

SIEVE Email Filtering Working
Group
Internet-Draft
Expires: January 14, 2007

C. Daboo
July 13, 2006

SIEVE Email Filtering: Spamtest and Virustest Extensions
draft-ietf-sieve-spamtestbis-05

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/lid-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 January 14, 2007.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

The SIEVE email filtering language "spamtest", "spamtestplus" and "virustest" extensions permit users to use simple, portable commands for spam and virus tests on email messages. Each extension provides a new test using matches against numeric "scores". It is the responsibility of the underlying SIEVE implementation to do the actual checks that result in proper input to the tests.

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

Internet-Draft

SIEVE Spamtest and Virustest

July 2006

Changes from -04:

1. Relaxed capability requirement so that both spamtest and spamtestplus can be present.
2. Tweaked text describing spam numeric value ranges.

Changes from -03:

1. Clarified that there are two possible ways to test for not-spam.
2. Clarified that 'not tested for xxx' also implies 'SIEVE could not determine whether a test was done or not'.

Changes from -02:

1. Changed formatting of tables.
2. Fixed missing 2119 definitions.
3. Moved reference to previous extension to informative.
4. Minor text improvements.
5. Fixed some single/double quote issues.
6. Reworded abstract, introduction and overview to use better SIEVE terminology when describing tests, commands and results.
7. Remove "untested" string result from ":percent" test.
8. Allow ":count" match type to be used for tested/untested checks.

Changes from -01:

1. Changed ACAP reference to i18n-comparators draft.
2. Changed MUST in security section for virus checker updates to plain must.
3. Return string "untested" when :percent is used and no test has been done.
4. Remove MUST NOT for having both spamtestplus and spamtest capabilities present, and instead make it a SHOULD NOT.
5. Add text to state that implementations MUST return an error if spamtestplus is not present when :percent is used.
6. Tweak first para of security considerations to better reflect reality of testing.
7. Syntax -> Usage.
8. Updated references to 3028bis and 3431bis.

Changes from -00:

1. Added description of how to check for untested when using :percent.
2. Changed requires item to "spamtestplus".
3. Changed text describing which requires item needs to be present.

Changes from [RFC3685](#):

1. Added ":percent" argument to spamtest.

Daboo

Expires January 14, 2007

[Page 2]

Internet-Draft

SIEVE Spamtest and Virustest

July 2006

Table of Contents

1.	Introduction and Overview	4
2.	Conventions Used in This Document	4
3.	SIEVE Extensions	5
3.1.	General Considerations	5
3.2.	Test spamtest	5
3.2.1.	spamtest without :percent argument	6
3.2.2.	spamtest with :percent argument	7
3.3.	Test virustest	9
4.	Security Considerations	11
5.	IANA Considerations	11
5.1.	spamtest registration	12
5.2.	virustest registration	12
5.3.	spamtestplus registration	13
6.	References	13
6.1.	Normative References	13
6.2.	Informative References	13
Appendix A.	Acknowledgments	14
	Author's Address	15
	Intellectual Property and Copyright Statements	16

1. Introduction and Overview

SIEVE scripts are frequently being used to do spam and virus filtering based on either implicit script tests (e.g. tests for "black-listed" senders directly encoded in the SIEVE script), or via testing messages modified by some external spam or virus checker that handled the message prior to SIEVE. The use of third-party spam and virus checker tools poses a problem since each tool has its own way of indicating the result of its checks. These usually take the form of a header added to the message, the content of which indicates the status using some syntax defined by the particular tool. Each user has to then create their own SIEVE scripts to match the contents of these headers to do filtering. This requires the script to stay in synchronization with the third party tool as it gets updated or perhaps replaced with another. Thus scripts become tied to specific environments, and lose portability.

The purpose of this document is to introduce two SIEVE tests that can be used to implement "generic" tests for spam and viruses in messages processed via SIEVE scripts. The spam and virus checks themselves are handled by the underlying SIEVE implementation in whatever manner is appropriate, so that the SIEVE spam and virus test commands can be used in a portable way.

In order to do numeric comparisons against the returned strings, server implementations MUST also support the SIEVE relational [[I-D.ietf-sieve-3431bis](#)] extension, in addition to the extensions described here. All examples below assume the relational extension is present.

