Internet Draft
Document: draft-melnikov-imap-ext-abnf
Expires: April 2006

A. Melnikov
Isode Ltd.
Cyrus Daboo
ISAMET, Inc.
October 2005

Collected extensions to IMAP4 ABNF
draft-melnikov-imap-ext-abnf-05

Status of this Memo

A revised version of this draft document will be submitted to the RFC editor as a Standard Track RFC for the Internet Community. Discussion and suggestions for improvement are requested, and should be sent to ietf-imapext@imc.org and/or lemonade@ietf.org. Distribution of this memo is unlimited.

Abstract

Over years many documents from IMAPEXT and LEMONADE working groups, as well as many individual documents have added syntactic extensions to many base IMAP commands described in RFC 3501. For ease of reference this document collects most of such ABNF changes in one place.

This document updates ABNF in RFC 3501, RFC 2342 and RFC 2088.

Table of Contents

1. Conventions Used in this Document

In examples, "C:" and "S:" indicate lines sent by the client and server respectively.

The key words "MUST", "MUST NOT", "SHOULD", "SHOULD NOT", and "MAY" in this document are to be interpreted as defined in "Key words for use in RFCs to Indicate Requirement Levels" [KEYWORDS].

<<Editorial comments and questions are enclosed like this>>

2. IMAP ABNF extensions

This section is not normative. It provides with some background on intended use of different extensions and it tries to give some guidance about how future extensions should extend the described commands.

2.1 Optional parameters with the SELECT/EXAMINE commands

This documents adds the ability to include one or more parameters with the IMAP SELECT or EXAMINE commands, to turn on or off certain standard behaviour, or to add new optional behaviours required for a particular extension.

There are two possible modes of operation:

o A global state change where a single use of the optional
   parameter will effect the session state from that time on,
   irrespective of subsequent SELECT/EXAMINE commands.

o A per-mailbox state change that will effect the session only for
   the duration of the new selected state. A subsequent SELECT/
   EXAMINE without the optional parameter will cancel its effect
   For the newly selected mailbox.

Optional parameters to the SELECT or EXAMINE commands are added as a parenthesised list of atoms or strings, and appear after the mailbox name in the standard SELECT or EXAMINE command. The order of individual parameters is arbitrary. Individual parameters may consist of one or more atoms or strings in a specific order. If a parameter consists of more than one atom or string, it SHOULD appear in its own parenthesised list. Any parameter not defined by extensions that the server supports must be rejected with a BAD response.

Example:

```
        C: a SELECT INBOX (ANNOTATE)
        S: ...
        S: a OK SELECT complete
```

In the above example, a single parameter is used with the SELECT command.

Example:

```
        C: a EXAMINE INBOX (ANNOTATE RESPONSES "UID Responses"
           CONDSTORE)
        S: ...
        S: a OK EXAMINE complete
```

In the above example, three parameters are used with the EXAMINE command. The second parameter consists of two items: an atom followed by a quoted string.

Example:

```
        C: a SELECT INBOX (BLURDYBLOOP)
        S: a BAD Unknown parameter in SELECT command
```

In the above example, a parameter not supported by the server is used. This results in the BAD response from the server.

2.2 Extended CREATE command

Arguments: mailbox name

```
             OPTIONAL list of CREATE parameters

Responses:   no specific responses for this command

Result:      OK - create completed
             NO - create failure: can't create mailbox with
                  that name
             BAD - argument(s) invalid
```

This documents adds the ability to include one or more parameters with the IMAP CREATE command (see section 6.3.3 of [IMAP4]), to turn on or off certain standard behaviour, or to add new optional behaviours required for a particular extension. No CREATE parameters are defined in this document.

Optional parameters to the CREATE command are added as a parenthesised list of attribute/value pairs after the mailbox name. Each value can be either an atom, a string or a list. The order of individual parameters is arbitrary. Individual parameters may consist of one or more atoms or strings in a specific order. If a parameter consists of more than one atom or string, it SHOULD appear in its own parenthesised list. Any parameter not defined by extensions that the server supports must be rejected with a BAD response.

