

Network Working Group
Request for Comments: 2249
Obsoletes: 1566
Category: Standards Track

N. Freed
Innosoft
S. Kille
ISODE Consortium
January 1998

Mail Monitoring MIB

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

Copyright (C) The Internet Society (1998). All Rights Reserved.

1. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. Specifically, this memo extends the basic Network Services Monitoring MIB [8] to allow monitoring of Message Transfer Agents (MTAs). It may also be used to monitor MTA components within gateways.

2. Table of Contents

1 Introduction	1
2 Table of Contents	1
3 The SNMPv2 Network Management Framework	2
3.1 Object Definitions	2
4 Message Flow Model	2
5 MTA Objects	3
6 Definitions	4
7 Changes made since RFC 1566	25
8 Acknowledgements	26
9 References	26
10 Security Considerations	27
11 Author and Chair Addresses	27
12 Full Copyright Statement	28

3. The SNMPv2 Network Management Framework

The SNMPv2 Network Management Framework consists of seven major components. They are:

- o RFC 1902 [1] which defines the SMI, the mechanisms used for describing and naming objects for the purpose of management.
- o RFC 1903 [2] defines textual conventions for SNMPv2.
- o RFC 1904 [3] defines conformance statements for SNMPv2.
- o RFC 1905 [4] defines transport mappings for SNMPv2.
- o RFC 1906 [5] defines the protocol operations used for network access to managed objects.
- o RFC 1907 [6] defines the Management Information Base for SNMPv2.
- o RFC 1908 [7] specifies coexistence between SNMP and SNMPv2.

The Framework permits new objects to be defined for the purpose of experimentation and evaluation.

3.1. Object Definitions

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the subset of Abstract Syntax Notation One (ASN.1) defined in the SMI. In particular, each object type is named by an OBJECT IDENTIFIER, an administratively assigned name. The object type together with an object instance serves to uniquely identify a specific instantiation of the object. For human convenience, we often use a textual string, termed the descriptor, to refer to the object type.

4. Message Flow Model

A general model of message flow inside an MTA has to be presented before a MIB can be described. Generally speaking, message flow is modelled as occurring in four steps:

- (1) Messages are received by the MTA from User Agents, Message Stores, other MTAs, and gateways.
- (2) The "next hop" for the each message is determined. This is simply the destination the message is to be transmitted to; it may or may not be the final destination of the message.

Multiple "next hops" may exist for a single message (as a result of either having multiple recipients or distribution list expansion); this may make it necessary to duplicate messages.

- (3) If necessary messages are converted into the format that's appropriate for the next hop. Conversion operations may be successful or unsuccessful.
- (4) Messages are transmitted to the appropriate destination, which may be a User Agent, Message Store, another MTA, or gateway.

Storage of messages in the MTA occurs at some point during this process. However, it is important to note that storage may occur at different and possibly even multiple points during this process. For example, some MTAs expand messages into multiple copies as they are received. In this case (1), (2), and (3) may all occur prior to storage. Other MTAs store messages precisely as they are received and perform all expansions and conversions during retransmission processing. So here only (1) occurs prior to storage. This leads to situations where, in general, a measurement of messages received may not equal a measurement of messages in store, or a measurement of messages stored may not equal a measurement of messages retransmitted, or both.

5. MTA Objects

If there are one or more MTAs on the host, the following MIB may be used to monitor them. Any number of the MTAs on a single host or group of hosts may be monitored. Each MTA is dealt with as a separate network service and has its own `applTable` entry in the Network Services Monitoring MIB.

The MIB described in this document covers only the portion which is specific to the monitoring of MTAs. The network service related part of the MIB is covered in a separate document [8].

This MIB defines four tables. The first of these contains per-MTA information that isn't specific to any particular part of MTA. The second breaks each MTA down into a collection of separate components called groups. Groups are described in detail in the comments embedded in the MIB below. The third table provides a means of correlating associations tracked by the network services MIB with specific groups within different MTAs. Finally, the fourth table provides a means of tracking any errors encountered during the operation of the MTA. The first two tables must be implemented to conform with this MIB; the last two are optional.