[2.](#) Conventions Used in This Document

Conventions for notations are as in [[I-D.ietf-sieve-3028bis](#)] [section 1.1](#).

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 [[RFC2119](#)].

The term "spam" is used in this document to refer to unsolicited or unwanted email messages. This document does not attempt to define what exactly constitutes spam, or how it should be identified, or what actions should be taken when detected.

The term "virus" is used in this document to refer to any type of message whose content can cause malicious damage. This document does not attempt to define what exactly constitutes a virus, or how it

Daboo

Expires January 14, 2007

[Page 4]

Internet-Draft

SIEVE Spamtest and Virustest

July 2006

should be identified, or what actions should be taken when detected.

[3.](#) SIEVE Extensions

[3.1.](#) General Considerations

The "spamtest" and "virustest" tests described below evaluate the results of implementation-specific spam and virus checks in a portable way. The implementation may, for example, check for third-party spam tool headers and determine how those map into the way the test commands are used. To do this, the underlying SIEVE implementation provides a normalized result string as one of the inputs to each test command. The normalized result string is considered to be the value on the left hand side of the test, and the comparison values given in the test command are considered to be on the right hand side.

The normalized result starts with a digit string, with its numeric value within the range of values used by the specific test, indicating the severity of spam or viruses in a message or whether any tests were done at all. This may optionally be followed by a

space (%x20) character and arbitrary text, or in one specific case a single keyword is returned. The numeric value can be compared to specific values using the SIEVE relational [[I-D.ietf-sieve-3431bis](#)] extension in conjunction with the "i;ascii-numeric" comparator [[I-D.newman-i18n-comparator](#)], which will test for the presence of a numeric value at the start of the string, ignoring any additional text in the string. The optional text can be used to carry implementation specific details about the tests and descriptive comments about the result. Tests can be done using standard string comparators against this text if it helps to refine behavior, however this will break portability of the script as the text will likely be specific to a particular implementation.

In addition, the SIEVE relational [[I-D.ietf-sieve-3431bis](#)] ":count" match type can be used to determine if the underlying implementation actually did a test. If the underlying spam or virus test was done, the ":count" of the normalized result will return the numeric value "1", whilst if the test was not done, or the SIEVE implementation could not determine if a test was done or not done, the ":count" value will be "0" (zero).

[3.2.](#) Test spamtest

```
Usage:      spamtest [":percent"] [COMPARATOR] [MATCH-TYPE]
            <value: string>
```

SIEVE implementations that implement the "spamtest" test use an identifier of either "spamtest" or "spamtestplus" for use with the capability mechanism.

If the ":percent" argument is not used with any spamtest test, then one or both of "spamtest" or "spamtestplus" capability identifiers MUST be present.

If the ":percent" argument is used with any spamtest test, then the "spamtestplus" capability identifier MUST be present. SIEVE implementations MUST return an error if the ":percent" argument is used and "spamtestplus" is not specified.

In the interests of brevity and clarity, scripts SHOULD NOT specify both "spamtestplus" and "spamtest" capability identifiers together.

The "spamtest" test evaluates to true if the normalized spamtest result matches the value. The type of match is specified by the optional match argument, which defaults to ":is" if not specified.

[3.2.1.](#) spamtest without :percent argument

When the ":percent" argument is not present in the "spamtest" test, the normalized result string provided for the left hand side of the test starts with a numeric value in the range "0" (zero) through "10", with meanings summarized below:

spamtest value	interpretation
0	message was not tested for spam, or SIEVE could not determine whether any test was done
1	message was tested and is clear of spam
2 - 9	message was tested and may contain spam; a higher number indicates a greater likelihood of spam
10	message was tested and definitely contains spam

The underlying SIEVE implementation will map whatever spam check is done into this numeric range, as appropriate.

Examples:

```
require ["spamtest", "fileinto",
        "relational", "comparator-i;ascii-numeric"];

if spamtest :value "eq" :comparator "i;ascii-numeric" "0"
{
    fileinto "INBOX.unclassified";
}
elsif spamtest :value "ge" :comparator "i;ascii-numeric" "3"
```

```

{
    fileinto "INBOX.spam-trap";
}

```

In this example, any message that has not passed through a spam check tool will be filed into the mailbox "INBOX.unclassified". Any message with a normalized result value greater than or equal to "3" is filed into a mailbox called "INBOX.spam-trap" in the user's mailstore.

