Definitions of Managed Objects  
for IEEE 802.3 Repeater Devices  
using SMIV2

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.

Abstract

This memo defines a portion of the Management Information Base (MIB) 
for use with network management protocols in the Internet community. 
In particular, it defines objects for managing IEEE 802.3 10 and 100 
Mb/second baseband repeaters based on IEEE Std 802.3 Section 30, "10 

Table of Contents

1. The SNMP Network Management Framework....................... 2
1.1. Object Definitions...................................... 2
2. Overview................................................................ 2
2.1. Relationship to RFC 1516.................................... 2
2.2. Repeater Management........................................ 3
2.3. Structure of the MIB....................................... 4
2.3.1. Basic Definitions...................................... 4
2.3.2. Monitor Definitions..................................... 4
2.3.3. Address Tracking Definitions.......................... 4
2.3.4. Top N Definitions...................................... 4
2.4. Relationship to Other MIBs.................................. 4
2.4.1. Relationship to MIB-II.................................. 4
2.4.1.1. Relationship to the 'system' group................. 5
2.4.1.2. Relationship to the 'interfaces' group............. 5
3. Definitions.................................................. 6
1. The SNMP Network Management Framework

The SNMP Network Management Framework presently consists of three major components. They are:

- the SMI, described in RFC 1902 [6] - the mechanisms used for describing and naming objects for the purpose of management.


Textual conventions are defined in RFC 1903 [7], and conformance statements are defined in RFC 1904 [8].

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

1.1. 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.

2. Overview

2.1. Relationship to RFC 1516

This MIB is intended as a superset of that defined by RFC 1516 [11], which will go to historic status. This MIB includes all of the objects contained in that MIB, plus several new ones which provide...
for significant additional capabilities. Implementors are encouraged to support all applicable conformance groups in order to make the best use of the new functionality provided by this MIB. The new objects provide support for:

- multiple repeaters
- 100BASE-T management
- port TopN capability
- address search and topology mapping

Certain objects have been deprecated; in particular, those scalar objects used for managing a single repeater are now of minimal use since they are duplicated in the new multiple-repeater definitions. Additional objects have been deprecated based on implementation experience with RFC 1516.

2.2. Repeater Management

Instances of the object types defined in this memo represent attributes of an IEEE 802.3 (Ethernet-like) repeater, as defined by Section 9, "Repeater Unit for 10 Mb/s Baseband Networks" in the IEEE 802.3/ISO 8802-3 CSMA/CD standard [1], and Section 27, "Repeater for 100 Mb/s Baseband Networks" in the IEEE Standard 802.3u-1995 [2].

These Repeater MIB objects may be used to manage non-standard repeater-like devices, but defining objects to describe implementation-specific properties of non-standard repeater-like devices is outside the scope of this memo.

The definitions presented here are based on Section 30.4, "Layer Management for 10 and 100 Mb/s Baseband Repeaters" and Annex 30A, "GDMO Specificataions for 802.3 managed objects" of [3].

Implementors of these MIB objects should note that [3] explicitly describes when, where, and how various repeater attributes are measured. The IEEE document also describes the effects of repeater actions that may be invoked by manipulating instances of the MIB objects defined here.

The counters in this document are defined to be the same as those counters in [3], with the intention that the same instrumentation can be used to implement both the IEEE and IETF management standards.
2.3. Structure of the MIB

Objects in this MIB are arranged into packages, each of which contains a set of related objects within a broad functional category. Objects within a package are generally defined under the same OID subtree. These packages are intended for organizational convenience ONLY, and have no relation to the conformance groups defined later in the document.

2.3.1. Basic Definitions

The basic definitions include objects which are applicable to all repeaters: status, parameter and control objects for each repeater within the managed system, for the port groups within the system, and for the individual ports themselves.

2.3.2. Monitor Definitions

The monitor definitions include monitoring statistics for each repeater within the system and for individual ports.

2.3.3. Address Tracking Definitions

This collection includes objects for tracking the MAC addresses of the DTEs attached to the ports within the system and for mapping the topology of a network.

Note: These definitions are based on a technology which has been patented by Hewlett-Packard Company. HP has granted rights to this technology to implementors of this MIB. See [12] and [13] for details.

2.3.4. Top N Definitions

These objects may be used for tracking the ports with the most activity within the system or within particular repeaters.

2.4. Relationship to Other MIBs

2.4.1. Relationship to MIB-II

It is assumed that a repeater implementing this MIB will also implement (at least) the ‘system’ group defined in MIB-II [5].
2.4.1.1. Relationship to the ‘system’ group

In MIB-II, the ‘system’ group is defined as being mandatory for all systems such that each managed entity contains one instance of each object in the ‘system’ group. Thus, those objects apply to the entity even if the entity’s sole functionality is management of repeaters.

2.4.1.2. Relationship to the ‘interfaces’ group

In MIB-II, the ‘interfaces’ group is defined as being mandatory for all systems and contains information on an entity’s interfaces, where each interface is thought of as being attached to a ‘subnetwork’. (Note that this term is not to be confused with ‘subnet’ which refers to an addressing partitioning scheme used in the Internet suite of protocols.)

This Repeater MIB uses the notion of ports on a repeater. The concept of a MIB-II interface has NO specific relationship to a repeater’s port. Therefore, the ‘interfaces’ group applies only to the one (or more) network interfaces on which the entity managing the repeater sends and receives management protocol operations, and does not apply to the repeater’s ports.

This is consistent with the physical-layer nature of a repeater. A repeater is a bitwise store-and-forward device. It recognizes activity and bits, but does not process incoming data based on any packet-related information (such as checksum or addresses). A repeater has no MAC address, no MAC implementation, and does not pass packets up to higher-level protocol entities for processing.

(When a network management entity is observing a repeater, it may appear as though the repeater is passing packets to a higher-level protocol entity. However, this is only a means of implementing management, and this passing of management information is not part of the repeater functionality.)
3. Definitions

SNMP-REPEATER-MIB DEFINITIONS ::= BEGIN

IMPORTS
  Counter32, Counter64, Integer32, Gauge32, TimeTicks,
  OBJECT-TYPE, MODULE-IDENTITY, NOTIFICATION-TYPE, mib-2
FROM SNMPv2-SMI
  TimeStamp, DisplayString, MacAddress, TEXTUAL-CONVENTION,
  RowStatus, TestAndIncr
FROM SNMPv2-TC
  OBJECT-GROUP, MODULE-COMPLIANCE
FROM SNMPv2-CONF
  OwnerString
FROM IF-MIB;

snmpRptrMod MODULE-IDENTITY
LAST-UPDATED    "9609140000Z"
ORGANIZATION    "IETF HUB MIB Working Group"
CONTACT-INFO
  "WG E-mail: hubmib@hprnd.rose.hp.com"

Chair: Dan Romascanu
Postal: Madge Networks (Israel) Ltd.
  Atidim Technology Park, Bldg. 3
  Tel Aviv 61131, Israel
  Tel: 972-3-6458414, 6458458
  Fax: 972-3-6487146
  E-mail: dromasca@madge.com

Editor: Kathryn de Graaf
Postal: 3Com Corporation
  118 Turnpike Rd.
  Southborough, MA 01772 USA
  Tel: (508)229-1627
  Fax: (508)490-5882
  E-mail: kdegraaf@isd.3com.com"

DESCRIPTION
  "Management information for 802.3 repeaters.

