Abstract

Mobile IMAP clients are notoriously slow to connect to IMAP mail servers because of the large volume of data exchanged in order to ensure that the client is synchronized with the server. This problem is amplified when the mobile terminal is moving which exasperates the connection problems they tend to have: low bandwidth allocations, out-of-range disconnects, and poor connection quality to the cell/WLAN network.

This problem causes the IMAP client to become nearly unusable due to the inordinate amount of time spent waiting for the client to reconnect and synchronize with the server. In order to significantly boost the proliferation of the IMAP email client in the mobile community a new reconnect system must be introduced.

This document defines a quick reconnect IMAP4 extension, which gives a mobile client the ability to resume a previously abandoned session, without the need to perform a full synchronization sequence.
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].

"Resumable session" (or just "session") refers to the entire sequence of client/server interaction from the initial session creation during LOGIN/AUTHENTICATE command until it is explicitly deleted upon client request or until the session expires some time after disconnect. A resumable session can be terminated by the [IMAP4] LOGOUT command. Note, that a resumable session doesn’t terminate when the connection is lost or closed with LOGOUT (PRESERVE) command (see Section 4). Also note that this term has a different meaning from the term "session" used by [IMAP4].

The term "event" is as defined in [C2S].

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

2. Introduction and Overview

A significant hurdle to the acceptance of IMAP enabled mobile email clients by the general public is the long wait times required to
connect the client to the server and resync all data before the user can use the client. The fact that mobile terminals are, by definition, mobile, causes more problems because of the connection problems mobility implies at the physical and network layer. Due to these problems, users tend to give up on the email client on their mobile terminal and defer email access to wireline terminals or proprietary, dedicated mobile email terminals.

To compound the problem, some operators charge for the amount of data sent/received to the terminal, creating access rates which may be out of the realm of cost vs. convenience for the majority of mobile users.

Usability of the IMAP email client can be improved by orders of magnitude if the IMAP client can connect/reconnect in a much shorter time and with fewer data transfers.

The majority of the connect process is spent fetching enough information from the server to ensure that the client is completely synchronized with the server. However, if the client lost connection to the server, the user will most likely try to reconnect to the server within a few minutes. If we assume such a short time scale between reconnects, we can also assume that no major modifications/changes have been done to the server since the last successful login.

Using these assumptions, a model can be generated which will allow the user to reconnect to the server in a fraction of the time required for a normal connection, by assuming that the capabilities and session state of the last connection are still valid and that the server remembers the last state of the client connection.

<<Note to RFC editor: This document is compliant with "transitional IMAP capabilities" document [TRANS-CAPA]. Please change the capability name below to "RECONNECT">>

The quick reconnect IMAP extension is present if an IMAP4 server returns "X-DRAFT-W02-RECONNECT" as one of the supported capabilities to the CAPABILITY command. Note, that this extension REQUIREs support for the CONDSTORE [CONDSTORE] IMAP extension, so it MUST be announced in the CAPABILITY response as well.

The text below outlines how quick reconnect extension is to be used. The description assumes that the quick reconnect is mainly used when the client closes the TCP connection (intentionally or accidentally) and tries to reconnect within a preset amount of time.

1. The client authenticates and ask the server to creates a new session by specifying a NEWSID parameter to LOGIN/Authenticate command. The server generates a new Session ID (SID) and returns it to the client.
2. The client optionally opens a mailbox and does other things.
3. At some point the client loses connectivity. The server retains all information about the client’s session, including information about the currently selected mailbox, for some amount of time. (case a) The client reconnects before the session timeout. It authenticates as the same user as before and specifies the previously used session ID. The client and server begin at the same mailbox as before the reconnect. (case b) The client reconnects after the session timeout. It authenticates as the same user and specifies the previously used session ID. Since the session has timed out on the
server, the server has no record of this session and creates a new session. This behaviour deprecates to a normal IMAPv4 login.