2.3 Extended RENAME command

```
Arguments:   existing mailbox name
             new mailbox name
             OPTIONAL list of RENAME parameters

Responses:   no specific responses for this command

Result:      OK - rename completed
             NO - rename failure: can't rename mailbox with
                  that name, can't rename to mailbox with
                  that name, etc.
             BAD - argument(s) invalid
```

This documents adds the ability to include one or more parameters with the IMAP RENAME command (see section 6.3.5 of [IMAP4]), to turn on or off certain standard behaviour, or to add new optional behaviours required for a particular extension. No RENAME parameters are defined in this document.

Optional parameters to the RENAME command are added as a parenthesised list of attribute/value pairs after the new mailbox name. Each value can be either an atom, a string or a list. The order of individual parameters is arbitrary. Individual parameters may consist of one or more atoms or strings in a specific order. If a parameter consists of more than one atom or string, it SHOULD appear in its own parenthesised list. Any parameter not defined by extensions that the server supports must be rejected with a BAD response.

2.4 Extensions to FETCH and UID FETCH Commands

```
 Arguments:  sequence set
             message data item names or macro
             OPTIONAL fetch modifiers

 Responses:  untagged responses: FETCH

 Result:     OK - fetch completed
             NO - fetch error: can't fetch that data
             BAD - command unknown or arguments invalid
```

This document extends the syntax of the FETCH and UID FETCH commands (see section 6.4.5 of [IMAP4]) to include optional FETCH modifiers. No fetch modifiers are defined in this document.

The order of individual modifiers is arbitrary. An individual modifier may consist of one or more atoms or strings in a specific order. If a modifier consists of more than one atom or string, it SHOULD appear in its own parenthesised list. Any modifiers not defined by extensions that the server supports must be rejected with a BAD response.

2.5 Extensions to STORE and UID STORE Commands

```
Arguments:  message set
            OPTIONAL store modifiers
            message data item name
            value for message data item

Responses:  untagged responses: FETCH

Result:     OK - store completed
            NO - store error: can't store that data
            BAD - command unknown or arguments invalid
```

This document extends the syntax of the STORE and UID STORE commands (see section 6.4.6 of [IMAP4]) to include optional STORE modifiers. No store modifiers are defined in this document.

The order of individual modifiers is arbitrary. Individual modifier may consist of one or more atoms or strings in a specific order. If a modifier consists of more than one atom or string, it SHOULD appear in its own parenthesised list. Any modifiers not defined by extensions that the server supports must be rejected with a BAD response.

2.6 Extensions to SEARCH Command

2.6.1 Extended SEARCH command

```
Arguments:  OPTIONAL result specifier
            OPTIONAL [CHARSET] specification
            searching criteria (one or more)

Responses:  REQUIRED untagged response: SEARCH (*)

Result:     OK - search completed
            NO - search error: can't search that [CHARSET] or
                 criteria
            BAD - command unknown or arguments invalid
```

This section updates definition of the SEARCH command described in section 6.4.4 of [IMAP4].

The SEARCH command is extended to allow for result options. This document doesn't define any result option.

The order of individual options is arbitrary. Individual options may optionally contain parameters enclosed in parentheses. If an option has parameters, they consist of one or more atoms or strings in a specific order. Any options not defined by extensions that the server supports must be rejected with a BAD response.

(*) - An extension to SEARCH command may require another untagged response, or no untagged response to be returned.

2.6.2 ESEARCH untagged response

```
Contents:   one or more search-return-data pairs
```

The ESEARCH response SHOULD be sent as a result of an extended SEARCH or UID SEARCH command specified in section 2.6.1.

The ESEARCH response is immediately followed by an optional search correlator. If it is missing than the response was not caused by a particular IMAP command, if it is present than it contains the tag of the command that caused the response to be returned.

The search correlator is followed by an optional UID indicator. If this indicator is present, all data in the ESEARCH response is referring to UIDs, otherwise all returned data is referring to message numbers.