The following references are used throughout this MIB module:

[IEEE 802.3 Std]
  refers to IEEE 802.3/ISO 8802-3 Information
  processing systems - Local area networks -
  Part 3: Carrier sense multiple access with
collision detection (CSMA/CD) access method and physical layer specifications (1993).

[IEEE 802.3 Mgt]
refers to IEEE 802.3u-1995, ‘10 Mb/s & 100 Mb/s Management, Section 30,’
Supplement to ANSI/IEEE 802.3.

The following terms are used throughout this MIB module. For complete formal definitions, the IEEE 802.3 standards should be consulted wherever possible:

System - A managed entity compliant with this MIB, and incorporating at least one managed 802.3 repeater.

Chassis - An enclosure for one managed repeater, part of a managed repeater, or several managed repeaters. It typically contains an integral power supply and a variable number of available module slots.

Repeater-unit - The portion of the repeater set that is inboard of the physical media interfaces. The physical media interfaces (MAUs, AUIs) may be physically separated from the repeater-unit, or they may be integrated into the same physical package.

Trivial repeater-unit - An isolated port that can gather statistics.

Group - A recommended, but optional, entity defined by the IEEE 802.3 management standard, in order to support a modular numbering scheme. The classical example allows an implementor to represent field-replaceable units as groups of ports, with the port numbering matching the modular hardware implementation.

System interconnect segment - An internal segment allowing interconnection of ports belonging to different physical entities into the same logical manageable repeater. Examples of implementation might be backplane busses in modular hubs, or chaining cables in stacks of hubs.
Stack - A scalable system that may include managed repeaters, in which modularity is achieved by interconnecting a number of different chassis.

Module - A building block in a modular chassis. It typically maps into one 'slot'; however, the range of configurations may be very large, with several modules entering one slot, or one module covering several slots.

```
REVISION "9309010000Z"
DESCRIPTION
  "Published as RFC 1516"
REVISION "9210010000Z"
DESCRIPTION
  "Published as RFC 1368"
::= { snmpDot3RptrMgt 5 }
```

```
OptMacAddr ::= TEXTUAL-CONVENTION
  DISPLAY-HINT    "1x:"
  STATUS          current
  DESCRIPTION    
  "Either a 6 octet address in the 'canonical'
  order defined by IEEE 802.1a, i.e., as if it
  were transmitted least significant bit first
  if a value is available or a zero length string."
  REFERENCE
  "See MacAddress in SNMPv2-TC. The only difference
  is that a zero length string is allowed as a value
  for OptMacAddr and not for MacAddress."
  SYNTAX OCTET STRING (SIZE (0 | 6))
```

```
-- Basic information at the repeater, group, and port level.

rptrBasicPackage
  OBJECT IDENTIFIER ::= { snmpDot3RptrMgt 1 }

rptrRptrInfo
  OBJECT IDENTIFIER ::= { rptrBasicPackage 1 }

rptrGroupInfo
```
OBJECT IDENTIFIER ::= { rptrBasicPackage 2 }
rptrPortInfo

OBJECT IDENTIFIER ::= { rptrBasicPackage 3 }
rptrAllRptrInfo

OBJECT IDENTIFIER ::= { rptrBasicPackage 4 }

-- Monitoring information at the repeater, group, and port level.
rptrMonitorPackage

OBJECT IDENTIFIER ::= { snmpDot3RptrMgt 2 }
rptrMonitorRptrInfo

OBJECT IDENTIFIER ::= { rptrMonitorPackage 1 }
rptrMonitorGroupInfo

OBJECT IDENTIFIER ::= { rptrMonitorPackage 2 }
rptrMonitorPortInfo

OBJECT IDENTIFIER ::= { rptrMonitorPackage 3 }
rptrMonitorAllRptrInfo

OBJECT IDENTIFIER ::= { rptrMonitorPackage 4 }

-- Address tracking information at the repeater, group,
-- and port level.
rptrAddrTrackPackage

OBJECT IDENTIFIER ::= { snmpDot3RptrMgt 3 }
rptrAddrTrackRptrInfo

OBJECT IDENTIFIER ::= { rptrAddrTrackPackage 1 }
rptrAddrTrackGroupInfo

-- this subtree is currently unused
OBJECT IDENTIFIER ::= { rptrAddrTrackPackage 2 }
rptrAddrTrackPortInfo

OBJECT IDENTIFIER ::= { rptrAddrTrackPackage 3 }

-- TopN information.
rptrTopNPackage

OBJECT IDENTIFIER ::= { snmpDot3RptrMgt 4 }
rptrTopNRptrInfo

-- this subtree is currently unused
OBJECT IDENTIFIER ::= { rptrTopNPackage 1 }
rptrTopNGroupInfo

-- this subtree is currently unused
OBJECT IDENTIFIER ::= { rptrTopNPackage 2 }
rptrTopNPortInfo

OBJECT IDENTIFIER ::= { rptrTopNPackage 3 }

-- Old version of basic information at the repeater level.
--
-- In a system containing a single managed repeater,
-- configuration, status, and control objects for the overall
-- repeater.
-- The objects contained under the rptrRptrInfo subtree are
-- intended for backwards compatibility with implementations of
-- RFC 1516 [11]. In newer implementations (both single- and
-- multiple-repeater implementations) the rptrInfoTable should
-- be implemented. It is the preferred source of this information,
-- as it contains the values for all repeaters managed by the
-- agent. In all cases, the objects in the rptrRptrInfo subtree
-- are duplicates of the corresponding objects in the first entry
-- of the rptrInfoTable.

rptrGroupCapacity OBJECT-TYPE
SYNTAX Integer32 (1..2147483647)
MAX-ACCESS read-only
STATUS deprecated
DESCRIPTION "********* THIS OBJECT IS DEPRECATED **********

The rptrGroupCapacity is the number of groups
that can be contained within the repeater. Within
each managed repeater, the groups are uniquely
numbered in the range from 1 to rptrGroupCapacity.

Some groups may not be present in the repeater, in
which case the actual number of groups present
will be less than rptrGroupCapacity. The number
of groups present will never be greater than
rptrGroupCapacity.

Note: In practice, this will generally be the
number of field-replaceable units (i.e., modules,
cards, or boards) that can fit in the physical
repeater enclosure, and the group numbers will
correspond to numbers marked on the physical
enclosure."

