Network Working Group
Request for Comments: 5530
Category: Standards Track

A. Gulbrandsen
Oryx Mail Systems GmbH

May 2009

IMAP Response Codes

Status of This Memo

Copyright Notice

Abstract

IMAP responses consist of a response type (OK, NO, BAD), an optional machine-readable response code, and a human-readable text.

This document collects and documents a variety of machine-readable response codes, for better interoperation and error reporting.

1. Introduction

Section 7.1 of [RFC3501] defines a number of response codes that can help tell an IMAP client why a command failed. However, experience has shown that more codes are useful. For example, it is useful for a client to know that an authentication attempt failed because of a server problem as opposed to a password problem.

Currently, many IMAP servers use English-language, human-readable text to describe these errors, and a few IMAP clients attempt to translate this text into the user's language.

This document names a variety of errors as response codes. It is based on errors that have been checked and reported on in some IMAP server implementations, and on the needs of some IMAP clients.

This document doesn't require any servers to test for these errors or any clients to test for these names. It only names errors for better reporting and handling.

2. Conventions Used in This Document

Formal syntax is defined by [RFC5234] as modified by [RFC3501].

Example lines prefaced by "C:" are sent by the client and ones prefaced by "S:" by the server. "[...]" means elision.

3. Response Codes

This section defines all the new response codes. Each definition is followed by one or more examples.

UNAVAILABLE
: Temporary failure because a subsystem is down. For example, an IMAP server that uses a Lightweight Directory Access Protocol (LDAP) or Radius server for authentication might use this response code when the LDAP/Radius server is down.

```
C: a LOGIN "fred" "foo"
S: a NO [UNAVAILABLE] User's backend down for maintenance
```

AUTHENTICATIONFAILED
: Authentication failed for some reason on which the server is unwilling to elaborate. Typically, this includes "unknown user" and "bad password".

This is the same as not sending any response code, except that when a client sees AUTHENTICATIONFAILED, it knows that the problem wasn't, e.g., UNAVAILABLE, so there's no point in trying the same login/password again later.

```
C: b LOGIN "fred" "foo"
S: b NO [AUTHENTICATIONFAILED] Authentication failed
```

AUTHORIZATIONFAILED

Authentication succeeded in using the authentication identity, but the server cannot or will not allow the authentication identity to act as the requested authorization identity. This is only applicable when the authentication and authorization identities are different.

```
C: c1 AUTHENTICATE PLAIN
[...]
S: c1 NO [AUTHORIZATIONFAILED] No such authorization-ID

C: c2 AUTHENTICATE PLAIN
[...]
S: c2 NO [AUTHORIZATIONFAILED] Authenticator is not an admin
```

EXPIRED

Either authentication succeeded or the server no longer had the necessary data; either way, access is no longer permitted using that passphrase. The client or user should get a new passphrase.

```
C: d login "fred" "foo"
S: d NO [EXPIRED] That password isn't valid any more
```

PRIVACYREQUIRED

The operation is not permitted due to a lack of privacy. If Transport Layer Security (TLS) is not in use, the client could try STARTTLS (see Section 6.2.1 of [RFC3501]) and then repeat the operation.

```
C: d login "fred" "foo"
S: d NO [PRIVACYREQUIRED] Connection offers no privacy

C: d select inbox
S: d NO [PRIVACYREQUIRED] Connection offers no privacy
```

CONTACTADMIN
The user should contact the system administrator or support desk.

```
C: e login "fred" "foo"
S: e OK [CONTACTADMIN]
```

NOPERM
The access control system (e.g., Access Control List (ACL), see [RFC4314]) does not permit this user to carry out an operation, such as selecting or creating a mailbox.

```
C: f select "/archive/projects/experiment-iv"
S: f NO [NOPERM] Access denied
```

INUSE
An operation has not been carried out because it involves sawing off a branch someone else is sitting on. Someone else may be holding an exclusive lock needed for this operation, or the operation may involve deleting a resource someone else is using, typically a mailbox.