2.1 Typical IMAP Session

A typical IMAP client session looks as follows:
(Modified dump from network sniffer)

S: * OK [CAPABILITY IMAP4REV1 LOGIN-REFERRALS STARTTLS AUTH=LOGIN]
C: 2 authenticate login
S: + VXNlciBOYW1lAA==
C: dGVzdDg=
S: + UGFzc3dvcmQA
C: dGVzdDg=
S: 2 OK [CAPABILITY IMAP4REV1 IDLE NAMESPACE MAILBOX-REFERRALS SCAN SORT THREAD=REFERENCES THREAD=ORDEREDSUBJECT MULTIAPPEND] User test8 authenticated
C: 3 namespace
S: * NAMESPACE ("" "/") ("#mhinbox" NIL) ("#mh/ " "/") ("-" "/")
("#shared/ " "/") ("#ftp/ " "/") ("#news." ".") ("#public/ " "/")
S: 3 OK completed
C: 4 lsub "" "/"
S: 4 OK LSUB completed
C: 5 lsub "" "#mhinbox"
S: 5 OK LSUB completed
C: 6 lsub "" "#mh/"
S: * NO /home/test8/.mh_profile not found, mh format names disabled
S: 6 OK LSUB completed
C: 7 lsub "" "/-
S: 7 OK LSUB completed
C: 8 lsub "" "#shared/"
S: 8 OK LSUB completed
C: 9 lsub "" "#ftp/"
S: 9 OK LSUB completed
C: 10 lsub "" "#news."
S: 10 OK LSUB completed
C: 11 lsub "" "#public/"
S: 11 OK LSUB completed
C: 12 list "" "INBOX"
S: * LIST (\NoInferiors) NIL INBOX
S: 12 OK LIST completed
C: 13 list "" "Trash"
S: 13 OK LIST completed
C: 14 create "Trash"
S: 14 NO CREATE failed: Can't create mailbox node /home/test8/:
Permission denied
C: 15 select "INBOX"
S: * 464 EXISTS
S: * 3 RECENT
S: * OK [UNSEEN 12] Message 12 is first unseen
S: * OK [UIDVALIDITY 3857529045] UIDs valid
S: * OK [UIDNEXT 550] Predicted next UID
S: * FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
S: * OK [PERMANENTFLAGS (\Deleted \Seen \*)] Limited
S: 15 OK [READ-WRITE] SELECT completed
C: 16 UID fetch 1:* (FLAGS)
S: * 1 FETCH (UID 41 FLAGS (\Seen))
S: * 2 FETCH (UID 43 FLAGS (\Seen \Flagged))
S: ...
S: * 464 FETCH (UID 541 FLAGS (Junk))
S: 16 fetch completed
C: 17 IDLE
S: + Waiting for DONE

(Note, that the client has recognized the CAPABILITY response code in the initial OK response, it has omitted the CAPABILITY command.)

The above dump was generated usig Ethereal and it reported that the above process generated just under 20 KB of data.

The above protocol sequence simply logged the user into the account, selected the INBOX mailbox, downloaded the updated flags for a subset of messages, and then went IDLE waiting for another command.
All this before the user can even start using the client program.

2.2 Reconnect IMAP Session

Now lets look at the same mailbox, but see how the protocol would react if the user had dropped connection and was reconnecting within a few minutes.

As in the previous section, the client has recognized the CAPABILITY response code in the initial OK response, so it has omitted the CAPABILITY command.

C: 2 authenticate login (SID P1234567890 56789 20010715194045000 41,43:211,214:541)
S: + VXNlciBOYW1lAA==
C: dGVzdDg=
S: + UGFzc3dvcmQA
C: dGVzdDg=
S: * FOLDER INBOX
S: * 464 EXISTS
S: * 3 RECENT
S: * OK [UIDVALIDITY 3857529045] UIDVALIDITY
S: * OK [UIDNEXT 550] Predicted next UID
S: * OK [HIGHESTMODSEQ 20010715194045007]
S: * 1 FETCH (UID 1 FLAGS (\Seen))
S: 2 OK [CAPABILITY IMAP4REV1 IDLE NAMESPACE MAILBOX-REFERRALS SCAN SORT THREAD=REFERENCES THREAD=ORDEREDSUBJECT MULTIAPPEND] User test8 authenticated
C: 3 IDLE
S: + Waiting for DONE

That’s it! And since no new messages arrived, no headers were downloaded. The total byte count is about 500 bytes, significantly less than the amount of data transferred during a normal connect request.