The rest of the ESEARCH response contains one or more search data pairs. Each pair starts with unique return item name, followed by a space and the corresponding data. Search data pairs may be returned in any order. Unless specified otherwise by an extension, any return item name SHOULD appear only once in an ESEARCH response.

```
Example:    S: * ESEARCH UID COUNT 5 ALL 4:19,21,28

Example:    S: * ESEARCH (TAG "a567") UID COUNT 5 ALL 4:19,21,28

Example:    S: * ESEARCH COUNT 5 ALL 1:17,21
```

2.7 Extensions to APPEND Command

The APPEND command is extended to allow the client to append data containing NULs by using the <literal8> syntax. The ABNF was rewritten to allow for easier extensibility by IMAP extensions.

3. Formal Syntax

The following syntax specification uses the Augmented Backus-Naur Form (ABNF) notation as specified in [ABNF].

Non-terminals referenced but not defined below are as defined by [IMAP4].

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.

```
append          = "APPEND" SP mailbox 1*append-message
                  ;; only a single append-message may appear
                  ;; if MULTIAPPEND [MULTIAPPEND] capability
                  ;; is not present

append-message  = append-opts SP append-data

append-ext      = <for extension only>
                  ;; This rule exists solely to be augmented by
                  ;; extensions via incremental alternative
                  ;; ("=/") rules.  It is strongly recommended
                  ;; that such extensions match a subset of the
                  ;; tagged-ext rule syntax

append-data     = literal / literal8
                  ;; IMAP extensions extending append-data
                  ;; should use the tagged-ext syntax,
                  ;; i.e. a mandatory label followed
                  ;; by parameters.

append-opts     = [SP flag-list] [SP date-time] *(SP append-ext)


create          = "CREATE" SP mailbox
                  [create-params]
                  ;; Use of INBOX gives a NO error

create-params   = SP "(" create-param *( SP create-param) ")"

create-param-name = tagged-ext-label

create-param      = create-param-name SP create-param-value

create-param-value= <for extension only>
                  ;; This rule exists solely to be augmented by
                  ;; extensions via incremental alternative
                  ;; ("=/") rules.  It is strongly recommended
                  ;; that such extensions match a subset of the
                  ;; tagged-ext-val rule syntax

esearch-response  = "ESEARCH" [search-correlator] [SP "UID"]
                     *(SP search-return-data)
                   ;; Note that SEARCH and ESEARCH responses
                   ;; SHOULD be mutually exclusive,
                   ;; i.e. only one of them should be
                   ;; returned as a result of a command.


examine         = "EXAMINE" SP mailbox [select-params]
                  ;; modifies the original IMAP EXAMINE command
                  ;; to accept optional parameters

fetch           = "FETCH" SP sequence-set SP ("ALL" / "FULL" /
                  "FAST" / fetch-att /
                  "(" fetch-att *(SP fetch-att) ")")
                  [SP fetch-modifiers]
                  ;; modifies the original IMAP4 FETCH command to
                  ;; accept optional modifiers

fetch-modifiers = "(" fetch-modifier *(SP fetch-modifier) ")"

fetch-modifier  = fetch-modifier-name [ SP fetch-modif-params ]
                  ;; Note that the original syntax defined
                  ;; in CONDSTORE was extended to allow
                  ;; for "()"

fetch-modif-params  = <for extension only>
                  ;; This rule exists solely to be augmented by
                  ;; extensions via incremental alternative
                  ;; ("=/") rules.  It is strongly recommended
                  ;; that such extensions match a subset of the
                  ;; tagged-ext-val rule syntax

fetch-modifier-name = tagged-ext-label

literal8        = "~{" number ["+"] "}" CRLF *OCTET
                   ;; A string that might contain NULs.
                   ;; <number> represents the number of OCTETs
                   ;; in the response string.
                   ;; The "+" is only allowed when both LITERAL+
                   ;; and BINARY are present.


mailbox-data      =/ Namespace-Response /
                     esearch-response

Namespace         = nil / "(" 1*Namespace-Descr ")"

Namespace-Command = "NAMESPACE"

Namespace-Descr   = "(" string SP
                       (DQUOTE QUOTED-CHAR DQUOTE / nil)
                        *(Namespace-Response-Extension) ")"

Namespace-Response-Extension = SP string SP
                  "(" string *(SP string) ")"

Namespace-Response = "NAMESPACE" SP Namespace
                     SP Namespace SP Namespace
      ;; The first Namespace is the Personal Namespace(s)
      ;; The second Namespace is the Other Users' Namespace(s)
      ;; The third Namespace is the Shared Namespace(s)

rename          = "RENAME" SP mailbox SP mailbox
                  [rename-params]
                  ;; Use of INBOX as a destination gives
                  ;; a NO error

rename-params     = SP "(" rename-param *( SP rename-param) ")"

rename-param      = rename-param-name SP rename-param-value

rename-param-name = tagged-ext-label

rename-param-value= <for extension only>
                  ;; This rule exists solely to be augmented by
                  ;; extensions via incremental alternative
                  ;; ("=/") rules.  It is strongly recommended
                  ;; that such extensions match a subset of the
                  ;; tagged-ext-val rule syntax

response-data   = "*" SP response-payload CRLF
```

```
response-payload= resp-cond-state / resp-cond-bye /
                  mailbox-data / message-data / capability-data

search          = "SEARCH" [search-return-opts]
                  search-program


search-correlator  = SP "(" "TAG" SP tag-string ")"

search-program     = [SP "CHARSET" SP astring] 1*(SP search-key)
                     ;; CHARSET argument to SEARCH MUST be
                     ;; registered with IANA

search-return-data = search-modifier-name SP search-return-value
                     ;; Note that not every SEARCH return option
                     ;; is required to have the corresponding
                     ;; ESEARCH return data

search-return-opts = "RETURN" SP "(" [search-return-opt
                     *(SP search-return-opt)] ")"

search-return-opt = search-modifier-name [SP search-mod-params]

search-return-value= tagged-ext-val
                     ;; data for the returned search option.
                     ;; A single "nz-number"/"number" value
                     ;; can be returned as an atom (i.e. without
                     ;; quoting). A sequence-set can be returned
                     ;; as an atom as well.

search-modifier-name = tagged-ext-label

search-mod-params = <for extension only>
                  ;; This rule exists solely to be augmented by
                  ;; extensions via incremental alternative
                  ;; ("=/") rules.  It is strongly recommended
                  ;; that such extensions match a subset of the
                  ;; tagged-ext-val rule syntax

select          = "SELECT" SP mailbox [select-params]
                  ;; modifies the original IMAP SELECT command to
                  ;; accept optional parameters

select-params   = SP "(" select-param *(SP select-param) ")"

select-param    = select-param-name SP select-param-value
                  ;; parameters to SELECT may contain one or
                  ;; more atoms or strings - multiple items
                  ;; are always parenthesised

select-param-name= tagged-ext-label
```

```
select-param-value= <for extension only>
                  ;; This rule exists solely to be augmented by
                  ;; extensions via incremental alternative
                  ;; ("=/") rules.  It is strongly recommended
                  ;; that such extensions match a subset of the
                  ;; tagged-ext-val rule syntax

status-att-list = status-att-val *(SP status-att-val)
                  ;; Redefines status-att-list from RFC 3501.
                  ;; status-att-val is defined in RFC 3501 errata

status-att-val  = ("MESSAGES" SP number) /
                  ("RECENT" SP number) /
                  ("UIDNEXT" SP nz-number) /
                  ("UIDVALIDITY" SP nz-number) /
                  ("UNSEEN" SP number)
                  ;; Extensions to the STATUS responses
                  ;; should extend this production.
                  ;; Extensions should use the generic
                  ;; syntax defined by tagged-ext.

store           = "STORE" SP sequence-set store-modifiers
                  SP store-att-flags
                  ;; extend [IMAP4] STORE command syntax
                  ;; to allow for optional store-modifiers

store-modifiers = [ SP "(" store-modifier *(SP store-modifier)
                    ")" ]

store-modifier  = store-modifier-name [SP store-modif-params]

store-modif-params = <for extension only>
                  ;; This rule exists solely to be augmented by
                  ;; extensions via incremental alternative
                  ;; ("=/") rules.  It is strongly recommended
                  ;; that such extensions match a subset of the
                  ;; tagged-ext-val rule syntax

store-modifier-name = tagged-ext-label

tag-string         = string
                     ;; tag of the command that caused
                     ;; the ESEARCH response, sent as
                     ;; a string.

tagged-ext          = tagged-ext-label SP tagged-ext-val
                        ;; recommended overarching syntax for
                        ;; extensions

tagged-ext-label    = atom
                       ;; <<or should this be astring?>>
```

```
   tagged-ext-comp     = astring /
                         tagged-ext-comp *(SP tagged-ext-comp) /
                         "(" tagged-ext-comp ")"
                          ;; extensions that follow this general
                          ;; syntax should use nstring instead of
                          ;; astring when appropriate in the context
                          ;; of the extension

   tagged-ext-val      = astring / "(" [tagged-ext-comp] ")"
```