6. Definitions

```
MTA-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```
    OBJECT-TYPE, Counter32, Gauge32, MODULE-IDENTITY, mib-2
    FROM SNMPv2-SMI
    DisplayString, TimeInterval
    FROM SNMPv2-TC
    MODULE-COMPLIANCE, OBJECT-GROUP
    FROM SNMPv2-CONF
    applIndex, URLString
    FROM NETWORK-SERVICES-MIB;
```

```
mta MODULE-IDENTITY
```

```
    LAST-UPDATED "9708170000Z"
    ORGANIZATION "IETF Mail and Directory Management Working Group"
    CONTACT-INFO
        "        Ned Freed
```

```
        Postal: Innosoft International, Inc.
                1050 Lakes Drive
                West Covina, CA 91790
                US
```

```
        Tel: +1 626 919 3600
        Fax: +1 626 919 3614
```

```
        E-Mail: ned.freed@innosoft.com"
```

```
DESCRIPTION
```

```
"The MIB module describing Message Transfer Agents (MTAs)"
```

```
REVISION "9311280000Z"
```

```
DESCRIPTION
```

```
"The original version of this MIB was published in RFC 1566"
```

```
::= {mib-2 28}
```

```
mtaTable OBJECT-TYPE
```

```
SYNTAX SEQUENCE OF MtaEntry
```

```
MAX-ACCESS not-accessible
```

```
STATUS current
```

```
DESCRIPTION
```

```
"The table holding information specific to an MTA."
```

```
::= {mta 1}
```

```
mtaStatusCode OBJECT-TYPE
```

```
SYNTAX INTEGER (4000000..5999999)
```

```
MAX-ACCESS not-accessible
```

```
STATUS current
```

DESCRIPTION

"An index capable of representing an Enhanced Mail System Status Code. Enhanced Mail System Status Codes are defined in RFC 1893 [14]. These codes have the form

```
class.subject.detail
```

Here 'class' is either 2, 4, or 5 and both 'subject' and 'detail' are integers in the range 0..999. Given a status code the corresponding index value is defined to be $((\text{class} * 1000) + \text{subject}) * 1000 + \text{detail}$. Both SMTP error response codes and X.400 reason and diagnostic codes can be mapped into these codes, resulting in a namespace capable of describing most error conditions a mail system encounters in a generic yet detailed way."

```
::= {mta 6}
```

mtaEntry OBJECT-TYPE

```
SYNTAX MtaEntry
```

```
MAX-ACCESS not-accessible
```

```
STATUS current
```

DESCRIPTION

"The entry associated with each MTA."

```
INDEX {applIndex}
```

```
::= {mtaTable 1}
```

```
MtaEntry ::= SEQUENCE {
    mtaReceivedMessages
        Counter32,
    mtaStoredMessages
        Gauge32,
    mtaTransmittedMessages
        Counter32,
    mtaReceivedVolume
        Counter32,
    mtaStoredVolume
        Gauge32,
    mtaTransmittedVolume
        Counter32,
    mtaReceivedRecipients
        Counter32,
    mtaStoredRecipients
        Gauge32,
    mtaTransmittedRecipients
        Counter32,
    mtaSuccessfulConvertedMessages
        Counter32,
    mtaFailedConvertedMessages
```

```
    Counter32,
    mtaLoopsDetected
    Counter32
}

mtaReceivedMessages OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of messages received since MTA initialization.
        This includes messages transmitted to this MTA from other
        MTAs as well as messages that have been submitted to the
        MTA directly by end-users or applications."
    ::= { mtaEntry 1 }

mtaStoredMessages OBJECT-TYPE
    SYNTAX Gauge32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The total number of messages currently stored in the MTA.
        This includes messages that are awaiting transmission to
        some other MTA or are waiting for delivery to an end-user
        or application."
    ::= { mtaEntry 2 }

mtaTransmittedMessages OBJECT-TYPE
    SYNTAX Counter32
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The number of messages transmitted since MTA initialization.
        This includes messages that were transmitted to some other
        MTA or are waiting for delivery to an end-user or
        application."
    ::= { mtaEntry 3 }

mtaReceivedVolume OBJECT-TYPE
    SYNTAX Counter32
    UNITS "K-octets"
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The total volume of messages received since MTA
        initialization, measured in kilo-octets. This volume should
        include all transferred data that is logically above the mail
        transport protocol level. For example, an SMTP-based MTA
```

should use the number of kilo-octets in the message header and body, while an X.400-based MTA should use the number of kilo-octets of P2 data. This includes messages transmitted to this MTA from other MTAs as well as messages that have been submitted to the MTA directly by end-users or applications."

::= {mtaEntry 4}

mtaStoredVolume OBJECT-TYPE

SYNTAX Gauge32

UNITS "K-octets"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total volume of messages currently stored in the MTA, measured in kilo-octets. This volume should include all stored data that is logically above the mail transport protocol level. For example, an SMTP-based MTA should use the number of kilo-octets in the message header and body, while an X.400-based MTA would use the number of kilo-octets of P2 data. This includes messages that are awaiting transmission to some other MTA or are waiting for delivery to an end-user or application."

