

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 19, 2019

A. Melnikov
Isode
June 17, 2019

IMAP QUOTA Extension
draft-ietf-extra-quota-00

Abstract

The QUOTA extension of the Internet Message Access Protocol ([RFC 3501](#)) permits administrative limits on resource usage (quotas) to be manipulated through the IMAP protocol.

This memo obsoletes [RFC 2087](#), but attempts to remain backwards compatible whenever possible.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

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."

This Internet-Draft will expire on December 19, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in [Section 4.e](#) of

Internet-Draft

IMAP QUOTA

June 2019

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1.	Document Conventions	3
2.	Introduction and Overview	3
3.	Terms	4
3.1.	Resource	4
3.1.1.	Name	4
3.1.2.	Definition	4
3.2.	Quota Root	5
4.	Definitions	5
4.1.	Commands	5
4.1.1.	GETQUOTA	5
4.1.2.	GETQUOTAROOT	6
4.1.3.	SETQUOTA	7
4.1.4.	New STATUS attributes	8
4.2.	Responses	8
4.2.1.	QUOTA	8
4.2.2.	QUOTAROOT	9
4.3.	Response Codes	9
4.3.1.	OVERQUOTA	9
5.	Resource Type Definitions	10
5.1.	STORAGE	10
5.2.	MESSAGE	11
5.3.	MAILBOX	11
6.	Formal syntax	11
7.	Security Considerations	13
8.	IANA Considerations	13
9.	Contributors	14

10.	Acknowledgments	14
11.	Changes since RFC 2087	14
12.	References	14
12.1.	Normative References	14
12.2.	Informative References	15

Author's Address	15
----------------------------	--------------------

[1.](#) Document Conventions

In protocol examples, this document uses a prefix of "C: " to denote lines sent by the client to the server, and "S: " for lines sent by the server to the client. Lines prefixed with "// " are comments explaining the previous protocol line. These prefixes and comments are not part of the protocol. Lines without any of these prefixes are continuations of the previous line, and no line break is present in the protocol unless specifically mentioned.

Again, for examples, the hierarchy separator on the server is presumed to be "/" throughout. None of these assumptions is required nor recommended by 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](#) [[RFC2119](#)].

Other capitalised words are IMAP4 [[RFC3501](#)] keywords or keywords from this document.

[2.](#) Introduction and Overview

The capability "QUOTA", denotes a [RFC2087](#) [[RFC2087](#)] compliant server. Some commands and responses defined in this document are not present in such servers, and clients MUST NOT rely on their presence in the absence of any capability beginning with "QUOTA=".

Any server compliant with this document MUST also return at least one capability starting with "QUOTA=RES-" prefix, as described in [Section 3.1](#).

This document also reserves all other capabilities starting with "QUOTA=" prefix to future standard track or experimental extensions

to this document.

Quotas can be used to restrict clients for administrative reasons, but the QUOTA extension can also be used to indicate system limits and current usage levels to clients.

Although [RFC2087](#) [[RFC2087](#)] specified an IMAP4 QUOTA extension, and this has seen deployment in servers, it has seen little deployment in clients. Since the meaning of the resources was left implementation-dependant, it was impossible for a client implementation to determine which resources were supported, and impossible to determine which

mailboxes were in a given quota root, without a priori knowledge of the implementation.

[3.](#) Terms

[3.1.](#) Resource

A resource has a name, a formal definition.

[3.1.1.](#) Name

[[CREF1: Fix IANA considerations section.]]

The resource name is an atom, as defined in IMAP4 [[RFC3501](#)]. These MUST be registered with IANA. Implementation specific resources begin with "V-" .

Supported resource names MUST be advertised as a capability, by prepending the resource name with "QUOTA=RES-". A server compliant with this specification is not required to support all reported resource types on all quota roots.

[3.1.2.](#) Definition

The resource definition or document containing it, while not visible through the protocol, SHOULD be registered with IANA.

The usage of a resource MUST be represented as a 32 bit unsigned integer. 0 indicates that the resource is exhausted. Usage integers

don't necessarily represent proportional use, so clients MUST NOT compare available resource between two separate quota roots on the same or different servers.

Limits will be specified as, and MUST be represented as, an integer. 0 indicates that any usage is prohibited.

Limits may be hard or soft - that is, an implementation MAY choose, or be configured, to disallow any command if the limit on a resource is or would be exceeded.