REFERENCE
"[IEEE 802.3 Mgt], 30.4.1.1.3,
aRepeaterGroupCapacity."
::= { rptrRptrInfo 1 }

rptrOperStatus OBJECT-TYPE
SYNTAX INTEGER {
    other(1), -- undefined or unknown
    ok(2),   -- no known failures
    rptrFailure(3), -- repeater-related failure
    groupFailure(4), -- group-related failure
    portFailure(5), -- port-related failure
    generalFailure(6) -- failure, unspecified type
The rptrOperStatus object indicates the operational state of the repeater. The rptrHealthText object may be consulted for more specific information about the state of the repeater’s health.

In the case of multiple kinds of failures (e.g., repeater failure and port failure), the value of this attribute shall reflect the highest priority failure in the following order, listed highest priority first:

- rptrFailure(3)
- groupFailure(4)
- portFailure(5)
- generalFailure(6).

The health text object is a text string that provides information relevant to the operational state of the repeater. Agents may use this string to provide detailed information on current failures, including how they were detected, and/or instructions for problem resolution. The contents are agent-specific.
MAX-ACCESS  read-write
STATUS      deprecated
DESCRIPTION  "********* THIS OBJECT IS DEPRECATED **********
Setting this object to reset(2) causes a transition to the START state of Fig 9-2 in section 9 [IEEE 802.3 Std] for a 10Mb/s repeater, and the START state of Fig 27-2 in section 27 of that standard for a 100Mb/s repeater.

Setting this object to noReset(1) has no effect. The agent will always return the value noReset(1) when this object is read.

After receiving a request to set this variable to reset(2), the agent is allowed to delay the reset for a short period. For example, the implementor may choose to delay the reset long enough to allow the SNMP response to be transmitted. In any event, the SNMP response must be transmitted.

This action does not reset the management counters defined in this document nor does it affect the portAdminStatus parameters. Included in this action is the execution of a disruptive Self-Test with the following characteristics: a) The nature of the tests is not specified. b) The test resets the repeater but without affecting management information about the repeater. c) The test does not inject packets onto any segment. d) Packets received during the test may or may not be transferred. e) The test does not interfere with management functions.

After performing this self-test, the agent will update the repeater health information (including rptrOperStatus and rptrHealthText), and send a rptrHealth trap."

REFERENCE
"[IEEE 802.3 Mgt], 30.4.1.2.1, acResetRepeater."
::= { rptrRptrInfo 4 }
rptrNonDisruptTest OBJECT-TYPE
SYNTAX      INTEGER {
            noSelfTest(1),
            selfTest(2)
Setting this object to selfTest(2) causes the repeater to perform a agent-specific, non-disruptive self-test that has the following characteristics: a) The nature of the tests is not specified. b) The test does not change the state of the repeater or management information about the repeater. c) The test does not inject packets onto any segment. d) The test does not prevent the relay of any packets. e) The test does not interfere with management functions.

After performing this test, the agent will update the repeater health information (including rptrOperStatus and rptrHealthText) and send a rptrHealth trap.

Note that this definition allows returning an 'okay' result after doing a trivial test.

Setting this object to noSelfTest(1) has no effect. The agent will always return the value noSelfTest(1) when this object is read.

REFERENCE
"[IEEE 802.3 Mgt], 30.4.1.2.2, acExecuteNonDisruptiveSelfTest."

::= { rptrRptrInfo 5 }

rptrTotalPartitionedPorts OBJECT-TYPE
SYNTAX Gauge32
MAX-ACCESS read-only
STATUS deprecated
DESCRIPTION "********** THIS OBJECT IS DEPRECATED **********

This object returns the total number of ports in the repeater whose current state meets all three of the following criteria: rptrPortOperStatus does not have the value notPresent(3), rptrPortAdminStatus is enabled(1), and rptrPortAutoPartitionState is autoPartitioned(2)."

::= { rptrRptrInfo 6 }
-- Basic information at the group level.
--
-- Configuration and status objects for each
-- managed group in the system, independent
-- of whether there is one or more managed
-- repeater-units in the system.

rptrGroupTable OBJECT-TYPE
SYNTAX SEQUENCE OF RptrGroupEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Table of descriptive and status information about
the groups of ports."
::= { rptrGroupInfo 1 }

rptrGroupEntry OBJECT-TYPE
SYNTAX RptrGroupEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"An entry in the table, containing information
about a single group of ports."
INDEX { rptrGroupIndex }
::= { rptrGroupTable 1 }

RptrGroupEntry ::= SEQUENCE {
  rptrGroupIndex
    Integer32,
  rptrGroupDescr
    DisplayString,
  rptrGroupObjectID
    OBJECT IDENTIFIER,
  rptrGroupOperStatus
    INTEGER,
  rptrGroupLastOperStatusChange
    TimeTicks,
  rptrGroupPortCapacity
    Integer32
}

rptrGroupIndex OBJECT-TYPE
SYNTAX Integer32 (1..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object identifies the group within the
system for which this entry contains
information.
REFERENCE
"[IEEE 802.3 Mgt], 30.4.2.1.1, aGroupID."
::= { rptrGroupEntry 1 }

rptrGroupDescr OBJECT-TYPE
SYNTAX      DisplayString (SIZE (0..255))
MAX-ACCESS  read-only
STATUS      deprecated
DESCRIPTION
"******** THIS OBJECT IS DEPRECATED **********
A textual description of the group. This value
should include the full name and version
identification of the group’s hardware type and
indicate how the group is differentiated from
other types of groups in the repeater. Plug-in
Module, Rev A’ or ‘Barney Rubble 10BASE-T 4-port
SIMM socket Version 2.1’ are examples of valid
group descriptions.

It is mandatory that this only contain printable
ASCII characters."
::= { rptrGroupEntry 2 }

rptrGroupObjectID OBJECT-TYPE
SYNTAX      OBJECT IDENTIFIER
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"The vendor’s authoritative identification of the
group. This value may be allocated within the SMI
enterprises subtree (1.3.6.1.4.1) and provides a
straight-forward and unambiguous means for
determining what kind of group is being managed.

For example, this object could take the value
1.3.6.1.4.1.4242.1.2.14 if vendor ‘Flintstones,
Inc.’ was assigned the subtree 1.3.6.1.4.1.4242,
and had assigned the identifier
1.3.6.1.4.1.4242.1.2.14 to its ‘Wilma Flintstone
6-Port FOIRL Plug-in Module.’"
::= { rptrGroupEntry 3 }

rptrGroupOperStatus OBJECT-TYPE
SYNTAX      INTEGER ( other(1),
operational(2),
malfunctioning(3),
notPresent(4),
underTest(5),
resetInProgress(6)
}
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"An object that indicates the operational status of the group.

A status of notPresent(4) indicates that the group is temporarily or permanently physically and/or logically not a part of the repeater. It is an implementation-specific matter as to whether the agent effectively removes notPresent entries from the table.

A status of operational(2) indicates that the group is functioning, and a status of malfunctioning(3) indicates that the group is malfunctioning in some way."
::= { rptrGroupEntry 4 }

rptrGroupLastOperStatusChange OBJECT-TYPE
SYNTAX      TimeTicks
MAX-ACCESS  read-only
STATUS      deprecated
DESCRIPTION
"******** THIS OBJECT IS DEPRECATED ********

An object that contains the value of sysUpTime at the time when the last of the following occurred:
1) the agent cold- or warm-started;
2) the row for the group was created (such as when the group was added to the system); or
3) the value of rptrGroupOperStatus for the group changed.

A value of zero indicates that the group’s operational status has not changed since the agent last restarted."
::= { rptrGroupEntry 5 }

rptrGroupPortCapacity OBJECT-TYPE
SYNTAX      Integer32 (1..2147483647)
MAX-ACCESS  read-only
STATUS        current
DESCRIPTION   "The rptrGroupPortCapacity is the number of ports that can be contained within the group. Valid range is 1-2147483647. Within each group, the ports are uniquely numbered in the range from 1 to rptrGroupPortCapacity.

Some ports may not be present in the system, in which case the actual number of ports present will be less than the value of rptrGroupPortCapacity. The number of ports present in the group will never be greater than the value of rptrGroupPortCapacity.

Note: In practice, this will generally be the number of ports on a module, card, or board, and the port numbers will correspond to numbers marked on the physical embodiment."
REFERENCE     "IEEE 802.3 Mgt, 30.4.2.1.2, aGroupPortCapacity."
::= { rptrGroupEntry 6 }

-- Basic information at the port level.
--
-- Configuration and status objects for
-- each managed repeater port in the system,
-- independent of whether there is one or more
-- managed repeater-units in the system.

rptrPortTable OBJECT-TYPE
SYNTAX        SEQUENCE OF RptrPortEntry
MAX-ACCESS    not-accessible
STATUS        current
DESCRIPTION   "Table of descriptive and status information about the repeater ports in the system. The number of entries is independent of the number of repeaters in the managed system."
::= { rptrPortInfo 1 }

rptrPortEntry OBJECT-TYPE
SYNTAX        RptrPortEntry
MAX-ACCESS    not-accessible
STATUS        current
DESCRIPTION   "An entry in the table, containing information about a single port."
INDEX { rptrPortGroupIndex, rptrPortIndex } ::= { rptrPortTable 1 }

RptrPortEntry ::= SEQUENCE {
    rptrPortGroupIndex Integer32,
    rptrPortIndex Integer32,
    rptrPortAdminStatus INTEGER,
    rptrPortAutoPartitionState INTEGER,
    rptrPortOperStatus INTEGER,
    rptrPortRptrId Integer32
}

rptrPortGroupIndex OBJECT-TYPE
SYNTAX Integer32 (1..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION "This object identifies the group containing the port for which this entry contains information."
 ::= { rptrPortEntry 1 }

rptrPortIndex OBJECT-TYPE
SYNTAX Integer32 (1..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION "This object identifies the port within the group for which this entry contains information. This identifies the port independently from the repeater it may be attached to. The numbering scheme for ports is implementation specific; however, this value can never be greater than rptrGroupPortCapacity for the associated group."
REFERENCE 
"[IEEE 802.3 Mgt], 30.4.3.1.1, aPortID."
 ::= { rptrPortEntry 2 }

rptrPortAdminStatus OBJECT-TYPE
SYNTAX INTEGER {
    enabled(1),
    disabled(2)
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION
"Setting this object to disabled(2) disables the
port. A disabled port neither transmits nor
receives. Once disabled, a port must be
explicitly enabled to restore operation. A port
which is disabled when power is lost or when a
reset is exerted shall remain disabled when normal
operation resumes.

The admin status takes precedence over auto-
partition and functionally operates between the
auto-partition mechanism and the AUI/PMA.

Setting this object to enabled(1) enables the port
and exerts a BEGIN on the port’s auto-partition
state machine.

(In effect, when a port is disabled, the value of
rptrPortAutoPartitionState for that port is frozen
until the port is next enabled. When the port
becomes enabled, the rptrPortAutoPartitionState
becomes notAutoPartitioned(1), regardless of its
pre-disabling state.)"

REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.2, aPortAdminState
and 30.4.3.2.1, acPortAdminControl."
::= { rptrPortEntry 3 }

rptrPortAutoPartitionState OBJECT-TYPE
SYNTAX      INTEGER {
             notAutoPartitioned(1),
             autoPartitioned(2)
             }
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"The autoPartitionState flag indicates whether the
port is currently partitioned by the repeater’s
auto-partition protection.

The conditions that cause port partitioning are
specified in partition state machine in Sections
9 and 27 of [IEEE 802.3 Std]. They are not
differentiated here."

REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.3, aAutoPartitionState."
::= { rptrPortEntry 4 }

rptrPortOperStatus  OBJECT-TYPE
SYNTAX       INTEGER {
    operational(1),
    notOperational(2),
    notPresent(3)
}
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"This object indicates the port’s operational status. The notPresent(3) status indicates the port is physically removed (note this may or may not be possible depending on the type of port.) The operational(1) status indicates that the port is enabled (see rptrPortAdminStatus) and working, even though it might be auto-partitioned (see rptrPortAutoPartitionState).

If this object has the value operational(1) and rptrPortAdminStatus is set to disabled(2), it is expected that this object’s value will soon change to notOperational(2)."
::= { rptrPortEntry 5 }

rptrPortRptrId OBJECT-TYPE
SYNTAX       Integer32 (0..2147483647)
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"This object identifies the repeater to which this port belongs. The repeater identified by a particular value of this object is the same as that identified by the same value of rptrInfoId. A value of zero indicates that this port currently is not a member of any repeater." 
::= { rptrPortEntry 6 }

-- New version of basic information at the repeater level.
--
-- Configuration, status, and control objects for each managed repeater in the system.

rptrInfoTable OBJECT-TYPE
SYNTAX SEQUENCE OF RptrInfoEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"A table of information about each
non-trivial repeater. The number of entries
depends on the physical configuration of the
managed system."
 ::= { rptrAllRptrInfo 1 }

rptrInfoEntry OBJECT-TYPE
SYNTAX RptrInfoEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"An entry in the table, containing information
about a single non-trivial repeater."
INDEX { rptrInfoId }
 ::= { rptrInfoTable 1 }