3. LOGIN and AUTHENTICATE commands

3.1 Extended LOGIN command
Arguments: user name
password
optional SID or NEWSID parameter

Data: OPTIONAL untagged responses: FOLDER (required if either SID or NEWSID parameter was specified, unless the server returns NOSID)
OPTIONAL untagged responses: NEWSID or NOSID
(required if NEWSID parameter was specified; also required if SID parameter was specified but the given session ID is invalid)
OPTIONAL - also untagged responses associated with SELECT

Result: OK - login completed, now in authenticated state (or selected) state; see section 3.3
NO - login failure: user name or password rejected
BAD - arguments invalid

This section extends the definition of the LOGIN command as defined in [IMAP4]. The LOGIN command identifies the client to the server and carries the plaintext password authenticating this user. In addition, the client can do one of two things: create a new session by specifying the NEWSID parameter or pass a session identifier ("SID") in the SID parameter in order to attempt to resume a previously created session. Either parameter is only used if login is successful.

If no SID or NEWSID is specified, the LOGIN command behaves as described in [IMAP4].

If either SID or NEWSID is specified, the server first performs all actions associated with a regular LOGIN command. Should authentication succeed, the server must try to resume the specified session as described in section 3.3 if the SID parameter was specified. If the client has specified NEWSID, the server is required to create a new session as described in Section 3.4.

The NEWSID parameter is mutually exclusive with the SID parameter. If the client has specified both the server MUST return BAD tagged response.

Example: First login, the client needs to perform a state-comparison-sync to get in sync.
C: A01 LOGIN joe password (NEWSID)
S: * NEWSID ABC412423rs
S: A01 OK LOGIN completed

Example: A successful Lemonade login resuming an old session
C: A02 LOGIN joe password (SID ABC412423rs)
S: * FOLDER ()
S: ...
S: A02 OK LOGIN completed

Example: A successful Lemonade login resuming an old session in SELECTED state with the INBOX selected.
C: A02 LOGIN joe password (SID ABC412423rs 67890007 20010715194045000 41,43:211,214:541) 
S: * FOLDER INBOX
S: * 14 EXISTS
S: * 49 FETCH (UID 117 FLAGS (\Seen \Answered))
S: ...
Example: A successful Lemonade login resuming an old session in SELECTED state with the INBOX selected, but where the server could not cache all the events since the last disconnect. The server sends the NEWSID response, which means that the client should perform full resynchronization.

C: A02 LOGIN joe password (SID ABC412423rs 67890007 20010715194045000 41,43:211,214:541)
S: * NEWSID jwejei9qe3r
S: * FOLDER INBOX
S: A02 OK LOGIN completed

Example: A successful Lemonade login resuming an old session in SELECTED state with the mailbox Drafts selected, but the mailbox was deleted and another one was renamed to become Drafts, while the client was offline. The server chooses to keep the Drafts mailbox open, however it sends the NEWSID response, which means that the client should perform full resynchronization.

C: A02 LOGIN joe password (SID Frt-egf-779 6896834 11 20:35)
S: * NEWSID Frt-egf-779
S: * FOLDER Drafts
S: * OK [UIDVALIDITY 1098183472] Current UIDValidity
S: * OK [UIDNEXT 12] Next available UID
S: * 7 EXISTS
S: * 0 RECENT
S: * FLAGS (\Answered \Flagged \Draft \Deleted \Seen)
S: * OK [PERMANENTFLAGS (\Answered \Flagged \Draft \Deleted \Seen \*)] Permanent flags
S: A02 OK [READ-WRITE] LOGIN completed

3.2 Extended AUTHENTICATE command

Arguments: authentication mechanism name
optional initial SASL response, as per [IMAP-SASL-IR]
optional SID or NEWSID parameter

Responses: continuation data can be requested

Data: OPTIONAL untagged responses: FOLDER (required if either SID or NEWSID parameter was specified, unless NOSID response is returned)
OPTIONAL untagged responses: NEWSID or NOSID (required if NEWSID parameter was specified; also required if SID parameter was specified but the given session ID is invalid)
OPTIONAL - also untagged responses associated with SELECT

Result: OK - authenticate completed, now in authenticated (or selected) state; see section 3.3
NO - authenticate failure: unsupported authentication
BAD - authenticate failure: arguments invalid, also authentication exchange cancelled

This section extends the definition of the AUTHENTICATE command as defined in [IMAP4]. The AUTHENTICATE command indicates a [SASL] authentication mechanism to the server. If the server supports the requested authentication mechanism, it performs an authentication protocol exchange to authenticate and identify the client. It MAY
also negotiate an OPTIONAL security layer for subsequent protocol interactions. If the requested authentication mechanism is not supported, the server SHOULD reject the AUTHENTICATE command by sending a tagged NO response. See [IMAP4] for more details.

The client can also pass a session identifier ("SID") in order to attempt to resume a previously created session. This is only used if authentication exchange is successful. If it is, then the presence of a SID tells the server to try to resume the specified session as described in section 3.3.

Alternatively, the client can pass a NEWSID parameter that requests that the server creates a new session. This is only used if authentication exchange is successful. See section 3.4 for detailed description of server’s behavior in this case.

The NEWSID parameter is mutually exclusive with the SID parameter. If the client has specified both the server MUST return the BAD tagged response.

<<Check how this will work together with SASL-INITIAL. >>

Example: A successful Lemonade authentication exchange resuming an old session in SELECTED state with the INBOX selected, but where the server could not cache all the events since the last disconnect. The server returns a new SID.

C: A02 AUTHENTICATE DIGEST-MD5 (SID B0rkjr5745745i)
S: + cmVhbG09ImVsd29vZC5pbn5vc29mdC5jb20ILG5vbmlNPSJPQ7ZjUz
RzIoRFHbTJoacIscW9wFSJ0dXRoIixhbGdvcm1oaGG9bWQ1LXNl
3MsY2hhcncNldDd11dGt0OA==
C: Y2hhcnNldDd11dGt0C1cxOCx1c2VvbmbFtZ0i12hyaxXMlILHJ1YXxtPSJlbH
dvb2uQaW5ub3NvZ29tIixub25jZT0i102ZT0i10ZTUz5dEVRR20yaGgi
LG5jP1TAwMDAwMDAxLG5ub25jZT0i10ZTUz5dVCVYUm5lGzUz
VzdC11cmk9InlmtYXAvZ2x3b29kLmNubm9zb2Z0LmNvbS1scmVzcG9u
cZU92DM40Gr2DkzD1YmQ3NjBhMTUyMzIzJj1jczSCW9wPW
F1dGg=
S: + cnNwYXV0aD11YTQwZjYwMz11YzQyN211NlT3YjQgOUJGJhYmNKZmZm
ZA==
C:
S: A02 OK User authenticated successfully
S: * NEWSID jwejei9ge3r
S: * FOLDER INBOX
<<Can we send untagged responses before the final OK? If we can’t, can we do this as shown?>>

3.3 Resuming existing session.

This section describes server behaviour if the client has specified the SID parameter to a LOGIN/AUTHENTICATE command.

The SID parameter is followed by four arguments: session identifier followed by three optional arguments: the mailbox UIDVALIDITY, the last known modification sequence and the list of known UIDs.

<<What the server should do if a mailbox is selected, but there are no optional parameters? Are they required if a mailbox is selected?>>

The server checks the session identifier (SID) first. If the session identifier by the SID is not recognized (for example, this is the first time the client tries to connect, or the the SID is
invalid because it is either not recognized, or recognized but the session has expired on the server), the server creates a new session as described in Section 3.4. Otherwise the server resumes a previously created session for the client.

If a session is resumed, the server MUST inform the client of the state of the server by sending an untagged FOLDER response. The FOLDER response contains the name of the currently selected mailbox name or "()" if there is no mailbox selected. If, before the disconnect, the session had a mailbox selected and the mailbox has been deleted since then (or the user has no longer permissions to access it), the server MUST return a special FOLDER untagged response that contains "()" as the mailbox name. This tells the client that the session has returned to the unauthenticated state.

If at this point there is no mailbox selected, the remaining arguments MUST be ignored by the server (however it MAY check them for syntactic validity).

Once the session has successfully resumed and there is a mailbox selected, the server checks the UIDVALIDITY value provided by the client. If the provided UIDVALIDITY doesn't match the UIDVALIDITY for the currently selected mailbox, the server MUST send the NEWSID response (that MAY contain the SID of the resumed session) in order to force the client to perform full resynchronization. The remaining arguments are ignored in this case.

If the provided UIDVALIDITY matches the actual, the server then checks the last known modification sequence <<if one was provided>>. The server sends the client any pending flag changes (using FETCH responses that MUST contain UIDs) that have occurred in this folder since the provided modification sequence.

If the list of known UIDs was also provided, the server should only report flag changes for the provided messages, plus it must report all messages from the list which were expunged. <<What if the list of UIDs was not provided?>> Thus, the client can just service these pending events and need not perform a full resynchronization. The result of those two steps is semantically equivalent to the client issuing:

```
tag1 UID FETCH <known-uids> (FLAGS) (CHANGEDSINCE <mod-sequence-value>)
```

(Note that messages not returned by this command were either expunged or their flags have not changed)

followed by:
```
tag2 UID SEARCH <known-uids>
```

Note, that the server will return the EXPUNGED response for the latter SEARCH (see also Section 5.4). If these events could not be cached for some reason, the server MUST return the NEWSID response to signal that the client should perform a state-comparison based synchronization. Note, that the NEWSID response MAY contain the same SID as specified in the SID parameter. Section 3.4 describes server behaviour in this case. <<There seems to be additional failure case: the server has sent the client a new EXISTS value, but the client has lost connection before receiving it>>

Example: A successful Lemonade login resuming an old session in SELECTED state with the INBOX selected.
C: A02 LOGIN joe password (SID ABC412423rs 67890007
2001071519405000 41,43:211,214:541)
S: * FOLDER INBOX
S: * 314 EXISTS
S: * 49 FETCH (UID 117 FLAGS (\Seen \Answered))
S: * 50 FETCH (UID 119 FLAGS (\Draft $MDNSent))
S: ...
S: * 100 FETCH (UID 541 FLAGS (\Seen $Forwarded))
S: * EXPUNGED 41,43:116,118,120:211,214:540
S: A02 OK LOGIN completed

Note that if, while the client was offline, the mailbox was deleted and another mailbox was renamed on top of the selected, the server may do one of the two things:

1. return the NEWSID response, as well as new UIDVALIDITY and all other responses required on select.
2. return "FOLDER ()" response to return the client to authenticated state.

<<Ideally, this should be a separate section, as it applies to both 3.3 and 3.4>> Whether a session was successfully resumed or a new one was created, the server is now required to remember the session state associated with the (username, SID) pair. This state includes the currently selected mailbox and any associated state, which MUST include the mailbox UIDVALIDITY value, the number of messages (EXISTS), the number of recent messages (RECENT), and the UID of the last message in the mailbox.

<<Read-only state? PERMANENTFLAGS/FLAGS? Other?>>. If the server supports CONDSTORE extension [CONDSTORE], it MUST also remember the HIGHESTMODSEQ value. Other IMAP extensions can define additional state that has to be tracked <<e.g. ACL MYRIGHTS?>>.

Note that TLS and/or SASL security layer state [SASL] created with STARTTLS/AUTHENTICATE command on a previous connection can’t be resumed using this mechanism. A client wishing to resume a TLS layer or a SASL security layer should use TLS session resumption or SASL fast reauthentication (if it is supported by the selected SASL mechanism).

<<Define how resuming a session interacts with ANNOTATE and other IMAP extensions>>

Each session’s state MUST be kept by the server for at least 30 minutes after a client disconnect caused by dropping TCP connection or LOGOUT (PRESCRIBE) command. After that the session is considered expired, the session ID is no longer valid and its associated state may be deleted.

Note that SIDs are chosen by the server and are opaque to the client. The same SID used with two different usernames refers to two different sessions, unless the two usernames refer to the same user account, e.g. one username is an alias for another.

3.4 Creating a new session

This section describes server behaviour if the client has specified the NEWSID parameter to a LOGIN/AUTHENTICATE command. The NEWSID parameter tells the server to create a new resumable
session and return a unique SID in the NEWSID response. This section also applies if the client has specified the SID parameter to a LOGIN/AUTHENTICATE command and the specified session can’t be resumed (see Section 3.3).

Each server MUST be able to keep track of at least 5 sessions for the same user. A server implementation might also have a per-server or per-username limit on maximum number of resumable sessions. This document doesn’t define this limit. If the server has reached the limit, it MUST return the NOSID response and proceed as if no NEWSID parameter was specified by the client. In particular it MUST NOT return NEWSID response.

4. PRESERVE parameter to LOGOUT command

This section updates the description of the LOGOUT command found in Section 6.1.3 of [IMAP4]. A LOGOUT command with no parameters tell the server to terminate a resumable session.

This document also adds a new parameter "PRESERVE" to the LOGOUT command. This tells the server that it should not terminate the current resumable session. If the client doesn’t want to terminate the current session on logout it SHOULD use "LOGOUT (PRESERVE)" instead of just dropping TCP connection. The former helps to avoid unnecessary resource consumption on the server caused by a TCP disconnect.

Example: C: A023 LOGOUT (PRESERVE)  
S: * BYE IMAP4rev1 Server logging out, state preserved  
S: A023 OK LOGOUT completed  
(Server and client then close the connection)

5. New responses

5.1 FOLDER Response

Contents: name of the selected mailbox or "()" if there is no selected mailbox

The FOLDER response tells the client that a session was successfully resumed and returns the name of the selected mailbox. The "()" sequence is used to signal that there is no mailbox selected.

5.2 NEWSID Response

Contents: generated session ID

The NEWSID response tells the client the unique identifier that can be used to resume the created session. It also tells the client to perform a full state resynchronization.

5.3 NOSID Response

Contents: none
The NOSID response tells the client that the server was unable to create a new session, probably because it has reached some implementation specific limit.

5.4 EXPUNGED Response

Contents:  list of UIDs

The EXPUNGED response reports that the specified UIDs have been permanently removed from the mailbox. This response is similar to the EXPUNGE response [IMAP4], however it may return information about multiple messages and it returns UIDs, instead of message numbers.

<<The EXPUNGED response also decrements the number of messages in the mailbox; it is not necessary to send an EXISTS response with the new value.>> <<It also affects message numbers?>>

An EXPUNGED response MUST NOT be sent when no command is in progress, nor while responding to a FETCH, STORE, or SEARCH command. This rule is necessary to prevent a loss of synchronization of message sequence numbers between client and server. A command is not "in progress" until the complete command has been received; in particular, a command is not "in progress" during the negotiation of command continuation.

Note: UID FETCH, UID STORE, and UID SEARCH are different commands from FETCH, STORE, and SEARCH. An EXPUNGED response MAY be sent during a UID command.

<<An alternative is to say that the EXPUNGED response is only allowed on LOGIN/AUTHENTICATE with SID parameter.>>

The update from the EXPUNGED response MUST be recorded by the client.

Example:  S: * EXPUNGED 5,7,9,15:21

6. 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] or [CONDSTORE].