The operation may succeed if the client tries again later.

```
C: g delete "/archive/projects/experiment-iv"
S: g NO [INUSE] Mailbox in use
```

EXPUNGEISSUED
Someone else has issued an EXPUNGE for the same mailbox. The client may want to issue NOOP soon. [RFC2180] discusses this subject in depth.

```
C: h search from fred@example.com
S: * SEARCH 1 2 3 5 8 13 21 42
S: h OK [EXPUNGEISSUED] Search completed
```

CORRUPTION
The server discovered that some relevant data (e.g., the mailbox) are corrupt. This response code does not include any information about what's corrupt, but the server can write that to its logfiles.

```
C: i select "/archive/projects/experiment-iv"
S: i NO [CORRUPTION] Cannot open mailbox
```

SERVERBUG
The server encountered a bug in itself or violated one of its own invariants.

```
C: j select "/archive/projects/experiment-iv"
S: j NO [SERVERBUG] This should not happen
```

CLIENTBUG
The server has detected a client bug. This can accompany all of OK, NO, and BAD, depending on what the client bug is.

```
C: k1 select "/archive/projects/experiment-iv"
[...]
S: k1 OK [READ-ONLY] Done
C: k2 status "/archive/projects/experiment-iv" (messages)
[...]
S: k2 OK [CLIENTBUG] Done
```

CANNOT
The operation violates some invariant of the server and can never succeed.

```
C: l create "///////"
S: l NO [CANNOT] Adjacent slashes are not supported
```

LIMIT
The operation ran up against an implementation limit of some kind, such as the number of flags on a single message or the number of flags used in a mailbox.

```
C: m STORE 42 FLAGS f1 f2 f3 f4 f5 ... f250
S: m NO [LIMIT] At most 32 flags in one mailbox supported
```

OVERQUOTA
The user would be over quota after the operation. (The user may or may not be over quota already.)

Note that if the server sends OVERQUOTA but doesn’t support the IMAP QUOTA extension defined by [RFC2087], then there is a quota, but the client cannot find out what the quota is.

```
C: n1 uid copy 1:* oldmail
S: n1 NO [OVERQUOTA] Sorry

C: n2 uid copy 1:* oldmail
S: n2 OK [OVERQUOTA] You are now over your soft quota
```

ALREADYEXISTS
The operation attempts to create something that already exists, such as when the CREATE or RENAME directories attempt to create a mailbox and there is already one of that name.

C: o RENAME this that
S: o NO [ALREADYEXISTS] Mailbox "that" already exists

NONEXISTENT
The operation attempts to delete something that does not exist. Similar to ALREADYEXISTS.

C: p RENAME this that
S: p NO [NONEXISTENT] No such mailbox

4. Formal Syntax

The following syntax specification uses the Augmented Backus-Naur Form (ABNF) notation as specified in [RFC5234]. [RFC3501] defines the non-terminal "resp-text-code".

Except as noted otherwise, all alphabetic characters are case-insensitive. The use of upper or lowercase characters to define token strings is for editorial clarity only.

```
resp-text-code =/ "UNAVAILABLE" / "AUTHENTICATIONFAILED" /
                  "AUTHORIZATIONFAILED" / "EXPIRED" /
                  "PRIVACYREQUIRED" / "CONTACTADMIN" / "NOPERM" /
                  "INUSE" / "EXPUNGEISSUED" / "CORRUPTION" /
                  "SERVERBUG" / "CLIENTBUG" / "CANNOT" /
                  "LIMIT" / "OVERQUOTA" / "ALREADYEXISTS" /
                  "NONEXISTENT"
```

5. Security Considerations

Revealing information about a passphrase to unauthenticated IMAP clients causes bad karma.

Response codes are easier to parse than human-readable text. This can amplify the consequences of an information leak. For example, selecting a mailbox can fail because the mailbox doesn't exist, because the user doesn't have the "l" right (right to know the mailbox exists) or "r" right (right to read the mailbox). If the server sent different responses in the first two cases in the past, only malevolent clients would discover it. With response codes it's possible, perhaps probable, that benevolent clients will forward the

leaked information to the user. Server authors are encouraged to be particularly careful with the NOPERM and authentication-related responses.