RptrInfoEntry ::= SEQUENCE {
    rptrInfoId
        Integer32,
    rptrInfoRptrType
        INTEGER,
    rptrInfoOperStatus
        INTEGER,
    rptrInfoReset
        INTEGER,
    rptrInfoPartitionedPorts
        Gauge32,
    rptrInfoLastChange
        TimeStamp
}

rptrInfoId OBJECT-TYPE
SYNTAX Integer32(1..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object identifies the repeater for which
this entry contains information."
 ::= { rptrInfoEntry 1 }

rptrInfoRptrType OBJECT-TYPE
SYNTAX INTEGER {
    other(1), -- undefined or unknown
tenMb(2),
onehundredMbClassI(3),
onehundredMbClassII(4)
}
MAX-ACCESS read-only
STATUS current
DESCRIPTION "The rptrInfoRptrType returns a value that identifies the CSMA/CD repeater type."
REFERENCE "[IEEE 802.3 Mgt], 30.4.1.1.2, aRepeaterType."
::= { rptrInfoEntry 2 }

rptrInfoOperStatus OBJECT-TYPE
SYNTAX INTEGER {
  other(1),
  ok(2),
  failure(3)
}
MAX-ACCESS read-only
STATUS current
DESCRIPTION "The rptrInfoOperStatus object indicates the operational state of the repeater."
REFERENCE "[IEEE 802.3 Mgt], 30.4.1.1.5, aRepeaterHealthState."
::= { rptrInfoEntry 3 }

rptrInfoReset OBJECT-TYPE
SYNTAX INTEGER {
  noReset(1),
  reset(2)
}
MAX-ACCESS read-write
STATUS current
DESCRIPTION "Setting this object to reset(2) causes a transition to the START state of Fig 9-2 in section 9 [IEEE 802.3 Std] for a 10Mb/s repeater, and to the START state of Fig 27-2 in section 27 of that standard for a 100Mb/s repeater.

Setting this object to noReset(1) has no effect. The agent will always return the value noReset(1) when this object is read.

After receiving a request to set this variable to reset(2), the agent is allowed to delay the reset
for a short period. For example, the implementor may choose to delay the reset long enough to allow the SNMP response to be transmitted. In any event, the SNMP response must be transmitted.

This action does not reset the management counters defined in this document nor does it affect the portAdminStatus parameters. Included in this action is the execution of a disruptive Self-Test with the following characteristics: a) The nature of the tests is not specified. b) The test resets the repeater but without affecting management information about the repeater. c) The test does not inject packets onto any segment. d) Packets received during the test may or may not be transferred. e) The test does not interfere with management functions.

After performing this self-test, the agent will update the repeater health information (including rptrInfoOperStatus), and send a rptrInfoResetEvent notification.

REFERENCE
"[IEEE 802.3 Mgt], 30.4.1.2.1, acResetRepeater."
::= { rptrInfoEntry 4 }

rptrInfoPartitionedPorts OBJECT-TYPE
SYNTAX        Gauge32
MAX-ACCESS    read-only
STATUS        current
DESCRIPTION    "This object returns the total number of ports in the repeater whose current state meets all three of the following criteria: rptrPortOperStatus does not have the value notPresent(3), rptrPortAdminStatus is enabled(1), and rptrPortAutoPartitionState is autoPartitioned(2)."
 ::= { rptrInfoEntry 5 }

rptrInfoLastChange OBJECT-TYPE
SYNTAX        TimeStamp
MAX-ACCESS    read-only
STATUS        current
DESCRIPTION    "The value of sysUpTime when any of the following conditions occurred:
1) agent cold- or warm-started;
2) this instance of repeater was created
(such as when a device or module was
added to the system);
3) a change in the value of rptrInfoOperStatus;
4) ports were added or removed as members of
the repeater; or
5) any of the counters associated with this
repeater had a discontinuity."

::= { rptrInfoEntry 6 }

--
-- Old version of statistics at the repeater level.
--
-- Performance monitoring statistics for the repeater
--
-- In a system containing a single managed repeater-unit,
-- the statistics object for the repeater-unit.

-- The objects contained under the rptrMonitorRptrInfo subtree are
-- intended for backwards compatibility with implementations of
-- RFC 1516 [11]. In newer implementations (both single- and
-- multiple-repeater implementations), the rptrMonitorTable will
-- be implemented. It is the preferred source of this information,
-- as it contains the values for all repeaters managed by the
-- agent. In all cases, the objects in the rptrMonitorRptrInfo
-- subtree are duplicates of the corresponding objects in the
-- first entry of the rptrMonitorTable.

rptrMonitorTransmitCollisions OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS deprecated
DESCRIPTION
"******** THIS OBJECT IS DEPRECATED **********

For a clause 9 (10Mb/s) repeater, this counter
is incremented every time the repeater state
machine enters the TRANSMIT COLLISION state
from any state other than ONE PORT LEFT
(Ref: Fig 9-2 [IEEE 802.3 Std]).

For a clause 27 repeater, this counter is
incremented every time the repeater core state
diagram enters the Jam state as a result of
Activity(ALL) > 1 (fig 27-2 [IEEE 802.3 Std]).
The approximate minimum time for rollover of this counter is 16 hours in a 10Mb/s repeater and 1.6 hours in a 100Mb/s repeater.

REFERENCE
"[IEEE 802.3 Mgt], 30.4.1.1.8, aTransmitCollisions."
::= { rptrMonitorRptrInfo 1 }

-- Statistics at the group level.
--
-- In a system containing a single managed repeater-unit,
-- the statistics objects for each group.

rptrMonitorGroupTable OBJECT-TYPE
SYNTAX SEQUENCE OF RptrMonitorGroupEntry
MAX-ACCESS not-accessible
STATUS deprecated
DESCRIPTION
"********* THIS OBJECT IS DEPRECATED ***********

Table of performance and error statistics for the groups within the repeater. The number of entries is the same as that in the rptrGroupTable."
::= { rptrMonitorGroupInfo 1 }

rptrMonitorGroupEntry OBJECT-TYPE
SYNTAX RptrMonitorGroupEntry
MAX-ACCESS not-accessible
STATUS deprecated
DESCRIPTION
"********* THIS OBJECT IS DEPRECATED ***********

An entry in the table, containing total performance and error statistics for a single group. Regular retrieval of the information in this table provides a means of tracking the performance and health of the networked devices attached to this group’s ports.

The counters in this table are redundant in the sense that they are the summations of information already available through other objects. However, these sums provide a considerable optimization of network management traffic over the otherwise necessary retrieval of the individual counters included in each sum.

Note: Group-level counters are
deprecated in this MIB. It is recommended that management applications instead use the repeater-level counters contained in the rptrMonTable.

INDEX { rptrMonitorGroupIndex }
::= { rptrMonitorGroupTable 1 }

RptrMonitorGroupEntry ::= SEQUENCE {
  rptrMonitorGroupIndex              Integer32,
  rptrMonitorGroupTotalFrames        Counter32,
  rptrMonitorGroupTotalOctets        Counter32,
  rptrMonitorGroupTotalErrors        Counter32
}

rptrMonitorGroupIndex OBJECT-TYPE
SYNTAX      Integer32 (1..2147483647)
MAX-ACCESS  read-only
STATUS      deprecated
DESCRIPTION "********** THIS OBJECT IS DEPRECATED **********
This object identifies the group within the repeater for which this entry contains information."
::= { rptrMonitorGroupEntry 1 }

rptrMonitorGroupTotalFrames OBJECT-TYPE
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      deprecated
DESCRIPTION "********** THIS OBJECT IS DEPRECATED **********
The total number of frames of valid frame length that have been received on the ports in this group and for which the FCSError and CollisionEvent signals were not asserted. This counter is the summation of the values of the rptrMonitorPortReadableFrames counters for all of the ports in the group.
This statistic provides one of the parameters necessary for obtaining the packet error rate.

de Graaf, et. al. Standards Track [Page 26]
The approximate minimum time for rollover of this counter is 80 hours in a 10Mb/s repeater.