All resources which the server handles must be advertised in a CAPABILITY consisting of the resource name prefixed by "QUOTA=RES-". For compatibility with [RFC2087](#) [[RFC2087](#)], a client which discovers resources available on the server which are not advertised through this mechanism MUST treat them as if they were completely opaque, and without any meaning.

The resources STORAGE ([Section 5.1](#)), MESSAGE ([Section 5.2](#)) and MAILBOX ([Section 5.3](#)) are defined in this document.

[3.2](#). Quota Root

Each mailbox has zero or more implementation-defined named "quota roots". Each quota root has zero or more resource limits (quotas). All mailboxes that share the same named quota root share the resource limits of the quota root.

Quota root names need not be mailbox names, nor is there any relationship defined by this memo between a Quota root name and a mailbox name. A quota root name is an astring, as defined in IMAP4 [[RFC3501](#)]. It SHOULD be treated as an opaque string by any clients.

Quota roots are used since not all implementations may be able to calculate usage, or apply quotas, on arbitrary mailboxes or mailbox hierarchies.

Not all resources may be limitable or calculatable for all quota roots. Further, not all resources may support all limits - some limits may be present in the underlying system. A server

implementation of this memo SHOULD advise the client of such inherent limits, by generating QUOTA ([Section 4.2.1](#)) responses and SHOULD advise the client of which resources are limitable for a particular quota root. A SETQUOTA ([Section 4.1.3](#)) command MAY also round a quota limit in an implementation dependant way, if the granularity of the underlying system demands it. A client MUST be prepared for a SETQUOTA ([Section 4.1.3](#)) command to fail if a limit cannot be set.

Implementation Notes:

This means that, for example under UNIX, a quota root may have a MESSAGE ([Section 5.2](#)) quota always set due to the number of inodes available on the filesystem, and similarly STORAGE ([Section 5.1](#)) may be rounded to the nearest block and limited by free filesystem space.

[4.](#) Definitions

[4.1.](#) Commands

The following commands exist for manipulation and querying quotas.

[4.1.1.](#) GETQUOTA

Arguments: quota root

Responses: REQUIRED untagged responses: QUOTA

Result: OK - getquota completed
NO - getquota error: no such quota root, permission denied
BAD - command unknown or arguments invalid

The GETQUOTA command takes the name of a quota root and returns the quota root's resource usage and limits in an untagged QUOTA response. The client can try using any of the resource types returned in CAPABILITY response (i.e. all capability items with "QUOTA=RES-" prefix), however the server is not required to support any specific resource type for any particular quota root.

Example:

```
S: * CAPABILITY [...] QUOTA QUOTA=RES-STORAGE [...]
[...]
```

```
C: G0001 GETQUOTA "!partition/sda4"
S: * QUOTA "!partition/sda4" (STORAGE 104 10923847)
S: G0001 OK Getquota complete
```

[4.1.2.](#) GETQUOTAROOT

Arguments: mailbox name

Responses: REQUIRED untagged responses: QUOTAROOT, QUOTA

Result: OK - getquotaroot completed

NO - getquotaroot error: no such mailbox, permission denied

BAD - command unknown or arguments invalid

The GETQUOTAROOT command takes the name of a mailbox and returns the list of quota roots for the mailbox in an untagged QUOTAROOT response. For each listed quota root, it also returns the quota root's resource usage and limits in an untagged QUOTA response.

[[CREF2: Need to clarify that the mailbox name doesn't have to reference an existing mailbox. This can be handy in order to determine which quotaroot would apply to a mailbox when it gets created.]]

Example:

```
S: * CAPABILITY [...] QUOTA QUOTA=RES-STORAGE QUOTA=RES-MESSAGE
[...]
[...]
C: G0002 GETQUOTAROOT INBOX
S: * QUOTAROOT INBOX "#user/alice" "!partition/sda4"
S: * QUOTA "#user/alice" (MESSAGE 42 1000)
S: * QUOTA "!partition/sda4" (STORAGE 104 10923847)
```

```
S: G0002 OK Getquotaroot complete
```

[4.1.3.](#) SETQUOTA

Arguments: quota root

list of resource limits

Responses: untagged responses: QUOTA

Result: OK - setquota completed

NO - setquota error: can't set that data

BAD - command unknown or arguments invalid

The SETQUOTA command takes the name of a mailbox quota root and a list of resource limits. The resource limits for the named quota root are changed to be the specified limits. Any previous resource limits for the named quota root are discarded.

