introduce security problems
ranging from degradation of performance to denial of service. If an
implementation uses a third-party regular expression library, that
library should be checked for potentially problematic regular
expressions, such as "(.*)*".

7. Normative References

[I-D.ietf-sieve-3028bis]
           Showalter, T. and P. Guenther, "Sieve: An Email Filtering
           Language", draft-ietf-sieve-3028bis-05 (work in progress),
           November 2005.

[I-D.ietf-sieve-variables]
           Homme, K., "Sieve Extension: Variables",
           draft-ietf-sieve-variables-08 (work in progress),
           December 2005.

[IEEE.1003-2.1992]
           Institute of Electrical and Electronics Engineers,
           "Information Technology - Portable Operating System
           Interface (POSIX) - Part 2: Shell and Utilities (Vol. 1)",
           IEEE Standard 1003.2, 1992.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
           Requirement Levels", BCP 14, RFC 2119, March 1997.

Appendix A. Acknowledgments

Most of the text documenting the interaction with Sieve variables was
taken from an early draft of Kjetil Homme's Sieve variables
specification.

Thanks to Tim Showalter, Alexey Melnikov, Tony Hansen, Phil Pennock,
Jutta Degener, and Ned Freed for their help with this document.

Author's Address

   Kenneth Murchison
   Carnegie Mellon University
   5000 Forbes Avenue
   Cyert Hall 285
   Pittsburgh, PA 15213
   US

   Phone: +1 412 268 2638
   Email: murch@andrew.cmu.edu

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment