

Internet Draft  
[draft-neystadt-imap-status-counters-01.txt](#)  
Expires: December 2002

J. Neystadt  
Comverse Technology  
A. Melnikov  
MessagingDirect  
June 2002

## IMAP4 rev1 - STATUS-COUNTERS Extension

### Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of [Section 10 of RFC2026](#).

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>.

### 1. Abstract

The STATUS-COUNTERS extends IMAP4rev1's STATUS command. This extension adds the following functionality:

Enable MUA to get mailbox counters per groups of messages, grouped according to the value of Message-Context header field. For example, the MUA can find out that the Inbox contains 5 unseen fax messages.

## [2. Table of Contents](#)

<a href="#">1. Abstract.....</a>	<a href="#">1</a>
<a href="#">2. Table of Contents.....</a>	<a href="#">2</a>
<a href="#">3. Introduction and Overview.....</a>	<a href="#">2</a>
<a href="#">4. Conventions Used in this Document.....</a>	<a href="#">3</a>
<a href="#">5. Features.....</a>	<a href="#">3</a>
<a href="#">5.1 CAPABILITY.....</a>	<a href="#">3</a>
<a href="#">5.2 STATUS command.....</a>	<a href="#">3</a>
<a href="#">5.3 COUNTERS response.....</a>	<a href="#">4</a>
<a href="#">6. Implementation Guidelines.....</a>	<a href="#">5</a>
<a href="#">6.1 Server Issues.....</a>	<a href="#">5</a>
<a href="#">6.2 Client Issues.....</a>	<a href="#">6</a>
<a href="#">7. Formal Syntax.....</a>	<a href="#">7</a>
<a href="#">8. Security Considerations.....</a>	<a href="#">8</a>
<a href="#">9. References.....</a>	<a href="#">8</a>
<a href="#">10. Author's Addresses.....</a>	<a href="#">9</a>
<a href="#">11. Change Log.....</a>	<a href="#">9</a>
<a href="#">12. Open Issues for Discussion.....</a>	<a href="#">10</a>
<a href="#">13. To Do.....</a>	<a href="#">10</a>

## [3. Introduction and Overview](#)

[IMAP4] defines the STATUS command that allows a Mail User Agent (MUA) to query status of a mailbox on the server. [\[UM-ISSUES\]](#) outlines a number of additional requirements that are beyond abilities of the standard STATUS command. Some extracts from these requirements are listed below.

[UM-ISSUES] 2.2.1 Mailbox Summary (Per-context counters in mailbox STATUS command):

The common TUI (Telephony User Interface) prompt "you have two new voice messages, six unheard messages, and one new fax message" requires more information than is made available by IMAP STATUS command. The STATUS command provides only a count of new and total messages and doesn't allow to extract standardized attributes from message headers. Without an extension to the STATUS command, in order to get any aggregate information about message types, a client has to open the mailbox with SELECT/EXAMINE and then either perform FETCH or one or more SEARCHes. Both scenarios are

computationally expensive on the server and may generate unnecessary traffic.

The STATUS-COUNTERS IMAP extension proposed in this document extends the STATUS command to return number of messages having an arbitrary IMAP flag set as well as belonging to a particular message-context class. A single extended STATUS command will be

able to extract enough data to count messages in various categories.

A server supporting this extension MAY extract all the information required to fulfil a particular client request on every request, however for an adequate performance it is RECOMMENDED that the server implements one of the following:

- an agent performing a message injection into the message store (MDA or IMAP server in the injection is done via IMAP APPEND) should parse the message and create indexes for the necessary information;
- IMAP server should parse and cache the necessary information after the first request.

STATUS-COUNTERS relies on a special support of \$Important keyword defined in [[IMAP-KEYWORDS](#)] by the IMAP server. In particular, the server MUST automatically set \$Important flag on injection of an "important" message as described in [[IMAP-KEYWORDS](#)].

#### [4.](#) 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](#)].

Space character may be represented in examples as "<SPACE>".

#### [5.](#) Features

##### [5.1](#) CAPABILITY

The STATUS-COUNTERS extension is implemented by any IMAP4 server returning "STATUS-COUNTERS" as one of the supported capabilities to the CAPABILITY command. If the server does not advertise the STATUS-COUNTERS capability, the client MUST NOT use the STATUS-COUNTERS extension and continue to operate with multiple old STATUS commands and FETCHes or SEARCHes as specified in the base IMAP specification (see [section 7.2](#) for some examples).

## [5.2](#) STATUS command

The existing [\[IMAP4\]](#) STATUS command is extended.

A new meta-data item named COUNTERS is added to the existing STATUS data items.

COUNTERS (list of counter names (or "attributes"))

This will provide client with different numeric summaries of messages grouped according to the value of Message-Context header field. The empty list specifies that only the total number of messages in a group should be returned.

If COUNTERS status data item was specified in the STATUS command, the untagged STATUS response code must contain COUNTERS data item.

Response to COUNTERS request can be slow, since some server implementations may need to go over whole mailbox and process all message headers in order to calculate counters.

Example:

```
C:    A CAPABILITY
S:    * CAPABILITY ... IMAP4rev1 STATUS-COUNTERS ...
S:    A OK CAPABILITY
...
C:    B STATUS Inbox (UIDNEXT UIDVALIDITY
      COUNTERS (\Seen $Important "Unseen-Important" \Recent))
S:    * STATUS Inbox (UIDNEXT 850 UIDVALIDITY 1234
      COUNTERS (All (100 \Seen 30 $Important 20<SPACE>
                  "Unseen-Important" 10 \Recent 5)<SPACE>
```

```

        "Voice-Message" (10 \Seen 3 $Important 2<SPACE>
            "Unseen-Important" 1 \Recent 5)<SPACE>
        "Fax-Message" (10 \Seen 3 $Important 2<SPACE>
            "Unseen-Important" 1 \Recent 5)))
S:      B OK STATUS completed.

```

### 5.3 COUNTERS response

Contents:   parenthesized list messages groups (by Message-Context)  
               each one including parenthesized list of counters

COUNTERS response list is of the following form:

```
(Group1 (Total Attrib1 Value Attrib2 Value) Group2 (...) ...)
```

Message-Context class of a message is determined by inspecting the Message-Context header field of the message and extracting its value. Group name is either a Message-Context class as defined in [[Message-Context](#)] (e.g. "Voice-Message" or "Fax-Message") or a special tag ALL. ALL is used to indicate aggregated counters of all

messages, ignoring Message-Context attribute. Also note, that according to [[Message-Context](#)] all messages without Message-Context header MUST be treated as if they have a Message-Context header with the value of "none". Comparison on Message-Context values MUST be case insensitive as per [[Message-Context](#)].

If messages of a particular Message-Context class are not present in the mailbox, the information for this class is not included in COUNTERS response list (for example, the IMAP example in the previous sections suggests that there are no SMS messages in the mailbox).

Valid group attributes are COUNTERS STATUS data items. Here are the details of the counters:

Any IMAP Flag	Number of messages with given IMAP flag set per given Message-Context class (or All). For example, \$Important or \Deleted.
"Unseen-Important"	Number of messages per given Message-Context class (or All) that have the \$Important

keyword set and don't have the \Seen flag set.

This extension allows IMAP server to implement additional attributes and provide it per message-context (i.e. "X-My-Special-Counter")


See previous section for the example of COUNTERS response.

## [6. Implementation Guidelines](#)

### [6.1 Server Issues](#)

Since in order to calculate message counters, the MIME headers of the messages are inspected, this calculation can consume significant amount of processing time. It is recommended for server implementations to generate the counters during the injection of the message (in MDA) or persistently cache them after they are calculated for the first time.

It is expected that MUAs, in order to display to the user the important or unseen messages, will query all subscribed mailboxes for several counters during every login. Thus it is recommended that the counters are cached permanently in the mailbox and their retrieval should be fast.

Contraversaly to [[IMAP-KEYWORDS](#)], server supporting STATUS-COUNTERS MUST be able to auto-set \$Important flag on every injected important message.

In order to satisfy the last MUST requirement a mail server MAY do one of the following:

1. Set the \$Important flag in [every] delivery agent using implementation specific delivery API. If the delivery agent is an IMAP client, it should specify the \$Important keyword in APPEND command.
2. Set the \$Important flag using Sieve script, if the mail server

supports Sieve and the Sieve extension for setting IMAP flags. When message is added to mailbox using APPEND, IMAP server should parse it and turn on the flag.

3. Implement the logic for setting \$Important flag when message is accessed first time by a MUA and persist this flag.

## 6.2 Client Issues

If an IMAP server implementation does not support this command, its logic can be achieved by one of the following algorithms:

A. For a given mailbox do:

1. a SELECT mailbox
2. b FETCH 1:\* (BODY[HEADER (Message-Context)] FLAGS)
3. Calculate the counters from FETCH responses.

or

B. For a given mailbox do:

1. a SELECT mailbox
2. b SEARCH HEADER "Message-Context" "Voice-Message"  
Construct message set <b> from the result; if <b> is empty, skip till step 5.
3. c SEARCH <b> UNSEEN.  
Construct message set <c> from the result (<c> is the list of all unseen voice messages); if <c> is empty, skip till step 5.
4. d SEARCH <c> KEYWORD \$Important.  
Result is the list of all important unseen voice messages;
5. Repeat steps 2-4 for each message class the client is interested in

6. Calculate the counters from SEARCH responses.

Scenario A may generate a lot of traffic for large mailboxes. Scenario B is more computationally expensive for the server than scenario A, but generates less traffic.

Note that for scenario B, there is no way for MUA to learn about

available Message-Contexts, as it is expected it will have to have this knowledge as part of its user-interface.

It is expected that MUAs will need to request STATUS-COUNTERS for all mailboxes it is interested in in order to generate a list to display. MUA should not request STATUS-COUNTERS for all mailboxes on the server, since the number of accessible mailboxes can be extremely large. Instead MUAs should request STATUS-COUNTERS only for mailboxes that would fit on a single screen or for all mailboxes the user is subscribed to (the latter may be obtained with LSUB).

## 7. 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.

```
status          = "STATUS" SP mailbox SP "(" status-att-ext
                  *(SP status-att-ext) ")"
                  ;; redefine STATUS command syntax defined in
                  ;; [IMAP4]

status-att-ext   = status-att / status-req-cnt
                  ;; status-att is defined in [IMAP4]

status-req-cnt   = "COUNTERS" SP "(" [counter-type
                  *(SP counter-type)] ")"

mailbox-data     =/ "STATUS" SP mailbox SP "("
                  [status-rsp-info *(SP status-rsp-info)] ")" /

status-rsp-info  = status-rsp-std / status-rsp-cnt
```



```

status-rsp-std  = status-att SP number

counter-type    = flag-fetch | counter-special
                  ;; note: per flag-fetch \Recent is allowed here

counter-special = <"> ("Unseen-Important" | cntr-special-ext <">

cntr-special-ext= atom
                  ;; any special counter extension, private
                  ;; extensions MUST start with "X-" prefix

status-rsp-cnt  = "COUNTERS" SP "(" sts-rsp-cnt-grp
                  *(SP sts-rsp-cnt-grp) ")"

sts-rsp-cnt-grp = msg-ctx-cls-all SP "(" total-value
                  *(SP counter-value) ")"

counter-value   = counter-type SP number
                  ;; counter type with the associated value for
                  ;; the corresponding Message-Context class

total-value     = number

msg-ctx-cls-all = <"> message-context-class <"> | "ALL"
                  ;; message-context-class is defined in
                  ;; [Message-Context]

```

## [8.](#) Security Considerations

There is a potential inconsistency that may be introduced by server between actual messages in the mailboxes and information reported by STATUS-COUNTERS. This can happen if the server reports inaccurate information. User Agents must not blindly rely on the correspondence between STATUS-COUNTERS and messages themselves.

It can be resource consuming for some server implementations to handle STATUS-COUNTERS command with a mailbox with many messages. This can potentially facilitate a "Deny of Service" attack. It is recommended for servers to implement artificial delay between two subsequent STATUS-COUNTERS commands with a very short delay between them.

## [9.](#) References

[[IMAP4](#)], Crispin, M., "Internet Message Access Protocol - Version 4rev1", [RFC 2060](#), December 1996.

## IMAP4 rev1 - STATUS-COUNTERS Extension

June 2002

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

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

[UM-ISSUES] Vaudreuil, G., "Messaging profile for telephone-based Messaging clients", [draft-vaudreuil-um-issues-00](#), February 21, 2002.

[Message-Context] Burger, E., et al, "Message Context for Internet Mail", [draft-ietf-vpim-hint-07](#), June 5, 2001.

[VPIM] Vaudreuil, G., "Voice Profile for Internet", [RFC 1911](#), February 1996.

[IMAP-KEYWORDS] Melnikov, A., et al, "Registration of common IMAP keywords", Work in progress: [draft-melnikov-imap-keywords-XX.txt](#), June 2002

## [10.](#) Author's Addresses

John Neystadt  
Comverse Technology  
Habarzel 29, Ramat Hahayal Tel-Aviv, Israel, 69710  
Phone: +972-3-645-4185  
Email: [john@comverse.com](mailto:john@comverse.com)

Alexey Melnikov  
ACI Worldwide/MessagingDirect  
Address: 22 The Quadrant, Richmond,  
Surrey, United Kingdom, TW9 1BP  
Phone: +44 20 8332 4508  
Email: [mel@messagingdirect.com](mailto:mel@messagingdirect.com)

## [11.](#) Change Log

- 00 Initial Revision.
- 01 - Added support for generic IMAP flag counting.
  - Removed Importance, since it now relies on \$Important flag from [[IMAP-KEYWORDS](#)].

- Implemented various comments received from the imap list regarding implementation options.
- Added ABNF.
- Major wording improvements.

## [12.](#) Open Issues for Discussion

- Currently there is no way to request counters of per only specific message context. If MUA requests new "COUNTERS" data item it will get them all. This can be wasteful on large amount of folders. On other hand this would complicate the command furthermore. If somebody thinks this must be added, please say so.
- Should EXAMINE and SELECT include COUNTERS untagged response as well? (Probably not)
- Is it useful to request just counters without grouping my message context class?

## [13.](#) To Do

Add a new command (similar to STATUS COUNTERS) that can be used in SELECTED state.

Neystadt

Expires - December 2002

[Page 10]