If the named quota root did not previously exist, an implementation may optionally create it and change the quota roots for any number of existing mailboxes in an implementation-defined manner.

[[CREF3: Should the server be sending untagged QUOTA responses for all side effect changes?]]

Example:

```
S: * CAPABILITY [...] QUOTA QUOTA=RES-STORAGE QUOTA=RES-MESSAGE
[...]
[...]
```

```
C: S0000 GETQUOTA "#user/alice"
```

```
S: * QUOTA "#user/alice" (STORAGE 54 111 MESSAGE 42 1000)
```

```
S: S0000 OK Getquota completed
```

```
C: S0001 SETQUOTA "#user/alice" (STORAGE 510)
```

```
S: * QUOTA "#user/alice" (STORAGE 58 512)
```

// The server has rounded the STORAGE quota limit requested to the nearest 512 blocks of 1024 octets, or else another client has performed a near simultaneous SETQUOTA, using a limit of 512.

```
S: S0001 OK Rounded quota
```

```
C: S0002 SETQUOTA "!partition/sda4" (STORAGE 99999999)
```

```
S: * QUOTA "!partition/sda4" (STORAGE 104 10923847)
```

// The server has not changed the quota, since this is a filesystem limit, and cannot be changed. The QUOTA response here is entirely optional.

```
S: S0002 NO Cannot change system limit
```


[4.1.4.](#) New STATUS attributes

DELETED-MESSAGES and DELETED-STORAGE status data items allow to estimate the amount of resource freed by an EXPUNGE on a mailbox.

DELETED-MESSAGES status data item requests the server to return the number of messages with \Deleted flag set.

DELETED-STORAGE status data item requests the server to return the amount of storage space that can be reclaimed by performing EXPUNGE on the mailbox. The server SHOULD return the exact value, however it is recognized that the server may have to do non-trivial amount of work to calculate it. If the calculation of the exact value would take a long time, the server MAY instead return the sum of [RFC822](#).SIZES of messages with the \Deleted flag set.

Example:

```
S: * CAPABILITY [...] QUOTA QUOTA=RES-STORAGE QUOTA=RES-MESSAGE  
[...]  
[...]  
C: S0003 STATUS INBOX (MESSAGES DELETED-MESSAGES DELETED-STORAGE)  
S: * STATUS INBOX (MESSAGES 12 DELETED-MESSAGES 4 DELETED-STORAGE  
8)
```

// 12 messages, 4 of which would be deleted when an EXPUNGE happens.

S: S0003 OK Status complete.

[4.2.](#) Responses

The following responses may be sent by the server.

[4.2.1.](#) QUOTA

Data: quota root name
list of resource names, usages, and limits

This response occurs as a result of a GETQUOTA or GETQUOTAROOT command. The first string is the name of the quota root for which this quota applies.

The name is followed by a S-expression format list of the resource usage and limits of the quota root. The list contains zero or more

triplets. Each triplet contains a resource name, the current usage of the resource, and the resource limit.

Resources not named in the list are not limited in the quota root. Thus, an empty list means there are no administrative resource limits in the quota root.

Example: S: * QUOTA "" (STORAGE 10 512)

[4.2.2.](#) QUOTAROOT

Data: mailbox name
zero or more quota root names

This response occurs as a result of a GETQUOTAROOT command. The first string is the mailbox and the remaining strings are the names of the quota roots for the mailbox.

Example:

S: * QUOTAROOT INBOX ""

S: * QUOTAROOT comp.mail.mime

[4.3.](#) Response Codes

[4.3.1.](#) OVERQUOTA

OVERQUOTA response code SHOULD be returned in the tagged NO response to an APPEND/COPY when the addition of the message(s) puts mailbox over any one of its quota limits.

