                Message Disposition Notification (MDN) profile for
                     Internet Message Access Protocol (IMAP)

Status of this Memo

   This document specifies an Internet standards track protocol for the
   Internet community, and requests discussion and suggestions for
   improvements. Please refer to the current edition of the "Internet
   Official Protocol Standards" (STD 1) for the standardization state
   and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Abstract

   The Message Disposition Notification (MDN) facility defined in RFC
   2298 provides a means by which a message can request that message
   processing by the recipient be acknowledged as well as a format to be
   used for such acknowledgements. However, it doesn't describe how
   multiple Mail User Agents (MUAs) should handle the generation of MDNs
   in an Internet Message Access Protocol (IMAP4) environment.

   This document describes how to handle MDNs in such an environment and
   provides guidelines for implementers of IMAP4 that want to add MDN
   support to their products.

Table of Contents

1. **Conventions Used in this Document**

    "C:" and "S:" in examples show lines sent by the client and server
    respectively.

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

2. **Introduction and Overview**

    This memo defines an additional [IMAP4] mailbox keyword that allows
    multiple Mail User Agents (MUAs) to know if a requested receipt
    notification was sent.

    Message Disposition Notification [MDN] does not require any special
    support of IMAP in the case where a user has access to the mailstore
    from only one computer and is using a single MUA. In this case, the
    MUA behaves as described in [MDN], i.e., the MUA performs automatic
    processing and generates corresponding MDNs, it performs requested
    action and, with the user's permission, sends appropriate MDNs. The
    MUA will not send MDN twice because the MUA keeps track of sent
    notifications in a local configuration. However, that does not work
    when IMAP is used to access the same mailstore from different
    locations or is using different MUAs.

   This document defines a new special purpose mailbox keyword $MDNSent
   that must be used by MUAs. It does not define any new command or
   response for IMAP, but describes a technique that MUAs should use to
   achieve interoperability.

   When a client opens a mailbox for the first time, it verifies that
   the server is capable of storing the $MDNSent keyword by examining
   the PERMANENTFLAGS response code. In order to support MDN in IMAP, a
   server MUST support either the $MDNSent keyword, or arbitrary message
   keywords.

3. Client behavior

   The use of IMAP requires few additional steps in mail processing on
   the client side. The following timeline modifies the timeline found
   in Section 4 of [MDN].

   -- User composes message.

   -- User tells MUA to send message.

   -- MUA passes message to MSA (original recipient information passed
      along). MUA [optionally] saves message to a folder for sent mail
      with $MDNSent flag set.

   -- MSA sends message to MTA.

   -- Final MTA receives message.

   -- Final MTA delivers message to MUA (possibly generating DSN).

   -- MUA logs into IMAP server, opens mailbox, verifies if mailbox can
      store $MDNSent keyword by examining PERMANENTFLAGS response.

   -- MUA performs automatic processing and generates corresponding MDNs
      ("dispatched", "processed", "deleted", "denied" or "failed"
      disposition type with "automatic-action" and "MDN-sent-
      automatically" disposition modes) for messages that do not have
      $MDNSent keyword, or \Draft flag set. (*)

   -- MUA sets the $MDNSent keyword for every message that required an
      automatic MDN to be sent, whether or not the MDN was sent.

   -- MUA displays a list of messages to user.

   -- User selects a message and requests that some action be performed
      on it.

-- MUA performs requested action and, with user's permission, sends
   appropriate MDN ("displayed", "dispatched", "processed",
   "deleted", "denied" or "failed" disposition type with "manual-
   action" and "MDN-sent-manually" or "MDN-sent-automatically"
   disposition mode). If the generated MDN is saved to a mailbox
   with the APPEND command, the client MUST specify the $MDNSent
   keyword in the APPEND.

-- MUA sets the $MDNSent keyword for all messages for which the user
   confirmed the dispatching of disposition (or was explicitly
   prohibited to do so).

-- User possibly performs other actions on message, but no further
   MDNs are generated.