4. Security Considerations

It is believed that this document doesn't add any new security concerns that were not already identified in RFC 3501.

5. IANA Considerations

This document doesn't define any new IMAP extension, so no action from IANA is required.

6. References

6.1 Normative References

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

[IMAP4] Crispin, M., "Internet Message Access Protocol - Version 4rev1", RFC 3501, University of Washington, March 2003.

[ABNF] Crocker, D. (Ed.) and P. Overell , "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.<<Needs updating to draft-crocker-abnf-rfc2234bis-00>>

[CHARSET] Freed, N. and J. Postel, "IANA Character Set Registration Procedures", RFC 2978, October 2000.

[MULTIAPPEND] Crispin, M., "Internet Message Access Protocol (IMAP) - MULTIAPPEND Extension", RFC 3502, March 2003.

[NAMESPACE] Gahrns, M. and C. Newman, "IMAP4 Namespace", RFC 2342, May 1998.

[LITERAL+] Myers, J., "IMAP4 non-synchronizing literals", RFC 2088, January 1997.

7. Acknowledgments

This documents is based on ideas proposed by Pete Resnick, Mark Crispin, Ken Murchison, Philip Guenther, Randall Gellens and Lyndon Nerenberg.

However all errors and omissions must be attributed to the authors of the document.

literal8 syntax was taken from RFC 3516.

8. Author's Addresses

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

Email: Alexey.Melnikov@isode.com

Cyrus Daboo
ISAMET, Inc.
5001 Baum Blvd.
Suite 650
Pittsburgh, PA 15213
US

EMail: cyrus@daboo.name

9. Full Copyright Statement

10. Intellectual Property

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.

11. Appendix A. Editorial.

<<Note that this section will be deleted before publication>>

11.1 Change Log

```
00   Initial Revision.
01   Added Cyrus as co-author. Added BINARY literals. Added section
     about APPEND. Clarified that the order of all parameters/modifiers is
     arbitrary. Unrecognized SELECT/EXAMINE parameter should cause the BAD,
     not the NO response.
02   Updated boilerplate. Extended SEARCH modifiers to be consistent
     with STORE modifiers. Rewrote FETCH modifier syntax for consistency.
03   Updated as per comments from Philip (ABNF suggestions, in
     particular addition of response-data; normative language).
     Incorporated RFC 3501 ABNF errata from Mark. Added extensions to
     CREATE and RENAME commands. Updated ABNF to use consistent grammer for
     all extension elements (this changed ABNF for SELECT/EXAMINE and
     FETCH).
04   Added ESEARCH response. Added search-program from IMAP URL.
     Removed the partition parameter from CREATE/RENAME. Added NAMESPACE
     command/response.
05   Added non-synchronizing literals (RFC 2088). Clarified generic
     syntax for STATUS responses.
```