Example:

```
S: C: A003 APPEND Drafts (\Seen $MDNSent) {310}
S: + Ready for literal data
C: Date: Mon, 7 Feb 1994 21:52:25 -0800 (PST)
C: From: Fred Foobar <foobar@Blurdybloop.COM>
C: Subject: afternoon meeting
C: To: mooch@owatagu.siam.edu
C: Message-Id: <B27397-0100000@Blurdybloop.COM>
C: MIME-Version: 1.0
C: Content-Type: TEXT/PLAIN; CHARSET=US-ASCII
C:
C: Hello Joe, do you think we can meet at 3:30 tomorrow?
C:
```

S: A003 NO [OVERQUOTA] APPEND Failed

Melnikov

Expires December 19, 2019

[Page 9]

Internet-Draft

IMAP QUOTA

June 2019

The OVERQUOTA response code MAY also be returned in an untagged NO response when the currently selected mailbox exceeds soft quota. [[CREF4: What about per-user quotas when no mailbox is selected or when the destination mailbox (different from the currently selected one) is affected?]] The response code MUST be followed by the tag of the command that caused this (such as APPEND or COPY). The tag MUST be omitted if an external event (e.g. LMTP delivery or APPEND/COPY in another IMAP connection) caused this event.

Example:

```
S: C: A003 APPEND Drafts (\Seen $MDNSent) {310}
S: + Ready for literal data
C: Date: Mon, 7 Feb 1994 21:52:25 -0800 (PST)
C: From: Fred Foobar <foobar@Blurdybloop.COM>
C: Subject: afternoon meeting
C: To: mooch@owatagu.siam.edu
C: Message-Id: <B27397-01000000@Blurdybloop.COM>
C: MIME-Version: 1.0
C: Content-Type: TEXT/PLAIN; CHARSET=US-ASCII
C:
C: Hello Joe, do you think we can meet at 3:30 tomorrow?
C:
S: * NO [OVERQUOTA A003] Soft quota has been exceeded
S: A003 OK [APPENDUID 38505 3955] APPEND completed
```

[5.](#) Resource Type Definitions

The following resource types are defined in this memo. A server supporting a resource type MUST advertise this as a CAPABILITY with a name consisting of the resource name prefixed by "QUOTA=RES-". A server MAY support multiple resource types, and MUST advertise all resource types it supports.

[5.1.](#) STORAGE

The physical space estimate, in units of 1024 octets, of the mailboxes governed by the quota root. This MAY not be the same as the sum of the [RFC822](#).SIZE of the messages. Some implementations MAY

include metadata sizes for the messages and mailboxes, other implementations MAY store messages in such a way that the physical space used is smaller, for example due to use of compression. Additional messages might not increase the usage. Client MUST NOT use the usage figure for anything other than informational purposes, for example, they MUST NOT refuse to APPEND a message if the limit less the usage is smaller than the [RFC822](#).SIZE divided by 1024 of the message, but it MAY warn about such condition.

The usage figure may change as a result of performing actions not associated with adding new messages to the mailbox, such as SEARCH, since this may increase the amount of metadata included in the calculations.

Support for this resource MUST be indicated by the server by advertising the CAPABILITY "QUOTA=RES-STORAGE".

A resource named the same was also given as an example in [RFC2087](#) [[RFC2087](#)], clients conformant to this specification connecting to servers which do not advertise "QUOTA=RES-STORAGE", yet allow a resource named STORAGE, MUST NOT assume that it is the same resource. [[CREF5: ?]]

[5.2.](#) MESSAGE

The number of messages stored within the mailboxes governed by the quota root. This MUST be an exact number, however, clients MUST NOT assume that a change in the usage indicates a change in the number of messages available, since the quota root may include mailboxes the client has no access to.

Support for this resource MUST be indicated by the server by advertising the CAPABILITY "QUOTA=RES-MESSAGE".

A resource named the same was also given as an example in [RFC2087](#) [[RFC2087](#)], clients conformant to this specification connecting to servers which do not advertise "QUOTA=RES-MESSAGE", yet allow a resource named MESSAGE, MUST NOT assume that it is the same resource.

[5.3.](#) MAILBOX

The number of mailboxes governed by the quota root. This MUST be an exact number, however, clients MUST NOT assume that a change in the usage indicates a change in the number of mailboxes, since the quota root may include mailboxes the client has no access to.

Support for this resource MUST be indicated by the server by advertising the CAPABILITY "QUOTA=RES-MAILBOX".

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 [\[RFC3501\]](#).