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.

```
login            = "LOGIN" SP userid SP password [login-params]
authenticate     = "AUTHENTICATE" SP auth-type
                   [SP (base64 / ")"] [login-params]
                   *(CRLF base64)
```
login-params = */ *(SP "(" login-param-data ")")
              ;; modifies the original IMAP4 LOGIN
              ;; command to accept optional parameters

login-param-data = login-param *(SP login-param)

login-param = astring /
              "(" astring SP astring *(SP astring) "")"
              ;; parameters to LOGIN may contain one or
              ;; more atoms or strings - multiple items
              ;; are always parenthesized

login-sid = "SID" SP session-id [SP uidvalidity SP
mod-sequence-value [SP known-uids]]
            ;; conforms to login-param-data syntax

uidvalidity = nz-number
known-uids = sequence-set
            ;; sequence of UIDs, "*" is not allowed

command-any = */ logout

logout = "LOGOUT" [logout-params]

logout-params = */ *(SP "(" logout-param-data ")")
              ;; modifies the original IMAP4 LOGOUT
              ;; command to accept optional parameters

logout-param-data = logout-param *(SP logout-param)

logout-param = astring /
              "(" astring SP astring *(SP astring) "")"
              ;; parameters to LOGOUT may contain one or
              ;; more atoms or strings - multiple items
              ;; are always parenthesized

logout-preserve = "PRESERVE"
                ;; conforms to logout-param-data syntax

login-newsid = "NEWSID"
             ;; conforms to login-param-data syntax

mailbox-data = */ folder-resp / newsid-resp / nosid-resp
                ;; <<Is it Ok to use mailbox-data?>>

message-data = */ expunged-resp

expunged-resp = "EXPUNGED" SP known-uids

folder-resp = "FOLDER" SP (folder | no_folder)

no_folder = "()"

newsid-resp = "NEWSID" SP session-id

session-id = string
            ;; a unique session identifier,
            ;; implementation specific. If starts from "P"
            ;; it must be followed by an international
            ;; phone number.
nosid-resp       = "NOSID"

7. Security Considerations

In order to mitigate DoS attacks cause by a client that tries to generate a numerous resumable sessions, a server implementation should consider imposing a limit on maximum number of resumable sessions associated with a particular user and/or coming from a particular IP address/network.

<< Add a warning about not being able to resume STARTTLS/AUTHENTICATE state >>

8. IANA Considerations

<<TDB>>

9. References

9.1 Normative References


[ABNF], DRUMS working group, Dave Crocker Editor, "Augmented BNF for Syntax Specifications: ABNF", Work in Progress.

[IMAP-DISC] Melnikov, A. "Synchronization Operations For Disconnected Imap4 Clients", work in progress, draft-melnikov-imap-disc-XX.txt


<<Normative??>>

<<[ACL] - informative? >>
9.2 Informative References


10. Acknowledgments

This document borrows some text and ideas from draft-maes-lemonade-command-extensions-00 by S. H. Maes, R. Lima, C. Kuang, R. Cromwell, V. Ha and E. Chiu.

Thanks to Arnt Gulbrandsen, Cyrus Daboo, Randall Gellens and Rob Siemborski for comments and corrections.

11. Author’s Addresses

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

Email: Alexey.Melnikov@isode.com

Corby Wilson
Enterprise Mobility Systems
Nokia
503 Martindale Street
Suite 610
Pittsburgh, PA 15212
USA

Email: Corby.Wilson@nokia.com

12. Full Copyright Statement

Copyright (C) The Internet Society (2004). 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.

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.

13. Intellectual Property

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.

Acknowledgement

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


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

14.1 Change Log

00 Initial Revision.
01 Added example protocol traces.
   Added interaction with SASL-IR (SASL initial response extension).
   SIDs are now server generated. Added NEWSID parameter to LOGIN/AUTHENTICATE and NEWSID response.
   Removed SESSION response. The FOLDER response can now tell if there is a mailbox selected or not.
   Added LOGOUT (PRESERVE).
02 Added NOSID response
   Added UIDVALIDITY argument to the SID parameter
   Added EXPUNGED response
   Added the following example: an opened mailbox was deleted and another was renamed on top, while the client was offline
   Various editorial changes, mostly rewriting text for clarity

14.2 Open Issues for Discussion

1. Untagged responses are not allowed during AUTHENTICATE command
2. See also issues marked with <<>>.

14.3 To Do

1. Add section about rationale?