(*) Note: MUA MUST NOT use \Recent flag as an indicator that it
    should send MDN, because according to [IMAP4], "If multiple
    connections have the same mailbox selected simultaneously, it is
    undefined which of these connections will see newly-arrived
    messages with \Recent set and which will see it without \Recent
    set". Thus, using \Recent as an indicator will cause
    unpredictable client behavior with different IMAP4 servers.
    However, the client MAY use \Seen flag as one of the indicators
    that MDN must not be sent. The client MUST NOT use any other
    standard flags, like \Draft or \Answered, to indicate that MDN
    was previously sent, because they have different well known
    meaning. In any case, in the presence of the $MDNSent keyword,
    the client MUST ignore all other flags or keywords for the
    purpose of generating an MDN and MUST NOT send the MDN.

When the client opens a mailbox for the first time, it must verify
that the server supports the $MDNSent keyword, or arbitrary message
keywords by examining PERMANENTFLAGS response code.

The client MUST NOT try to set the $MDNSent keyword if the server is
incapable of storing it permanently.

The client MUST be prepared to receive NO from the server as the
result of STORE $MDNSent when the server advertises the support of
storing arbitrary keywords, because the server may limit the number
of message keywords it can store in a particular mailbox. A client
SHOULD NOT send MDN if it fails to store the $MDNSent keyword.

Once the $MDNSent keyword is set, it MUST NOT be unset by a client.
The client MAY set the $MDNSent keyword when a user denies sending
the notification. This prohibits all other MUAs from sending MDN for
this message.

3.1. Client behavior when receiving a message

The client MUST NOT send MDN if a message has the $MDNSent keyword
set. It also MUST NOT send MDN if a message has \Draft flag, because
some clients use this flag to mark a message as incomplete.

See the timeline in section 3 for details on client behavior when
receiving a message.

3.2. Client behavior when copying a message

The client SHOULD verify that $MDNSent is preserved on a COPY
operation. Furthermore, when a message is copied between servers
with the APPEND command, the client MUST set the $MDNSent keyword
correctly.

3.3. Client behavior when sending a message

When saving a sent message to any folder, the client MUST set the
$MDNSent keyword to prevent another client from sending MDN for the
message.

3.4. Client behavior when saving a temporary message

When saving an unfinished message to any folder client MUST set
$MDNSent keyword to prevent another client from sending MDN for the
message.

4. Server behavior

Server implementors that want to follow this specification must
insure that their server complies with either section 4.1 or section
4.2. If the server also supports the IMAP [ACL] extension, it MUST
also comply with the section 4.3.

4.1. Server that supports arbitrary keywords

No changes are required from the server to make it compatible with
the extension described in this document if it supports arbitrary
keywords.

4.2. Server that supports only $MDNSent keyword

Servers that support only the $MDNSent keyword MUST preserve it on
the COPY operation. It is also expected that a server that supports
SEARCH <flag> will also support the SEARCH KEYWORD $MDNSent.

4.3. Interaction with IMAP ACL extension

     Any server that conforms to either 4.1 or 4.2 and also supports the
     IMAP [ACL] extension, SHOULD preserve the $MDNSent keyword on COPY
     even if the client does not have 'w' right. This will prevent the
     generation of a duplicated MDN for the same message. Note that the
     server MUST still check if the client has rights to perform the COPY
     operation on a message according to [ACL].