Melnikov

Expires December 19, 2019

[Page 11]

Internet-Draft

IMAP QUOTA

June 2019

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.

```
getquota           = "GETQUOTA" SP quota-root-name
getquotaroot       = "GETQUOTAROOT" SP mailbox
quota-list         = "(" quota-resource *(SP quota-resource) ")"
quota-resource     = resource-name SP resource-usage SP resource-
                    limit
quota-response     = "QUOTA" SP quota-root-name SP quota-list
quotaroot-response = "QUOTAROOT" SP mailbox *(SP quota-root-name)
setquota           = "SETQUOTA" SP quota-root-name SP setquota-list
setquota-list      = "(" [setquota-resource *(SP setquota-resource)]
                    ")"
setquota-resource  = resource-name SP resource-limit
```

quota-root-name = astring
 resource-limit = number
 resource-name = "STORAGE" / "MESSAGE" / "MAILBOX" /
 resource-name-vnd /
 resource-name-ext
 resource-name-vnd = "V-" atom
 ;; Vendor specific, must be registered with IANA.
 ;; The "V-" prefix should be followed by a domain
 name
 ;; under vendor's control.
 resource-name-ext = atom
 ;; Not starting with V- and defined
 ;; in a Standard Track or Experimental RFC
 resource-names = "(" [resource-name *(SP resource-name)] ")"
 resource-usage = number
 ;; must be less than corresponding resource-limit

capability-quota = capa-quota-res
 capa-quota-res = "QUOTA=RES-" resource-name
 status-att =/ "DELETED-MESSAGES" / "DELETED-STORAGE"
 [[CREF6: Should this be optional unless the
 server implements MESSAGE/STORAGE?]]
 resp-text-code =/ "OVERQUOTA" [SP tag]

7. Security Considerations

Implementors should be careful to make sure the implementation of these commands does not violate the site's security policy. The resource usage of other users is likely to be considered confidential information and should not be divulged to unauthorized persons.

8. IANA Considerations

IMAP4 capabilities are registered by publishing a standards track or IESG approved experimental RFC. The registry is currently located at:

<http://www.iana.org/assignments/imap4-capabilities>

IANA is requested to update definition of the QUOTA extension to point to this document.

IANA is also requested to create a new registry for IMAP quota resource types. Registration policy for this registry is "Specification Required". When registering a new quota resource type, the registrant need to provide the following: Name of the quota resource type, Author/Change Controller name and email address, short description and a reference to the specification.

This document includes initial registrations for the following IMAP quota resource type: STORAGE ([Section 5.1](#)), MESSAGE ([Section 5.2](#)) and MAILBOX ([Section 5.3](#)).

IANA is requested to reserve the prefix "QUOTA=RES-" in the IMAP4 capabilities registry and add a pointer to this document and to the IMAP quote resource type registry established above.

IANA is requested to reserve all other capabilities starting with "QUOTA=" prefix to future standard track or experimental extensions to this document.

9. Contributors

Dave Cridland wrote lots of text in an earlier draft that became the base for this document.

10. Acknowledgments

Editors of this document would like to thank the following people who provided useful comments or participated in discussions that lead to this update to [RFC 2087](#):

John Myers,
Cyrus Daboo,
Lyndon Nerenberg

This document is a revision of [RFC 2087](#). It borrows a lot of text from [RFC 2087](#). Thus work of the [RFC 2087](#) author John Myers is appreciated.

[11.](#) Changes since [RFC 2087](#)

This document is a revision of [RFC 2087](#). It tries to clarify meaning of different terms used by [RFC 2087](#). It also provides more examples, gives guidance on allowed server behaviour, defines IANA registry for quota resource types and provides initial registrations for 3 of them.

When compared with [RFC 2087](#), this document defines one more commonly used resource type, adds optional OVERQUOTA response code and defines two extra STATUS data items ("DELETED-MESSAGES" and "DELETED-STORAGE")

[12.](#) References

[12.1.](#) Normative References

- [ABNF] Crocker, D., Ed. and P. Overell, Ed., "Augmented BNF for Syntax Specifications: ABNF", [RFC 4234](#), October 2005.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.

- [RFC3501] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", [RFC 3501](#), DOI 10.17487/RFC3501, March 2003, <<https://www.rfc-editor.org/info/rfc3501>>.

12.2. Informative References

[RFC2087] Myers, J., "IMAP4 QUOTA extension", [RFC 2087](#), DOI 10.17487/RFC2087, January 1997, <<https://www.rfc-editor.org/info/rfc2087>>.

Author's Address

Alexey Melnikov
Isode Limited

Email: alexey.melnikov@isode.com
URI: <https://www.isode.com>