::= { rptrMonitorGroupEntry 2 }

rptrMonitorGroupTotalOctets OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS deprecated
DESCRIPTION "********* THIS OBJECT IS DEPRECATED **********

The total number of octets contained in the valid frames that have been received on the ports in this group. This counter is the summation of the values of the rptrMonitorPortReadableOctets counters for all of the ports in the group.

This statistic provides an indicator of the total data transferred. The approximate minimum time for rollover of this counter is 58 minutes in a 10Mb/s repeater.

::= { rptrMonitorGroupEntry 3 }

rptrMonitorGroupTotalErrors OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS deprecated
DESCRIPTION "********* THIS OBJECT IS DEPRECATED **********

The total number of errors which have occurred on all of the ports in this group. This counter is the summation of the values of the rptrMonitorPortTotalErrors counters for all of the ports in the group.

::= { rptrMonitorGroupEntry 4 }

-- Statistics at the port level.
--

rptrMonitorPortTable OBJECT-TYPE
SYNTAX SEQUENCE OF RptrMonitorPortEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "Table of performance and error statistics for the ports. The number of entries is the same as that
in the rptrPortTable.

The columnar object rptrMonitorPortLastChange
is used to indicate possible discontinuities
of counter type columnar objects in the table.
::= { rptrMonitorPortInfo 1 }

rptrMonitorPortEntry OBJECT-TYPE
SYNTAX RptrMonitorPortEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "An entry in the table, containing performance and
error statistics for a single port."
INDEX { rptrMonitorPortGroupIndex, rptrMonitorPortIndex }
::= { rptrMonitorPortTable 1 }

RptrMonitorPortEntry ::= SEQUENCE {
  rptrMonitorPortGroupIndex Integer32,
  rptrMonitorPortIndex Integer32,
  rptrMonitorPortReadableFrames Counter32,
  rptrMonitorPortReadableOctets Counter32,
  rptrMonitorPortFCSErrors Counter32,
  rptrMonitorPortAlignmentErrors Counter32,
  rptrMonitorPortFrameTooLongs Counter32,
  rptrMonitorPortShortEvents Counter32,
  rptrMonitorPortRunts Counter32,
  rptrMonitorPortCollisions Counter32,
  rptrMonitorPortLateEvents Counter32,
  rptrMonitorPortVeryLongEvents Counter32,
  rptrMonitorPortDataRateMismatches Counter32,
  rptrMonitorPortAutoPartitions Counter32,
  rptrMonitorPortTotalErrors
}
Counter32,
rptrMonitorPortLastChange
TimeStamp
}

rptrMonitorPortGroupIndex OBJECT-TYPE
SYNTAX Integer32 (1..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION "This object identifies the group containing the port for which this entry contains information."
::= { rptrMonitorPortEntry 1 }

rptrMonitorPortIndex OBJECT-TYPE
SYNTAX Integer32 (1..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION "This object identifies the port within the group for which this entry contains information."
REFERENCE "[IEEE 802.3 Mgt], 30.4.3.1.1, aPortID."
::= { rptrMonitorPortEntry 2 }

rptrMonitorPortReadableFrames OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION "This object is the number of frames of valid frame length that have been received on this port. This counter is incremented by one for each frame received on this port whose OctetCount is greater than or equal to minFrameSize and less than or equal to maxFrameSize (Ref: IEEE 802.3 Std, 4.4.2.1) and for which the FCSError and CollisionEvent signals are not asserted.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes.

This statistic provides one of the parameters necessary for obtaining the packet error rate. The approximate minimum time for rollover of this counter is 80 hours at 10Mb/s."

REFERENCE
rptrMonitorPortReadableOctets OBJECT-TYPE
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
  "This object is the number of octets contained in
valid frames that have been received on this port.
This counter is incremented by OctetCount for each
frame received on this port which has been
determined to be a readable frame (i.e., including
FCS octets but excluding framing bits and dribble
bits).

A discontinuity may occur in the value
when the value of object
rptrMonitorPortLastChange changes.

This statistic provides an indicator of the total
data transferred. The approximate minimum time
for rollover of this counter in a 10Mb/s repeater
is 58 minutes.

For ports receiving traffic at a maximum rate in
a 100Mb/s repeater, this counter can roll over
in less than 6 minutes. Since that amount of time
could be less than a management station’s poll cycle
time, in order to avoid a loss of information a
management station is advised to also poll the
rptrMonitorPortUpper32Octets object, or to use the
64-bit counter defined by
rptrMonitorPortHCReadableOctets instead of the
two 32-bit counters."
REFERENCE
  "[IEEE 802.3 Mgt], 30.4.3.1.5, aReadableOctets."
 ::= { rptrMonitorPortEntry 4 }

rptrMonitorPortFCSErrors OBJECT-TYPE
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
  "This counter is incremented by one for each frame
received on this port with the FCSError signal
asserted and the FramingError and CollisionEvent
signals deasserted and whose OctetCount is greater
than or equal to minFrameSize and less than or equal to maxFrameSize (Ref: 4.4.2.1, IEEE 802.3 Std).

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes.

The approximate minimum time for rollover of this counter is 80 hours at 10Mb/s."

REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.6, aFrameCheckSequenceErrors."
::= { rptrMonitorPortEntry 5 }

rptrMonitorPortAlignmentErrors OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This counter is incremented by one for each frame received on this port with the FCSError and FramingError signals asserted and CollisionEvent signal deasserted and whose OctetCount is greater than or equal to minFrameSize and less than or equal to maxFrameSize (Ref: IEEE 802.3 Std, 4.4.2.1). If rptrMonitorPortAlignmentErrors is incremented then the rptrMonitorPortFCSErrors Counter shall not be incremented for the same frame.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes.

The approximate minimum time for rollover of this counter is 80 hours at 10Mb/s."

REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.7, aAlignmentErrors."
::= { rptrMonitorPortEntry 6 }

rptrMonitorPortFrameTooLongs OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This counter is incremented by one for each frame received on this port whose OctetCount is greater..."
than maxFrameSize (Ref: 4.4.2.1, IEEE 802.3 Std). If rptrMonitorPortFrameTooLongs is incremented then neither the rptrMonitorPortAlignmentErrors nor the rptrMonitorPortFCSErrors counter shall be incremented for the frame.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes.

