

Network Working Group	N. Freed	
Internet-Draft	Sun Microsystems	
Intended status: Standards Track	December 25, 2008	
Expires: June 28, 2009		

## **Sieve Email Filtering: Ihave Extension draft-freed-sieve-ihave-04**

### **Status of this Memo**

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and 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/1id-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 June 28, 2009.

### **Copyright Notice**

Copyright (c) 2008 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents in effect on the date of publication of this document (<http://trustee.ietf.org/license-info>). Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

### **Abstract**

This document describes the "ihave" extension to the Sieve email filtering language. The "ihave" extension provides a means to write scripts that can take advantage of optional Sieve features but can still run when those optional features are not available. The extension also defines a new error control command intended to be used to report situations where no combination of available extensions satisfies the needs of the script.

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

Changed the comparator used in the ihave test from "i;ascii-casemap" to "i;octet".

Updated the IANA registration template.

Simplified the semantics of ihave to be independent of block structure.

Moved the environment extension to a separate document so the standards status of the two extensions can be different.

Added error action.

Added some text to make the portability advantages of ihave clearer.

Added a note about the possibility that the argument to error uses UTF-8 characters.

(from WGLC) Various editorial fixups.

(from WGLC) Incorporated the same short-circuit, left to right requirements the variables extension imposes, because without it invocation of the variables extension could potentially change the meaning of ihave constructs in anyof or allof clauses.

(from WGLC) Added a restriction that ihave MUST NOT be used with any extension that changes the underlying Sieve grammar. Hopefully there won't be any such extensions, but better safe than sorry.

---

## 1. Introduction

Sieve [\[RFC5228\]](#) (Guenther, P. and T. Showalter, "Sieve: An Email Filtering Language," January 2008.) is a language for filtering email messages at or around the time of final delivery. It is designed to be implementable on either a mail client or mail server. It is suitable for running on a mail server where users may not be allowed to execute arbitrary programs, such as on black box Internet Message Access Protocol [\[RFC3501\]](#) (Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1," March 2003.) servers, as it has no user-controlled loops or the ability to run external programs.

Various sieve extensions have already been defined, e.g., [\[RFC5229\]](#) (Homme, K., "Sieve Email Filtering: Variables Extension," January 2008.) [\[RFC5230\]](#) (Showalter, T. and N. Freed, "Sieve Email Filtering: Vacation Extension," January 2008.) [\[RFC5231\]](#) (Segmuller, W. and B. Leiba, "Sieve Email Filtering: Relational Extension," January 2008.) [\[RFC5232\]](#) (Melnikov, A., "Sieve Email Filtering: Imap4flags Extension," January 2008.) [\[RFC5233\]](#) (Murchison, K., "Sieve Email Filtering: Subaddress Extension," January 2008.) [\[RFC5235\]](#) (Daboo, C., "Sieve Email Filtering: Spamtest and Virustest Extensions," January 2008.), and many more are sure to be created over time. Sieve's require clause is used to specify the extensions a particular sieve needs; an error results if the script's require clause calls for an extension that isn't available. This mechanism is sufficient in most situations. However, there can be cases where a script may be able to

take advantage of an extension if it is available but can still operate if it is not, possibly with some degradation of functionality. Cases can also arise where a script would prefer one extension but can employ a different one if the first one is not available.

The "ihave" extension provides a means to write scripts that make use of extensions only when they are actually available. It defines a new ihave test that takes a list of capability names as an argument and succeeds if and only if all of those capabilities are present.

Additionally, specification of the "ihave" extension in the require clause disables parse time checking of extension use in scripts; run-time checking must be used instead. This makes it possible to write portable scripts that can operate in multiple environments making effective use of whatever extensions are available even though differing sets of extensions are provided in different places.

The "ihave" extension also defines a new error control command. Error causes script execution to terminate with the error message given as the argument to the error control.

---

## **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 [\[RFC2119\] \(Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," March 1997.\)](#).

The terms used to describe the various components of the Sieve language are taken from Section 1.1 of [\[RFC5228\] \(Guenther, P. and T. Showalter, "Sieve: An Email Filtering Language," January 2008.\)](#).

---

## **3. Capability Identifiers**

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

---

## **4. Ihave Test**

Usage:   ihave <capabilities: string-list>

The ihave test provides a means for Sieve scripts to test for the existence of a given extension prior to actually using it. The

capabilities argument to `ihave` is the same as the similarly-named argument to the `require` control statement: It specifies the names of one or more Sieve extensions or comparators. The `ihave` test succeeds if all the extensions specified in the capabilities list are available to the script.

Unlike most Sieve tests, `ihave` accepts no match or comparator arguments. The type of match for `ihave` is always `:is` and the comparator is always `i;octet`.

The strings in the capabilities list are constant strings in the context of Sieve variables [\[RFC5229\] \(Homme, K., "Sieve Email Filtering: Variables Extension," January 2008.\)](#). It is an error to pass a non-constant string as an argument to `ihave`.