[3.2.2.](#) spamtest with :percent argument

When the ":percent" argument is present in the "spamtest" test, the normalized result string provided for the left hand side of the test starts with a numeric value in the range "0" (zero) through "100", with meanings summarized below:

spamtest value	interpretation
0	message was tested and is clear of spam, or was not tested for spam, or SIEVE could not determine whether any test was done
1 - 99	message was tested and may contain spam; a higher percentage indicates a greater likelihood of spam
100	message was tested and definitely contains spam

The underlying SIEVE implementation will map whatever spam check is done into the numeric range, as appropriate.

To determine whether the message was tested for spam or not, two options can be used:

- a. a test with or without the ":percent" argument and ":count" match type, testing for the value "0" as described in [Section 3.1](#).

- b. a test without the ":percent" argument using the ":value" match

type, testing for the normalized result value "0" as described in [Section 3.2.1](#).

Examples:

```
require ["spamtestplus", "fileinto",
        "relational", "comparator-i;ascii-numeric"];

if spamtest :value "eq"
    :comparator "i;ascii-numeric" "0"
{
    fileinto "INBOX.unclassified";
}
elsif spamtest :percent :value "eq"
    :comparator "i;ascii-numeric" "0"
{
    fileinto "INBOX.not-spam";
}
elsif spamtest :percent :value "lt"
    :comparator "i;ascii-numeric" "37"
{
    fileinto "INBOX.spam-trap";
}
else
{
    discard;
}
```

In this example, any message that has not passed through a spam check tool will be filed into the mailbox "INBOX.unclassified". Any message that is classified as definitely not containing spam (normalized result value "0") will be filed into the mailbox "INBOX.not-spam". Any message with a normalized result value less than "37" is filed into a mailbox called "INBOX.spam-trap" in the user's mailstore. Any other normalized result value will result in the message being discarded.

Alternatively, the SIEVE relational [\[I-D.ietf-sieve-3431bis\]](#) ":count" match type can be used:

Examples:

```
if spamtest :percent :count "eq"
    :comparator "i;ascii-numeric" "0"
{
    fileinto "INBOX.unclassified";
}
elsif spamtest :percent :value "eq"
    :comparator "i;ascii-numeric" "0"
{
    fileinto "INBOX.not-spam";
}
elsif spamtest :percent :value "lt"
    :comparator "i;ascii-numeric" "37"
{
    fileinto "INBOX.spam-trap";
}
else
{
    discard;
}
```

This example will result in exactly the same behavior as the previous one.

[3.3.](#) Test virustest

Usage: virustest [COMPARATOR] [MATCH-TYPE]
 <value: string>

SIEVE implementations that implement the "virustest" test have an identifier of "virustest" for use with the capability mechanism.

The "virustest" test evaluates to true if the normalized result string matches the value. The type of match is specified by the optional match argument, which defaults to ":is" if not specified.

The normalized result string provided for the left side of the test starts with a numeric value in the range "0" (zero) through "5", with meanings summarized below:

virustest value	interpretation
0	message was not tested for viruses, or SIEVE could not determine whether any test was done
1	message was tested and contains no known viruses
2	message was tested and contained a known virus which was replaced with harmless content
3	message was tested and contained a known virus which was "cured" such that it is now harmless
4	message was tested and possibly contains a known virus
5	message was tested and definitely contains a known virus

The underlying SIEVE implementation will map whatever virus checks are done into this numeric range, as appropriate. If the message has not been categorized by any virus checking tools, then the virustest result is "0".

Example:

```
require ["virustest", "fileinto",
        "relational", "comparator-i;ascii-numeric"];

if virustest :value "eq" :comparator "i;ascii-numeric" "0"
{
    fileinto "INBOX.unclassified";
}
if virustest :value "eq" :comparator "i;ascii-numeric" "4"
{
    fileinto "INBOX.quarantine";
}
```

```
}
elseif virustest :value "eq" :comparator "i;ascii-numeric" "5"
{
    discard;
}
```

In this example, any message that has not passed through a virus check tool will be filed into the mailbox "INBOX.unclassified". Any message with a normalized result value equal to "4" is filed into a

Daboo

Expires January 14, 2007

[Page 10]

Internet-Draft

SIEVE Spamtest and Virustest

July 2006

mailbox called "INBOX.quarantine" in the user's mailstore. Any message with a normalized result value equal to "5" is discarded (removed) and not delivered to the user's mailstore.