The approximate minimum time for rollover of this counter is 61 days in a 10Mb/s repeater.

REFERENCE

"[IEEE 802.3 Mgt], 30.4.3.1.8, aFramesTooLong."

::= { rptrMonitorPortEntry 7 }

rptrMonitorPortShortEvents OBJECT-TYPE
  SYNTAX      Counter32
  MAX-ACCESS  read-only
  STATUS      current
  DESCRIPTION
    "This counter is incremented by one for each CarrierEvent on this port with ActivityDuration less than ShortEventMaxTime. ShortEventMaxTime is greater than 74 bit times and less than 82 bit times. ShortEventMaxTime has tolerances included to provide for circuit losses between a conformance test point at the AUI and the measurement point within the state machine.

Notes:

ShortEvents may indicate externally generated noise hits which will cause the repeater to transmit Runts to its other ports, or propagate a collision (which may be late) back to the transmitting DTE and damaged frames to the rest of the network.

Implementors may wish to consider selecting the ShortEventMaxTime towards the lower end of the allowed tolerance range to accommodate bit losses suffered through physical channel devices not budgeted for within this standard.

The significance of this attribute is different in 10 and 100 Mb/s collision domains. Clause 9 repeaters perform fragment extension of short
events which would be counted as runts on the interconnect ports of other repeaters. Clause 27 repeaters do not perform fragment extension.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes.

The approximate minimum time for rollover of this counter is 16 hours in a 10Mb/s repeater.

REFERENCE

"[IEEE 802.3 Mgt], 30.4.3.1.9, aShortEvents."

::= { rptrMonitorPortEntry 8 }

rptrMonitorPortRuns OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION "This counter is incremented by one for each CarrierEvent on this port that meets one of the following two conditions. Only one test need be made. a) The ActivityDuration is greater than ShortEventMaxTime and less than ValidPacketMinTime and the CollisionEvent signal is deasserted. b) The OctetCount is less than 64, the ActivityDuration is greater than ShortEventMaxTime and the CollisionEvent signal is deasserted. ValidPacketMinTime is greater than or equal to 552 bit times and less than 565 bit times. An event whose length is greater than 74 bit times but less than 82 bit times shall increment either the shortEvents counter or the runts counter but not both. A CarrierEvent greater than or equal to 552 bit times but less than 565 bit times may or may not be counted as a runt. ValidPacketMinTime has tolerances included to provide for circuit losses between a conformance test point at the AUI and the measurement point within the state machine. Runts usually indicate collision fragments, a normal network event. In certain situations associated with large diameter networks a percentage of collision fragments may exceed ValidPacketMinTime."
A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes.

The approximate minimum time for rollover of this counter is 16 hours in a 10Mb/s repeater.

REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.10, aRunts."
::= { rptrMonitorPortEntry 9 }

rptrMonitorPortCollisions OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"For a clause 9 repeater, this counter is incremented by one for any CarrierEvent signal on any port for which the CollisionEvent signal on this port is asserted. For a clause 27 repeater port the counter increments on entering the Collision Count Increment state of the partition state diagram (fig 27-8 of [IEEE 802.3 Std]).

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes.

The approximate minimum time for rollover of this counter is 16 hours in a 10Mb/s repeater.

REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.11, aCollisions."
::= { rptrMonitorPortEntry 10 }

rptrMonitorPortLateEvents OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"For a clause 9 repeater port, this counter is incremented by one for each CarrierEvent on this port in which the CollIn(X) variable transitions to the value SQE (Ref: 9.6.6.2, IEEE 802.3 Std) while the ActivityDuration is greater than the LateEventThreshold. For a clause 27 repeater port, this counter is incremented by one on entering the Collision Count Increment state.
of the partition state diagram (fig 27-8) while the ActivityDuration is greater than the LateEvent- Threshold. Such a CarrierEvent is counted twice, as both a collision and as a lateEvent.

The LateEventThreshold is greater than 480 bit times and less than 565 bit times. LateEventThreshold has tolerances included to permit an implementation to build a single threshold to serve as both the LateEventThreshold and ValidPacketMinTime threshold.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes.

The approximate minimum time for rollover of this counter is 81 hours in a 10Mb/s repeater.

REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.12, aLateEvents."

::= { rptrMonitorPortEntry 11 }

rptrMonitorPortVeryLongEvents OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"For a clause 9 repeater port, this counter is incremented by one for each CarrierEvent whose ActivityDuration is greater than the MAU Jabber Lockup Protection timer TW3 (Ref: 9.6.1 & 9.6.5, IEEE 802.3 Std).

For a clause 27 repeater port, this counter is incremented by one on entry to the Rx Jabber state of the receiver timer state diagram (fig 27-7). Other counters may be incremented as appropriate.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes."

REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.13, aVeryLongEvents."

::= { rptrMonitorPortEntry 12 }

rptrMonitorPortDataRateMismatches OBJECT-TYPE
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"This counter is incremented by one for each frame received by this port that meets all of the conditions required by only one of the following two measurement methods:

Measurement method A:  1) The CollisionEvent signal is not asserted (10Mb/s operation) or the Collision Count Increment state of the partition state diagram (fig 27-8 of [IEEE 802.3 Std]) has not been entered (100Mb/s operation).  2) The ActivityDuration is greater than ValidPacketMinTime.  3) The frequency (data rate) is detectably mismatched from the local transmit frequency.

Measurement method B:  1) The CollisionEvent signal is not asserted (10Mb/s operation) or the Collision Count Increment state of the partition state diagram (fig 27-8 of [IEEE 802.3 Std]) has not been entered (100Mb/s operation).  2) The OctetCount is greater than 63.  3) The frequency (data rate) is detectably mismatched from the local transmit frequency. The exact degree of mismatch is vendor specific and is to be defined by the vendor for conformance testing.

When this event occurs, other counters whose increment conditions were satisfied may or may not also be incremented, at the implementor’s discretion. Whether or not the repeater was able to maintain data integrity is beyond the scope of this standard.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes."

REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.14, aDataRateMismatches."
::= { rptrMonitorPortEntry 13 }

rptrMonitorPortAutoPartitions OBJECT-TYPE
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION "This counter is incremented by one for each time the repeater has automatically partitioned this port.

The conditions that cause a clause 9 repeater port to partition are specified in the partition state diagram in clause 9 of [IEEE 802.3 Std]. They are not differentiated here. A clause 27 repeater port partitions on entry to the Partition Wait state of the partition state diagram (fig 27-8 in [IEEE 802.3 Std]).

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes."
REFERENCE  
"[IEEE 802.3 Mgt], 30.4.3.1.15, aAutoPartitions."
::= { rptrMonitorPortEntry 14 }
rptrMonitorPortTotalErrors OBJECT-TYPE
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION "The total number of errors which have occurred on this port. This counter is the summation of the values of other error counters (for the same port), namely:

    rptrMonitorPortFCSErrors,
    rptrMonitorPortAlignmentErrors,
    rptrMonitorPortFrameTooLongs,
    rptrMonitorPortShortEvents,
    rptrMonitorPortLateEvents,
    rptrMonitorPortVeryLongEvents,
    rptrMonitorPortDataRateMismatches, and
    rptrMonitorPortSymbolErrors.