::= {mtaEntry 5}

mtaTransmittedVolume OBJECT-TYPE

SYNTAX Counter32

UNITS "K-octets"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total volume of messages transmitted since MTA initialization, measured in kilo-octets. This volume should include all transferred data that is logically above the mail transport protocol level. For example, an SMTP-based MTA should use the number of kilo-octets in the message header and body, while an X.400-based MTA should use the number of kilo-octets of P2 data. This includes messages that were transmitted to some other MTA or are waiting for delivery to an end-user or application."

::= {mtaEntry 6}

mtaReceivedRecipients OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of recipients specified in all messages

received since MTA initialization. Recipients this MTA has no responsibility for, i.e. inactive envelope recipients or ones referred to in message headers, should not be counted even if information about such recipients is available. This includes messages transmitted to this MTA from other MTAs as well as messages that have been submitted to the MTA directly by end-users or applications."

::= {mtaEntry 7}

mtaStoredRecipients OBJECT-TYPE

SYNTAX Gauge32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of recipients specified in all messages currently stored in the MTA. Recipients this MTA has no responsibility for, i.e. inactive envelope recipients or ones referred to in message headers, should not be counted. This includes messages that are awaiting transmission to some other MTA or are waiting for delivery to an end-user or application."

::= {mtaEntry 8}

mtaTransmittedRecipients OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of recipients specified in all messages transmitted since MTA initialization. Recipients this MTA had no responsibility for, i.e. inactive envelope recipients or ones referred to in message headers, should not be counted. This includes messages that were transmitted to some other MTA or are waiting for delivery to an end-user or application."

::= {mtaEntry 9}

mtaSuccessfulConvertedMessages OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of messages that have been successfully converted from one form to another since MTA initialization."

::= {mtaEntry 10}

mtaFailedConvertedMessages OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of messages for which an unsuccessful attempt was made to convert them from one form to another since MTA initialization."

::= { mtaEntry 11 }

mtaLoopsDetected OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A message loop is defined as a situation where the MTA decides that a given message will never be delivered to one or more recipients and instead will continue to loop endlessly through one or more MTAs. This variable counts the number of times the MTA has detected such a situation since MTA initialization. Note that the mechanism MTAs use to detect loops (e.g. trace field counting, count of references to this MTA in a trace field, examination of DNS or other directory information, etc.), the level at which loops are detected (e.g. per message, per recipient, per directory entry, etc.), and the handling of a loop once it is detected (e.g. looping messages are held, looping messages are bounced or sent to the postmaster, messages that the MTA knows will loop won't be accepted, etc.) vary widely from one MTA to the next and cannot be inferred from this variable."

::= { mtaEntry 12 }

-- MTAs typically group inbound reception, queue storage, and
-- outbound transmission in some way, rather than accounting for
-- such operations only across the MTA as a whole. In the most
-- extreme case separate information will be maintained for each
-- different entity that receives messages and for each entity
-- the MTA stores messages for and delivers messages to. Other
-- MTAs may elect to treat all reception equally, all queue
-- storage equally, all deliveries equally, or some combination
-- of this. Overlapped groupings are also possible, where an MTA
-- decomposes its traffic in different ways for different
-- purposes.

-- In any case, a grouping abstraction is an extremely useful for
-- breaking down the activities of an MTA. For purposes of
-- labelling this will be called a "group" in this MIB.

-- Each group contains all the variables needed to monitor all aspects of an MTA's operation. However, the fact that all groups contain all possible variables does not imply that all groups must use all possible variables. For example, a single group might be used to monitor only one kind of event (inbound processing, outbound processing, or storage). In this sort of configuration all unused counters would be inaccessible; e.g., returning either a noSuchName error (for an SNMPv1 get), or a noSuchInstance exception (for an SNMPv2 get).

-- Groups can be created at any time after MTA initialization. Once a group is created it should not be deleted or its mtaGroupIndex changed unless the MTA is reinitialized.

-- Groups are not necessarily mutually exclusive. A given event may be recorded by more than one group, a message may be seen as stored by more than one group, and so on. Groups should be all inclusive, however: if groups are implemented all aspects of an MTA's operation should be registered in at least one group. This freedom lets implementors use different sets of groups to provide different "views" of an MTA.

-- The possibility of overlap between groups means that summing variables across groups may not produce values equal to those in the mtaTable. mtaTable should always provide accurate information about the MTA as a whole.

-- The term "channel" is often used in MTA implementations; channels are usually, but not always, equivalent to a group. However, this MIB does not use the term "channel" because there is no requirement that an MTA supporting this MIB has to map its "channel" abstraction one-to-one onto the MIB's group abstraction.