The Sieve base specification demands that all Sieve extensions used in a given script be specified in the initial `require` control statement. It is an error for a script to call for extensions the interpreter doesn't support or to attempt to use extensions that have not been listed in the script's `require` clause. Using `ihave` changes Sieve interpreter behavior and the underlying requirements in the following ways:

1. Use of a given extension is allowed subsequent to the successful evaluation of an `ihave` test on that extension all the way to the end of the script, even outside the block enclosed by the `ihave` test. In other words, subsequent to a successful `ihave` things operate just as if the extension had been specified in the script's `require` clause. The extension cannot be used prior to the evaluation of such a test and a runtime error **MUST** be generated if such usage is attempted. However, subsequent use of that extension may still need to be conditionally handled via an `ihave` test to deal with the case where it is not supported.
2. Sieve interpreters normally have the option of checking extension use at either parse time or execution time. The specification of `"ihave"` in a script's `require` clause changes this behavior: Scripts **MUST** either defer extension checking to run time or else take the presence of `ihave` tests into account at parse time. Note that since `ihave` can be used inside of `anyof`, `allof`, and `not` tests full parse time checking of `ihave` may be very difficult to implement.
3. Although it makes little sense to do so, an extension can be specified in both the `require` control statement and in an `ihave` test. If this is done and the extension has been implemented the extension can be used anywhere in the script and an `ihave` test of that extension will always return true.

4. The `ihave` test accepts a list of capabilities. The test fails and none of the capabilities are enabled if any of the specified capabilities are unavailable.
5. The Sieve base specification does not require that interpreters evaluate arguments in any particular order or that test evaluation be short-circuited. If `ihave` is enabled the interpreter **MUST** short-circuit tests, i.e., not perform more tests than necessary to find the result. Additionally, evaluation order **MUST** be left to right if `ihave` is enabled.

The `ihave` extension is designed to be used with other extensions that add tests, actions, comparators, or arguments. Implementations **MUST NOT** allow it to be used with extensions that change the underlying Sieve grammar or extensions like encoded-character [\[RFC5228\] \(Guenther, P. and T. Showalter, "Sieve: An Email Filtering Language," January 2008.\)](#) or variables [\[RFC5229\] \(Homme, K., "Sieve Email Filtering: Variables Extension," January 2008.\)](#) that change how the content of Sieve scripts are interpreted. The test **MUST** fail and the extension **MUST NOT** be enabled if such usage is attempted.

---

## 5. Error Control

Usage: `error <message: string>`

The error control causes script execution to terminate with a run-time error. The message argument provides a text description of the error condition that **SHOULD** be included in any generated report regarding the error. Section 2.10.6 of [\[RFC5228\] \(Guenther, P. and T. Showalter, "Sieve: An Email Filtering Language," January 2008.\)](#) describes how run-time errors are handled in Sieve.

Note that the message argument, like all Sieve strings, employs the UTF-8 charset and can contain non-US-ASCII characters. This must be taken into consideration when reporting script errors.

The error control is included as part of the `ihave` extension so that it is unconditionally available to scripts using `ihave`.

---

## 6. Security Considerations

A potential security issue with Sieve scripts is that when a script fails to run due to the lack of some extension it may fail to block dangerous email. The `ihave` extension makes it possible to improve

script portability and generality, which may improve the overall security provided by Sieve.

Script robustness aside, ihave is essentially a more flexible variant of Sieve's existing require mechanism. As such, it does not add any additional capabilities to a Sieve implementation that could create security issues. Of course all of the security considerations given in the base Sieve specification and in any extensions that are employed are still relevant.

---

## 7. IANA Considerations

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

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

Capability name: ihave  
Description: The "ihave" extension provides a means to write scripts that make use of other extensions only when they are actually available.  
RFC number: RFC XXXX  
Contact address: Sieve discussion list <ietf-mta-filters@imc.org>

---

## 8. References

### 8.1. Normative references

[RFC2119]	<a href="#">Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels,"</a> BCP 14, RFC 2119, March 1997 ( <a href="#">TXT</a> , <a href="#">HTML</a> , <a href="#">XML</a> ).
[RFC5228]	Guenther, P. and T. Showalter, " <a href="#">Sieve: An Email Filtering Language</a> ," RFC 5228, January 2008 ( <a href="#">TXT</a> ).

### 8.2. Informative references

[RFC3501]	Crispin, M., " <a href="#">INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1</a> ," RFC 3501, March 2003 ( <a href="#">TXT</a> ).
-----------	--

[RFC5229]	Homme, K., " <a href="#">Sieve Email Filtering: Variables Extension</a> ," RFC 5229, January 2008 ( <a href="#">TXT</a> ).
[RFC5230]	Showalter, T. and N. Freed, " <a href="#">Sieve Email Filtering: Vacation Extension</a> ," RFC 5230, January 2008 ( <a href="#">TXT</a> ).
[RFC5231]	Segmuller, W. and B. Leiba, " <a href="#">Sieve Email Filtering: Relational Extension</a> ," RFC 5231, January 2008 ( <a href="#">TXT</a> ).
[RFC5232]	Melnikov, A., " <a href="#">Sieve Email Filtering: Imap4flags Extension</a> ," RFC 5232, January 2008 ( <a href="#">TXT</a> ).
[RFC5233]	Murchison, K., " <a href="#">Sieve Email Filtering: Subaddress Extension</a> ," RFC 5233, January 2008 ( <a href="#">TXT</a> ).
[RFC5235]	Daboo, C., " <a href="#">Sieve Email Filtering: Spamtest and Virustest Extensions</a> ," RFC 5235, January 2008 ( <a href="#">TXT</a> ).

---

## Appendix A. Acknowledgements

Stephan Bosch, Cyrus Daboo, Arnt Gulbrandsen, Andrew McKeon, and Alexey Melnikov provided helpful suggestions and corrections.

---

## Author's Address

	Ned Freed
	Sun Microsystems
	800 Royal Oaks
	Monrovia, CA 91016-6347
	USA
Phone:	+1 909 457 4293
Email:	<a href="mailto:ned.freed@mrochek.com">ned.freed@mrochek.com</a>