[4.](#) Security Considerations

SIEVE implementations SHOULD ensure that "spamtest" and "virustest" tests only report spam and virus test results for messages that actually have gone through a legitimate spam or virus check process. In particular, if such checks rely on the addition and subsequent checking of private header fields, it is the responsibility of the implementation to ensure that such headers cannot be spoofed by the sender or intermediary and thereby prevent the implementation from being tricked into returning the wrong result for the test.

Server administrators must ensure that the virus checking tools are kept up to date, to provide reasonable protection for users using the "virustest" test. Users should be made aware of the fact that the "virustest" test does not provide a 100% reliable way to remove all viruses, and they should continue to exercise caution when dealing with messages of unknown content and origin.

Beyond that, the "spamtest" and "virustest" extensions do not raise any security considerations that are not present in the base [\[I-D.ietf-sieve-3028bis\]](#) protocol, and these issues are discussed in [\[I-D.ietf-sieve-3028bis\]](#).

[5.](#) IANA Considerations

The following templates specify the IANA registration of the Sieve

extensions specified in this document. The registrations for "spamtest" and "virustest" replace those from in [[RFC3685](#)]:

Daboo Expires January 14, 2007 [Page 11]

Internet-Draft SIEVE Spamtest and Virustest July 2006

[5.1.](#) spamtest registration

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

Capability name: spamtest
Capability keyword: spamtest
Capability arguments: N/A
Standards Track/IESG-approved experimental RFC number: this RFC
Person and email address to contact for further information:

Cyrus Daboo

[<mailto:cyrus@daboo.name>](mailto:cyrus@daboo.name)

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

[5.2.](#) virustest registration

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

Capability name: virustest
Capability keyword: virustest

Capability arguments: N/A

Standards Track/IESG-approved experimental RFC number: this RFC

Person and email address to contact for further information:

Cyrus Daboo

<mailto:cyrus@daboo.name>

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

Daboo

Expires January 14, 2007

[Page 12]

Internet-Draft

SIEVE Spamtest and Virustest

July 2006

[5.3.](#) spamtestplus registration

To: iana@iana.org

Subject: Registration of new Sieve extension

Capability name: spamtestplus

Capability keyword: spamtestplus

Capability arguments: :percent

Standards Track/IESG-approved experimental RFC number: this RFC

Person and email address to contact for further information:

Cyrus Daboo

<mailto:cyrus@daboo.name>

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

[6.](#) References

[6.1.](#) Normative References

- [I-D.ietf-sieve-3028bis]
Showalter, T. and P. Guenther, "Sieve: An Email Filtering Language", [draft-ietf-sieve-3028bis-07](#) (work in progress), June 2006.
- [I-D.ietf-sieve-3431bis]
Segmuller, W. and B. Leiba, "Sieve Extension: Relational Tests", [draft-ietf-sieve-3431bis-04](#) (work in progress), December 2005.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.

[6.2.](#) Informative References

- [I-D.newman-i18n-comparator]
Newman, C., "Internet Application Protocol Collation Registry", [draft-newman-i18n-comparator-12](#) (work in progress), June 2006.
- [RFC3685] Daboo, C., "SIEVE Email Filtering: Spamtest and VirusTest Extensions", [RFC 3685](#), February 2004.

Daboo	Expires January 14, 2007	[Page 13]
-------	--------------------------	-----------

Internet-Draft	SIEVE Spamtest and Virustest	July 2006
----------------	------------------------------	-----------

[Appendix A.](#) Acknowledgments

Thanks to Mark E. Mallett, Tony Hansen, Jutta Degener, Ned Freed, Ashish Gawarikar, Alexey Melnikov, Nigel Swinson and Aaron Stone for comments and corrections.

Daboo

Expires January 14, 2007

[Page 14]

Internet-Draft

SIEVE Spamtest and Virustest

July 2006

Author's Address

Cyrus Daboo

Email: cyrus@daboo.name

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.

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.