-- An MTA may create a group or group of groups at any time. Once created, however, an MTA cannot delete an entry for a group from the group table. Deletion is only allowed when the MTA is reinitialized, and is not required even then. This restriction is imposed so that monitoring agents can rely on group assignments being consistent across multiple query operations.

-- Groups may be laid out so as to form a hierarchical arrangement, with some groups acting as subgroups for other groups. Alternately, disjoint groups of groups may be used to provide different sorts of "snapshots" of MTA operation. The mtaGroupHierarchy variable provides an indication of how each group fits into the overall arrangement being used.

mtaGroupTable OBJECT-TYPE

SYNTAX SEQUENCE OF MtaGroupEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The table holding information specific to each MTA group."

::= {mta 2}

mtaGroupEntry OBJECT-TYPE

SYNTAX MtaGroupEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The entry associated with each MTA group."

INDEX {applIndex, mtaGroupIndex}

::= {mtaGroupTable 1}

MtaGroupEntry ::= SEQUENCE {

mtaGroupIndex

INTEGER,

mtaGroupReceivedMessages

Counter32,

mtaGroupRejectedMessages

Counter32,

mtaGroupStoredMessages

Gauge32,

mtaGroupTransmittedMessages

Counter32,

mtaGroupReceivedVolume

Counter32,

mtaGroupStoredVolume

Gauge32,

mtaGroupTransmittedVolume

Counter32,

mtaGroupReceivedRecipients

Counter32,

mtaGroupStoredRecipients

Gauge32,

mtaGroupTransmittedRecipients

Counter32,

mtaGroupOldestMessageStored

TimeInterval,

mtaGroupInboundAssociations

Gauge32,

mtaGroupOutboundAssociations

Gauge32,

mtaGroupAccumulatedInboundAssociations

Counter32,

mtaGroupAccumulatedOutboundAssociations

```

    Counter32,
    mtaGroupLastInboundActivity
        TimeInterval,
    mtaGroupLastOutboundActivity
        TimeInterval,
    mtaGroupLastOutboundAssociationAttempt
        TimeInterval,
    mtaGroupRejectedInboundAssociations
        Counter32,
    mtaGroupFailedOutboundAssociations
        Counter32,
    mtaGroupInboundRejectionReason
        DisplayString,
    mtaGroupOutboundConnectFailureReason
        DisplayString,
    mtaGroupScheduledRetry
        TimeInterval,
    mtaGroupMailProtocol
        OBJECT IDENTIFIER,
    mtaGroupName
        DisplayString,
    mtaGroupSuccessfulConvertedMessages
        Counter32,
    mtaGroupFailedConvertedMessages
        Counter32,
    mtaGroupDescription
        DisplayString,
    mtaGroupURL
        URLString,
    mtaGroupCreationTime
        TimeInterval,
    mtaGroupHierarchy
        INTEGER,
    mtaGroupOldestMessageId
        DisplayString,
    mtaGroupLoopsDetected
        Counter32
}

mtaGroupIndex OBJECT-TYPE
    SYNTAX INTEGER (1..2147483647)
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The index associated with a group for a given MTA."
    ::= {mtaGroupEntry 1}

mtaGroupReceivedMessages OBJECT-TYPE

```

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The number of messages received to this group since
 group creation."
 ::= { mtaGroupEntry 2 }

mtaGroupRejectedMessages OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The number of messages rejected by this group since
 group creation."
 ::= { mtaGroupEntry 3 }

mtaGroupStoredMessages OBJECT-TYPE
SYNTAX Gauge32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The total number of messages currently stored in this
 group's queue."
 ::= { mtaGroupEntry 4 }

mtaGroupTransmittedMessages OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The number of messages transmitted by this group since
 group creation."
 ::= { mtaGroupEntry 5 }