6. IANA Considerations

The IANA has created the IMAP Response Codes registry. The registry has been populated with the following codes:

NEWNAME	RFC 2060 (obsolete)
REFERRAL	RFC 2221
ALERT	RFC 3501
BADCHARSET	RFC 3501
PARSE	RFC 3501
PERMANENTFLAGS	RFC 3501
READ-ONLY	RFC 3501
READ-WRITE	RFC 3501
TRYCREATE	RFC 3501
UIDNEXT	RFC 3501
UIDVALIDITY	RFC 3501
UNSEEN	RFC 3501
UNKNOWN-CTE	RFC 3516
UIDNOTSTICKY	RFC 4315
APPENDUID	RFC 4315
COPYUID	RFC 4315
URLMECH	RFC 4467
TOOBIG	RFC 4469
BADURL	RFC 4469
HIGHESTMODSEQ	RFC 4551
NOMODSEQ	RFC 4551
MODIFIED	RFC 4551
COMPRESSIONACTIVE	RFC 4978
CLOSED	RFC 5162
NOTSAVED	RFC 5182
BADCOMPARATOR	RFC 5255
ANNOTATE	RFC 5257
ANNOTATIONS	RFC 5257
TEMPFAIL	RFC 5259
MAXCONVERTMESSAGES	RFC 5259
MAXCONVERTPARTS	RFC 5259
NOUPDATE	RFC 5267
METADATA	RFC 5464
NOTIFICATIONOVERFLOW	RFC 5465
BADEVENT	RFC 5465
UNDEFINED-FILTER	RFC 5466
UNAVAILABLE	RFC 5530
AUTHENTICATIONFAILED	RFC 5530
AUTHORIZATIONFAILED	RFC 5530

EXPIRED	RFC 5530
PRIVACYREQUIRED	RFC 5530
CONTACTADMIN	RFC 5530
NOPERM	RFC 5530
INUSE	RFC 5530
EXPUNGEISSUED	RFC 5530
CORRUPTION	RFC 5530
SERVERBUG	RFC 5530
CLIENTBUG	RFC 5530
CANNOT	RFC 5530
LIMIT	RFC 5530
OVERQUOTA	RFC 5530
ALREADYEXISTS	RFC 5530
NONEXISTENT	RFC 5530

The new registry can be extended by sending a registration request to IANA. IANA will forward this request to a Designated Expert, appointed by the responsible IESG Area Director, CCing it to the IMAP Extensions mailing list at <ietf-imapext@imc.org> (or a successor designated by the Area Director). After either allowing 30 days for community input on the IMAP Extensions mailing list or a successful IETF Last Call, the expert will determine the appropriateness of the registration request and either approve or disapprove the request by sending a notice of the decision to the requestor, CCing the IMAP Extensions mailing list and IANA. A denial notice must be justified by an explanation, and, in cases where it is possible, concrete suggestions on how the request can be modified so as to become acceptable should be provided.

For each response code, the registry contains a list of relevant RFCs that describe (or extend) the response code and an optional response code status description, such as "obsolete" or "reserved to prevent collision with deployed software". (Note that in the latter case, the RFC number can be missing.) Presence of the response code status description means that the corresponding response code is NOT RECOMMENDED for widespread use.

The intention is that any future allocation will be accompanied by a published RFC (including direct submissions to the RFC Editor). But in order to allow for the allocation of values prior to the RFC being approved for publication, the Designated Expert can approve allocations once it seems clear that an RFC will be published, for example, before requesting IETF LC for the document.

The Designated Expert can also approve registrations for response codes used in deployed software when no RFC exists. Such registrations must be marked as "reserved to prevent collision with deployed software".

Response code registrations may not be deleted; response codes that are no longer believed appropriate for use (for example, if there is a problem with the syntax of said response code or if the specification describing it was moved to Historic) should be marked "obsolete" in the registry, clearly marking the lists published by IANA.

7. Acknowledgements

Peter Coates, Mark Crispin, Philip Guenther, Alexey Melnikov, Ken Murchison, Chris Newman, Timo Sirainen, Philip Van Hoof, Dale Wiggins, and Sarah Wilkin helped with this document.

8. References

8.1. Normative References

[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.

9. Informative References

[RFC2087] Myers, J., "IMAP4 QUOTA extension", RFC 2087, January 1997.

[RFC2180] Gahrns, M., "IMAP4 Multi-Accessed Mailbox Practice", RFC 2180, July 1997.

[RFC4314] Melnikov, A., "IMAP4 Access Control List (ACL) Extension", RFC 4314, December 2005.

Author's Address

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

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