Network Working Group
Request for Comments: 5161
Category: Standards Track

A. Gulbrandsen, Ed.
Oryx Mail Systems GmbH
A. Melnikov, Ed.
Isode Limited

March 2008

The IMAP ENABLE Extension

Status of This Memo

Abstract

Most IMAP extensions are used by the client when it wants to and the server supports it. However, a few extensions require the server to know whether a client supports that extension. The ENABLE extension allows an IMAP client to say which extensions it supports.

1. Overview

Several IMAP extensions allow the server to return unsolicited responses specific to these extensions in certain circumstances. However, servers cannot send those unsolicited responses until they know that the clients support such extensions and thus won’t choke on the extension response data.

Up until now, extensions have typically stated that a server cannot send the unsolicited responses until after the client has used a command with the extension data (i.e., at that point the server knows the client is aware of the extension). CONDSTORE ([RFC4551]), ANNOTATE ([ANNOTATE]), and some extensions under consideration at the moment use various commands to enable server extensions. For example, CONDSTORE uses a SELECT or FETCH parameter, and ANNOTATE uses a side effect of FETCH.

The ENABLE extension provides an explicit indication from the client that it supports particular extensions. This is done using a new ENABLE command.

An IMAP server that supports ENABLE advertises this by including the word ENABLE in its capability list.

Most IMAP extensions do not require the client to enable the extension in any way.

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].

Formal syntax is defined by [RFC5234] and [RFC3501].

Example lines prefaced by "C:" are sent by the client and ones prefaced by "S:" by the server. The five characters [...] means that something has been elided.

3. Protocol Changes

3.1. The ENABLE Command

Arguments: capability names

Result: OK: Relevant capabilities enabled
BAD: No arguments, or syntax error in an argument

The ENABLE command takes a list of capability names, and requests the server to enable the named extensions. Once enabled using ENABLE, each extension remains active until the IMAP connection is closed. For each argument, the server does the following:

- If the argument is not an extension known to the server, the server MUST ignore the argument.

- If the argument is an extension known to the server, and it is not specifically permitted to be enabled using ENABLE, the server MUST ignore the argument. (Note that knowing about an extension doesn’t necessarily imply supporting that extension.)

- If the argument is an extension that is supported by the server and that needs to be enabled, the server MUST enable the extension for the duration of the connection. At present, this applies only to CONDSTORE ([RFC4551]). Note that once an extension is enabled, there is no way to disable it.

If the ENABLE command is successful, the server MUST send an untagged ENABLED response (see Section 3.2).

Clients SHOULD only include extensions that need to be enabled by the server. At the time of publication, CONDSTORE is the only such extension (i.e., ENABLE CONDSTORE is an additional "CONDSTORE enabling command" as defined in [RFC4551]). Future RFCs may add to this list.

The ENABLE command is only valid in the authenticated state (see [RFC3501]), before any mailbox is selected. Clients MUST NOT issue ENABLE once they SELECT/EXAMINE a mailbox; however, server implementations don't have to check that no mailbox is selected or was previously selected during the duration of a connection.

The ENABLE command can be issued multiple times in a session. It is additive; i.e., "ENABLE a b", followed by "ENABLE c" is the same as a single command "ENABLE a b c". When multiple ENABLE commands are issued, each corresponding ENABLED response SHOULD only contain extensions enabled by the corresponding ENABLE command.

There are no limitations on pipelining ENABLE. For example, it is possible to send ENABLE and then immediately SELECT, or a LOGIN immediately followed by ENABLE.

The server MUST NOT change the CAPABILITY list as a result of executing ENABLE; i.e., a CAPABILITY command issued right after an ENABLE command MUST list the same capabilities as a CAPABILITY command issued before the ENABLE command. This is demonstrated in the following example:

```
C: t1 CAPABILITY
S: * CAPABILITY IMAP4rev1 ID LITERAL+ ENABLE X-GOOD-IDEA
S: t1 OK foo
C: t2 ENABLE CONDSTORE X-GOOD-IDEA
S: * ENABLED X-GOOD-IDEA
S: t2 OK foo
C: t3 CAPABILITY
S: * CAPABILITY IMAP4rev1 ID LITERAL+ ENABLE X-GOOD-IDEA
S: t3 OK foo again
```

In the following example, the client enables CONDSTORE:

```
C: a1 ENABLE CONDSTORE
S: * ENABLED CONDSTORE
S: a1 OK Conditional Store enabled
```

3.2. The ENABLED Response

Contents: capability listing

The ENABLED response occurs as a result of an ENABLE command. The capability listing contains a space-separated listing of capability names that the server supports and that were successfully enabled. The ENABLED response may contain no capabilities, which means that no extensions listed by the client were successfully enabled.

3.3. Note to Designers of Extensions That May Use the ENABLE Command

Designers of IMAP extensions are discouraged from creating extensions that require ENABLE unless there is no good alternative design. Specifically, extensions that cause potentially incompatible behavior changes to deployed server responses (and thus benefit from ENABLE) have a higher complexity cost than extensions that do not.

4. Formal Syntax

The following syntax specification uses the Augmented Backus-Naur Form (ABNF) notation as specified in [RFC5234] including the core rules in Appendix B.1. [RFC3501] defines the non-terminals "capability" and "command-any".

Except as noted otherwise, all alphabetic characters are case-insensitive. The use of upper or lower case characters to define token strings is for editorial clarity only. Implementations MUST accept these strings in a case-insensitive fashion.

```
capability    =/ "ENABLE"

command-any   =/ "ENABLE" 1*(SP capability)

response-data =/ "*" SP enable-data CRLF

enable-data   = "ENABLED" *(SP capability)
```

5. Security Considerations

It is believed that this extension doesn’t add any security considerations that are not already present in the base IMAP protocol [RFC3501].

6. IANA Considerations

The IANA has added ENABLE to the IMAP4 Capabilities Registry.

7. Acknowledgments

The editors would like to thank Randy Gellens, Chris Newman, Peter Coates, Dave Cridland, Mark Crispin, Ned Freed, Dan Karp, Cyrus Daboo, Ken Murchison, and Eric Burger for comments and corrections. However, this doesn't necessarily mean that they endorse this extension, agree with all details, or are responsible for errors introduced by the editors.

8. Normative References

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

[RFC3501] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, March 2003.

[RFC5234] Crocker, D., Ed., and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, January 2008.

[RFC4551] Melnikov, A. and S. Hole, "IMAP Extension for Conditional STORE Operation or Quick Flag Changes Resynchronization", RFC 4551, June 2006.

9. Informative References

[ANNOTATE] Daboo, C. and R. Gellens, "IMAP ANNOTATE Extension", Work in Progress, August 2006.

Editors' Addresses

Arnt Gulbrandsen
Oryx Mail Systems GmbH
Schweppermannstr. 8
D-81671 Muenchen
Germany

Fax: +49 89 4502 9758
EMail: arnt@oryx.com

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

EMail: Alexey.Melnikov@isode.com

Full Copyright Statement

Intellectual Property