mtaGroupReceivedVolume OBJECT-TYPE
SYNTAX Counter32
UNITS "K-octets"
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The total volume of messages received to this group since
 group creation, measured in kilo-octets. This volume
 should include all transferred data that is logically above
 the mail transport protocol level. For example, an
 SMTP-based MTA should use the number of kilo-octets in the
 message header and body, while an X.400-based MTA should use
 the number of kilo-octets of P2 data."

```
::= {mtaGroupEntry 6}
```

```
mtaGroupStoredVolume OBJECT-TYPE
```

```
SYNTAX Gauge32
```

```
UNITS "K-octets"
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"The total volume of messages currently stored in this
group's queue, measured in kilo-octets. This volume should
include all stored data that is logically above the mail
transport protocol level. For example, an SMTP-based
MTA should use the number of kilo-octets in the message
header and body, while an X.400-based MTA would use the
number of kilo-octets of P2 data."
```

```
::= {mtaGroupEntry 7}
```

```
mtaGroupTransmittedVolume OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
UNITS "K-octets"
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"The total volume of messages transmitted by this group
since group creation, measured in kilo-octets. This
volume should include all transferred data that is logically
above the mail transport protocol level. For example, an
SMTP-based MTA should use the number of kilo-octets in the
message header and body, while an X.400-based MTA should use
the number of kilo-octets of P2 data."
```

```
::= {mtaGroupEntry 8}
```

```
mtaGroupReceivedRecipients OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"The total number of recipients specified in all messages
received to this group since group creation.
Recipients this MTA has no responsibility for should not
be counted."
```

```
::= {mtaGroupEntry 9}
```

```
mtaGroupStoredRecipients OBJECT-TYPE
```

```
SYNTAX Gauge32
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
    "The total number of recipients specified in all messages
    currently stored in this group's queue. Recipients this
    MTA has no responsibility for should not be counted."
 ::= { mtaGroupEntry 10 }
```

mtaGroupTransmittedRecipients OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

```
    "The total number of recipients specified in all messages
    transmitted by this group since group creation.
    Recipients this MTA had no responsibility for should not
    be counted."
```

```
 ::= { mtaGroupEntry 11 }
```

mtaGroupOldestMessageStored OBJECT-TYPE

SYNTAX TimeInterval

MAX-ACCESS read-only

STATUS current

DESCRIPTION

```
    "Time since the oldest message in this group's queue was
    placed in the queue."
```

```
 ::= { mtaGroupEntry 12 }
```

mtaGroupInboundAssociations OBJECT-TYPE

SYNTAX Gauge32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

```
    "The number of current associations to the group, where the
    group is the responder."
```

```
 ::= { mtaGroupEntry 13 }
```

mtaGroupOutboundAssociations OBJECT-TYPE

SYNTAX Gauge32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

```
    "The number of current associations to the group, where the
    group is the initiator."
```

```
 ::= { mtaGroupEntry 14 }
```

mtaGroupAccumulatedInboundAssociations OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of associations to the group since group creation, where the MTA was the responder."
 ::= {mtaGroupEntry 15}

mtaGroupAccumulatedOutboundAssociations OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of associations from the group since group creation, where the MTA was the initiator."

::= {mtaGroupEntry 16}

mtaGroupLastInboundActivity OBJECT-TYPE

SYNTAX TimeInterval

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Time since the last time that this group had an active inbound association for purposes of message reception."

::= {mtaGroupEntry 17}

mtaGroupLastOutboundActivity OBJECT-TYPE

SYNTAX TimeInterval

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Time since the last time that this group had a successful outbound association for purposes of message delivery."

::= {mtaGroupEntry 18}

mtaGroupLastOutboundAssociationAttempt OBJECT-TYPE

SYNTAX TimeInterval

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Time since the last time that this group attempted to make an outbound association for purposes of message delivery."

::= {mtaGroupEntry 34}

mtaGroupRejectedInboundAssociations OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of inbound associations the group has

rejected, since group creation. Rejected associations are not counted in the accumulated association totals."
 ::= {mtaGroupEntry 19}

mtaGroupFailedOutboundAssociations OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number associations where the group was the initiator and association establishment has failed, since group creation. Failed associations are not counted in the accumulated association totals."

::= {mtaGroupEntry 20}

mtaGroupInboundRejectionReason OBJECT-TYPE

SYNTAX DisplayString

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The failure reason, if any, for the last association this group refused to respond to. An empty string indicates that the last attempt was successful. If no association attempt has been made since the MTA was initialized the value should be 'never'."

::= {mtaGroupEntry 21}

mtaGroupOutboundConnectFailureReason OBJECT-TYPE

SYNTAX DisplayString

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The failure reason, if any, for the last association attempt this group initiated. An empty string indicates that the last attempt was successful. If no association attempt has been made since the MTA was initialized the value should be 'never'."

::= {mtaGroupEntry 22}

mtaGroupScheduledRetry OBJECT-TYPE

SYNTAX TimeInterval

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The time when this group is scheduled to next attempt to make an association."

::= {mtaGroupEntry 23}

mtaGroupMailProtocol OBJECT-TYPE

SYNTAX OBJECT IDENTIFIER

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"An identification of the protocol being used by this group. For an group employing OSI protocols, this will be the Application Context. For Internet applications, the IANA maintains a registry of the OIDs which correspond to well-known message transfer protocols. If the application protocol is not listed in the registry, an OID value of the form {applTCPProtoID port} or {applUDPProtoID port} are used for TCP-based and UDP-based protocols, respectively. In either case 'port' corresponds to the primary port number being used by the group. applTCPProtoID and applUDPPProtoID are defined in [8]."

::= {mtaGroupEntry 24}

mtaGroupName OBJECT-TYPE

SYNTAX DisplayString

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A descriptive name for the group. If this group connects to a single remote MTA this should be the name of that MTA. If this in turn is an Internet MTA this should be the domain name. For an OSI MTA it should be the string encoded distinguished name of the managed object using the format defined in RFC 1779 [9]. For X.400(1984) MTAs which do not have a Distinguished Name, the RFC 1327 [12] syntax 'mta in globalid' should be used."

::= {mtaGroupEntry 25}

mtaGroupSuccessfulConvertedMessages OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of messages that have been successfully converted from one form to another in this group since group creation."

::= {mtaGroupEntry 26}

mtaGroupFailedConvertedMessages OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of messages for which an unsuccessful attempt was made to convert them from one form to another in this group since group creation."
 ::= {mtaGroupEntry 27}

mtaGroupDescription OBJECT-TYPE
SYNTAX DisplayString
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"A description of the group's purpose. This information is intended to identify the group in a status display."
 ::= {mtaGroupEntry 28}

mtaGroupURL OBJECT-TYPE
SYNTAX URLString
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"A URL pointing to a description of the group. This information is intended to identify and briefly describe the group in a status display."
 ::= {mtaGroupEntry 29}

mtaGroupCreationTime OBJECT-TYPE
SYNTAX TimeInterval
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Time since this group was first created."
 ::= {mtaGroupEntry 30}

mtaGroupHierarchy OBJECT-TYPE
SYNTAX INTEGER (-2147483648..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Describes how this group fits into the hierarchy. A positive value is interpreted as an mtaGroupIndex value for some other group whose variables include those of this group (and usually others). A negative value is interpreted as a group collection code: Groups with common negative hierarchy values comprise one particular breakdown of MTA activity as a whole. A zero value means that this MIB implementation doesn't implement hierarchy indicators and thus the overall group hierarchy cannot be determined."
 ::= {mtaGroupEntry 31}

mtaGroupOldestMessageId OBJECT-TYPE

SYNTAX DisplayString

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Message ID of the oldest message in the group's queue. Whenever possible this should be in the form of an RFC 822 [13] msg-id; X.400 may convert X.400 message identifiers to this form by following the rules laid out in RFC1327 [12]."

::= {mtaGroupEntry 32}

mtaGroupLoopsDetected OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A message loop is defined as a situation where the MTA decides that a given message will never be delivered to one or more recipients and instead will continue to loop endlessly through one or more MTAs. This variable counts the number of times the MTA has detected such a situation in conjunction with something associated with this group since group creation. Note that the mechanism MTAs use to detect loops (e.g. trace field counting, count of references to this MTA in a trace field, examination of DNS or other directory information, etc.), the level at which loops are detected (e.g. per message, per recipient, per directory entry, etc.), and the handling of a loop once it is detected (e.g. looping messages are held, looping messages are bounced or sent to the postmaster, messages that the MTA knows will loop won't be accepted, etc.) vary widely from one MTA to the next and cannot be inferred from this variable."

::= {mtaGroupEntry 33}

-- The mtaGroupAssociationTable provides a means of correlating
-- entries in the network services association table with the
-- MTA group responsible for the association.

mtaGroupAssociationTable OBJECT-TYPE

SYNTAX SEQUENCE OF MtaGroupAssociationEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The table holding information regarding the associations for each MTA group."

```
::= {mta 3}
```

```
mtaGroupAssociationEntry OBJECT-TYPE
    SYNTAX MtaGroupAssociationEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The entry holding information regarding the associations
         for each MTA group."
    INDEX {applIndex, mtaGroupIndex, mtaGroupAssociationIndex}
    ::= {mtaGroupAssociationTable 1}
```

```
MtaGroupAssociationEntry ::= SEQUENCE {
    mtaGroupAssociationIndex
        INTEGER
}
```

```
mtaGroupAssociationIndex OBJECT-TYPE
    SYNTAX INTEGER (1..2147483647)
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "Reference into association table to allow correlation of
         this group's active associations with the association table."
    ::= {mtaGroupAssociationEntry 1}
```

```
-- The mtaGroupErrorTable gives each group a way of tallying
-- the specific errors it has encountered. The mechanism
-- defined here uses RFC 1893 [14] status codes to identify
-- various specific errors. There are also classes for generic
-- errors of various sorts, and the entire mechanism is also
-- extensible, in that new error codes can be defined at any
-- time.
```

```
mtaGroupErrorTable OBJECT-TYPE
    SYNTAX SEQUENCE OF MtaGroupErrorEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
        "The table holding information regarding accumulated errors
         for each MTA group."
    ::= {mta 5}
```

```
mtaGroupErrorEntry OBJECT-TYPE
    SYNTAX MtaGroupErrorEntry
    MAX-ACCESS not-accessible
    STATUS current
    DESCRIPTION
```

"The entry holding information regarding accumulated errors for each MTA group."