5. Examples

     1) MUA opens mailbox for the first time.

     a) The server supports storing of arbitrary keywords

     C: a100 select INBOX
     S: * FLAGS (\Flagged \Draft \Deleted \Seen)
     S: * OK [PERMANENTFLAGS (\Flagged \Draft \Deleted \Seen *)]
     S: * 5 EXISTS
     S: * 3 RECENT
     S: * OK [UIDVALIDITY 894294713]
     S: a100 OK [READ-WRITE] Completed

     b) The server supports storing of the $MDNSent keyword

     C: a100 select INBOX
     S: * FLAGS (\Flagged \Draft \Deleted \Seen $MDNSent)
     S: * OK [PERMANENTFLAGS (\Flagged \Draft \Deleted \Seen $MDNSent)]
     S: * 5 EXISTS
     S: * 3 RECENT
     S: * OK [UIDVALIDITY 894294713]
     S: a100 OK [READ-WRITE] Completed

     2) The MUA successfully sets the $MDNSent keyword

     C: a200 STORE 4 +FLAGS ($MDNSent)
     S: * 4 FETCH (FLAGS (\Flagged \Seen $MDNSent))
     S: * FLAGS ($MDNSent \Flagged \Deleted \Draft \Seen)
     S: * OK [PERMANENTFLAGS ($MDNSent \Flagged \Deleted \Draft \Seen *)]
     S: a200 OK STORE completed

     3) The server refuses to store the $MDNSent keyword

     C: a200 STORE 4 +FLAGS ($MDNSent)
     S: a200 NO STORE failed : no space left to store $MDNSent keyword

4) All clients and servers MUST treat the $MDNSent keyword as case
insensitive in all operations, as stated in [IMAP].

```
C: a300 FETCH 1:* FLAGS
S: * 1 FETCH (FLAGS (\Seen))
S: * 2 FETCH (FLAGS (\Answered \Seen $MdnSENt))
S: * 3 FETCH (FLAGS ())
S: * 4 FETCH (FLAGS (\Flagged \Seen $MdnSENT))
S: * 5 FETCH (FLAGS ($MDNSent))
S: * 6 FETCH (FLAGS (\Recent))
S: a300 OK FETCH completed
C: a400 SEARCH KEYWORDS $mdnsent
S: * SEARCH 2 4 5
S: a400 OK SEARCH completed
```

6. Security Considerations

There are no known security issues with this extension, not found in
[MDN] and/or [IMAP4].

Section 4.3 changes ACL checking requirements on an IMAP server that
implements IMAP [ACL] extension.

7. Formal Syntax

The following syntax specification uses the augmented Backus-Naur
Form (BNF) notation as specified in [RFC-822], as modified by
[IMAP4]. 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.

```
flag_keyword     ::= "$MDNSent" / other_keywords

other_keywords  ::= atom
```

8. Acknowledgments

This document is the product of discussions that took place on the
IMAP mailing list. Special gratitude to Cyrus Daboo and Randall
Gellens for reviewing the document.

Thank you to my father who as he has helped to make me what I am. I
miss you terribly.

9. Normative References

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

[MDN]   Fajman, R., "An Extensible Message Format for Message
          Disposition Notifications", RFC 2298, March 1998.

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

[ACL]   Myers, J., "IMAP4 ACL extension", RFC 2086, January 1997.

10. Author's Address

Alexey Melnikov
ACI Worldwide/MessagingDirect
59 Clarendon Road
Watford, Hertfordshire
United Kingdom, WD17 1FQ

Phone: +44 1923 81 2877
EMail: mel@messagingdirect.com

11. Full Copyright Statement

    Copyright (C) The Internet Society (2003). All Rights Reserved.

    This document and translations of it may be copied and furnished to
    others, and derivative works that comment on or otherwise explain it
    or assist in its implementation may be prepared, copied, published
    and distributed, in whole or in part, without restriction of any
    kind, provided that the above copyright notice and this paragraph are
    included on all such copies and derivative works. However, this
    document itself may not be modified in any way, such as by removing
    the copyright notice or references to the Internet Society or other
    Internet organizations, except as needed for the purpose of
    developing Internet standards in which case the procedures for
    copyrights defined in the Internet Standards process must be
    followed, or as required to translate it into languages other than
    English.

    The limited permissions granted above are perpetual and will not be
    revoked by the Internet Society or its successors or assigns.

    This document and the information contained herein is provided on an
    "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING
    TASK FORCE DISCLAIMS 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.

Acknowledgement

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