INDEX {applIndex, mtaGroupIndex, mtaStatusCode}
 ::= {mtaGroupErrorTable 1}

```
MtaGroupErrorEntry ::= SEQUENCE {  
    mtaGroupInboundErrorCount  
        Counter32,  
    mtaGroupInternalErrorCount  
        Counter32,  
    mtaGroupOutboundErrorCount  
        Counter32  
}
```

mtaGroupInboundErrorCount OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Count of the number of errors of a given type that have been accumulated in association with a particular group while processing incoming messages. In the case of SMTP these will typically be errors reporting by an SMTP server to the remote client; in the case of X.400 these will typically be errors encountered while processing an incoming message."

::= {mtaGroupErrorEntry 1}

mtaGroupInternalErrorCount OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Count of the number of errors of a given type that have been accumulated in association with a particular group during internal MTA processing."

::= {mtaGroupErrorEntry 2}

mtaGroupOutboundErrorCount OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Count of the number of errors of a given type that have been accumulated in association with a particular group's outbound connection activities. In the case of an SMTP client these will typically be errors reported while

```
        attempting to contact or while communicating with the
        remote SMTP server. In the case of X.400 these will
        typically be errors encountered while constructing
        or attempting to deliver an outgoing message."
 ::= {mtaGroupErrorEntry 3}

-- Conformance information

mtaConformance OBJECT IDENTIFIER ::= {mta 4}

mtaGroups          OBJECT IDENTIFIER ::= {mtaConformance 1}
mtaCompliances     OBJECT IDENTIFIER ::= {mtaConformance 2}

-- Compliance statements

mtaCompliance MODULE-COMPLIANCE
  STATUS current
  DESCRIPTION
    "The compliance statement for SNMPv2 entities which
    implement the Mail Monitoring MIB for basic
    monitoring of MTAs."
  MODULE -- this module
  MANDATORY-GROUPS {mtaGroup}
  ::= {mtaCompliances 1}

mtaAssocCompliance MODULE-COMPLIANCE
  STATUS current
  DESCRIPTION
    "The compliance statement for SNMPv2 entities which
    implement the Mail Monitoring MIB for monitoring of
    MTAs and their associations."
  MODULE -- this module
  MANDATORY-GROUPS {mtaGroup, mtaAssocGroup}
  ::= {mtaCompliances 2}

mtaErrorCompliance MODULE-COMPLIANCE
  STATUS current
  DESCRIPTION
    "The compliance statement for SNMPv2 entities which
    implement the Mail Monitoring MIB for monitoring of
    MTAs and detailed errors."
  MODULE -- this module
  MANDATORY-GROUPS {mtaGroup, mtaErrorGroup}
  ::= {mtaCompliances 3}

mtaFullCompliance MODULE-COMPLIANCE
  STATUS current
  DESCRIPTION
```

"The compliance statement for SNMPv2 entities which implement the full Mail Monitoring MIB for monitoring of MTAs, associations, and detailed errors."

MODULE -- this module

MANDATORY-GROUPS {mtaGroup, mtaAssocGroup, mtaErrorGroup}
 ::= {mtaCompliances 4}

-- Units of conformance

mtaGroup OBJECT-GROUP

OBJECTS {
 mtaReceivedMessages, mtaStoredMessages,
 mtaTransmittedMessages, mtaReceivedVolume, mtaStoredVolume,
 mtaTransmittedVolume, mtaReceivedRecipients,
 mtaStoredRecipients, mtaTransmittedRecipients,
 mtaSuccessfulConvertedMessages, mtaFailedConvertedMessages,
 mtaGroupReceivedMessages, mtaGroupRejectedMessages,
 mtaGroupStoredMessages, mtaGroupTransmittedMessages,
 mtaGroupReceivedVolume, mtaGroupStoredVolume,
 mtaGroupTransmittedVolume, mtaGroupReceivedRecipients,
 mtaGroupStoredRecipients, mtaGroupTransmittedRecipients,
 mtaGroupOldestMessageStored, mtaGroupInboundAssociations,
 mtaGroupOutboundAssociations, mtaLoopsDetected,
 mtaGroupAccumulatedInboundAssociations,
 mtaGroupAccumulatedOutboundAssociations,
 mtaGroupLastInboundActivity, mtaGroupLastOutboundActivity,
 mtaGroupLastOutboundAssociationAttempt,
 mtaGroupRejectedInboundAssociations,
 mtaGroupFailedOutboundAssociations,
 mtaGroupInboundRejectionReason,
 mtaGroupOutboundConnectFailureReason,
 mtaGroupScheduledRetry, mtaGroupMailProtocol, mtaGroupName,
 mtaGroupSuccessfulConvertedMessages,
 mtaGroupFailedConvertedMessages, mtaGroupDescription,
 mtaGroupURL, mtaGroupCreationTime, mtaGroupHierarchy,
 mtaGroupOldestMessageId, mtaGroupLoopsDetected}
 STATUS current
 DESCRIPTION
 "A collection of objects providing basic monitoring of MTAs."
 ::= {mtaGroups 1}

mtaAssocGroup OBJECT-GROUP

OBJECTS {
 mtaGroupAssociationIndex}
 STATUS current
 DESCRIPTION
 "A collection of objects providing monitoring of MTA associations."

```
::= {mtaGroups 2}
```

```
mtaErrorGroup OBJECT-GROUP
```

```
OBJECTS {
```

```
    mtaGroupInboundErrorCount, mtaGroupInternalErrorCount,  
    mtaGroupOutboundErrorCount}
```

```
STATUS current
```

```
DESCRIPTION
```

```
    "A collection of objects providing monitoring of  
    detailed MTA errors."
```

```
::= {mtaGroups 3}
```

```
END
```

7. Changes made since RFC 1566

The only changes made to this document since it was issued as RFC 1566 [11] are the following:

- (1) A number of DESCRIPTION fields have been reworded, hopefully making them clearer.
- (2) mtaGroupDescription and mtaGroupURL fields have been added. These fields are intended to identify and describe the MTA and the various MTA groups.
- (3) The time since the last outbound association attempt is now distinct from the time since the last successful outbound association attempt.
- (4) Conversion operation counters have been added.
- (5) A mechanism to explicitly describe group hierarchies has been added.
- (6) A mechanism to count specific sorts of errors has been added.
- (7) A field for the ID of the oldest message in a group's queue has been added.
- (8) Per-MTA and per-group message loop counters have been added.
- (9) A new table has been added to keep track of any errors an MTA encounters.

8. Acknowledgements

This document is a work product of the Mail and Directory Management (MADMAN) Working Group of the IETF. It is based on an earlier MIB designed by S. Kille, T. Lenggenhager, D. Partain, and W. Yeong. The Electronic Mail Association's TSC committee was instrumental in providing feedback on and suggesting enhancements to RFC 1566 [11] that have led to the present document.

9. References

- [1] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1902, January 1996.
- [2] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1903, January 1996.
- [3] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Conformance Statements for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1904, January 1996.
- [4] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, January 1996.
- [5] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, January 1996.
- [6] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1907, January 1996.
- [7] SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Coexistence between Version 1 and Version 2 of the Internet-standard Network Management Framework", RFC 1908, January 1996.
- [8] Freed, N., and S. Kille, "The Network Services Monitoring MIB", RFC 2248, January 1998.
- [9] Kille, S., "A String Representation of Distinguished Names", RFC 1779, March 1995.

- [10] Berners-Lee, T., Masinter, L. and M. McCahill, Uniform Resource Locators (URL)", RFC 1738, December 1994.
- [11] Freed, N. and S. Kille, "Mail Monitoring MIB", RFC 1566, January 1994.
- [12] Kille, S., "Mapping between X.400(1988) / ISO 10021 and RFC 822", RFC 1327, May 1992.
- [13] Crocker, D., "Standard for the Format of ARPA Internet Text Message", RFC 822, August 1982.
- [14] Vaudreuil, G., "Enhanced Mail System Status Codes", RFC 1893, January 1996.

10. Security Considerations

This MIB does not offer write access, and as such cannot be used to actively attack a system. However, this MIB does provide passive information about the existence, type, and configuration of applications on a given host that could potentially indicate some sort of vulnerability. Finally, the information MIB provides about network usage could be used to analyze network traffic patterns.

11. Author and Chair Addresses

Ned Freed
Innosoft International, Inc.
1050 Lakes Drive
West Covina, CA 91790
USA

Phone: +1 626 919 3600
Fax: +1 626 919 3614
EMail: ned.freed@innosoft.com

Steve Kille, MADMAN WG Chair
ISODE Consortium
The Dome, The Square
Richmond TW9 1DT
UK

Phone: +44 181 332 9091
EMail: S.Kille@isode.com

12. Full Copyright Statement

Copyright (C) The Internet Society (1998). 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.

RPO