This counter is redundant in the sense that it is the summation of information already available through other objects. However, it is included specifically because the regular retrieval of this object as a means of tracking the health of a port provides a considerable optimization of network management traffic over the otherwise necessary
retrieval of the summed counters.

Note that rptrMonitorPortRunts is not included in this total; this is because runts usually indicate collision fragments, a normal network event.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes."

::= { rptrMonitorPortEntry 15 }

rptrMonitorPortLastChange OBJECT-TYPE
SYNTAX      TimeStamp
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"The value of sysUpTime when the last of the following occurred:
1) the agent cold- or warm-started;
2) the row for the port was created (such as when a device or module was added to the system); or
3) any condition that would cause one of the counters for the row to experience a discontinuity."

::= { rptrMonitorPortEntry 16 }

rptrMonitor100PortTable OBJECT-TYPE
SYNTAX      SEQUENCE OF RptrMonitor100PortEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
"Table of additional performance and error statistics for 100Mb/s ports, above and beyond those parameters that apply to both 10 and 100Mbps ports. Entries exist only for ports attached to 100Mbps repeaters.

The columnar object rptrMonitorPortLastChange is used to indicate possible discontinuities of counter type columnar objects in this table."

::= { rptrMonitorPortInfo 2 }

rptrMonitor100PortEntry OBJECT-TYPE
SYNTAX      RptrMonitor100PortEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
"An entry in the table, containing performance and error statistics for a single 100Mb/s port."
INDEX
{ rptrMonitorPortGroupIndex, rptrMonitorPortIndex }
::= { rptrMonitor100PortTable 1 }

RptrMonitor100PortEntry ::= SEQUENCE {
  rptrMonitorPortIsolates
    Counter32,
  rptrMonitorPortSymbolErrors
    Counter32,
  rptrMonitorPortUpper32Octets
    Counter32,
  rptrMonitorPortHCReadableOctets
    Counter64
}

rptrMonitorPortIsolates OBJECT-TYPE
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"This counter is incremented by one each time that the repeater port automatically isolates as a consequence of false carrier events. The conditions which cause a port to automatically isolate are defined by the transition from the False Carrier state to the Link Unstable state of the carrier integrity state diagram (figure 27-9) [IEEE 802.3 Standard].

Note: Isolates do not affect the value of the PortOperStatus object.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes."
REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.16, aIsolates."
::= { rptrMonitor100PortEntry 1 }

rptrMonitorPortSymbolErrors OBJECT-TYPE
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"This counter is incremented by one each time when
valid length packet was received at the port and there was at least one occurrence of an invalid data symbol. This can increment only once per valid carrier event. A collision presence at any port of the repeater containing port N, will not cause this attribute to increment.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes.

The approximate minimum time for rollover of this counter is 7.4 hours at 100Mb/s.

REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.17, aSymbolErrorDuringPacket."
::= { rptrMonitor100PortEntry 2 }

rptrMonitorPortUpper32Octets OBJECT-TYPE
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"This object is the number of octets contained in valid frames that have been received on this port, modulo 2**32. That is, it contains the upper 32 bits of a 64-bit octets counter, of which the lower 32 bits are contained in the rptrMonitorPortReadableOctets object.

This two-counter mechanism is provided for those network management protocols that do not support 64-bit counters (e.g. SNMP V1) and are used to manage a repeater type of 100Mb/s.

Conformance clauses for this MIB are defined such that implementation of this object is not required in a system which does not support 100Mb/s. However, systems with mixed 10 and 100Mb/s ports may implement this object across all ports, including 10Mb/s. If this object is implemented, it must be according to the definition in the first paragraph of this description; that is, the value of this object MUST be a valid count.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes."
rptrMonitorPortHCReadableOctets OBJECT-TYPE
SYNTAX Counter64
MAX-ACCESS read-only
STATUS current
DESCRIPTION "This object is the number of octets contained in valid frames that have been received on this port. This counter is incremented by OctetCount for each frame received on this port which has been determined to be a readable frame (i.e., including FCS octets but excluding framing bits and dribble bits).

This statistic provides an indicator of the total data transferred.

This counter is a 64-bit version of rptrMonitorPortReadableOctets. It should be used by network management protocols which support 64-bit counters (e.g. SNMPv2).

Conformance clauses for this MIB are defined such that implementation of this object is not required in a system which does not support 100Mb/s. However, systems with mixed 10 and 100Mb/s ports may implement this object across all ports, including 10Mb/s. If this object is implemented, it must be according to the definition in the first paragraph of this description; that is, the value of this object MUST be a valid count.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes."
REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.5, aReadableOctets."

::= { rptrMonitor100PortEntry 4 }

-- New version of statistics at the repeater level.
-- Statistics objects for each managed repeater
-- in the system.

rptrMonTable OBJECT-TYPE
SYNTAX SEQUENCE OF RptrMonEntry
MAX-ACCESS  not-accessible
CLASS    Monitoring
STATUS    current
DESCRIPTION
"A table of information about each
non-trivial repeater. The number of entries
in this table is the same as the number of
entries in the rptrInfoTable.

The columnar object rptrInfoLastChange is
used to indicate possible discontinuities of
counter type columnar objects in this table." ::= { rptrMonitorAllRptrInfo 1 }

rptrMonEntry OBJECT-TYPE
SYNTAX      RptrMonEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
"An entry in the table, containing information
about a single non-trivial repeater." INDEX    { rptrInfoId }
 ::= { rptrMonTable 1 }

RptrMonEntry ::= SEQUENCE {
    rptrMonTxCollisions     Counter32,
    rptrMonTotalFrames      Counter32,
    rptrMonTotalErrors      Counter32,
    rptrMonTotalOctets      Counter32
}

rptrMonTxCollisions OBJECT-TYPE
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"For a clause 9 (10Mb/s) repeater, this counter
is incremented every time the repeater state
machine enters the TRANSMIT COLLISION state
from any state other than ONE PORT LEFT
(Ref: Fig 9-2 [IEEE 802.3 Std]).

For a clause 27 repeater, this counter is
incremented every time the repeater core state..."
diagram enters the Jam state as a result of
Activity(ALL) > 1 (fig 27-2 [IEEE 802.3 Std]).

The approximate minimum time for rollover of this
counter is 16 hours in a 10Mb/s repeater and 1.6
hours in a 100Mb/s repeater."

REFERENCE
"[IEEE 802.3 Mgt], 30.4.1.1.8, aTransmitCollisions"

::= { rptrMonEntry 1 }

rptrMonTotalFrames OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The number of frames of valid frame length
that have been received on the ports in this repeater
and for which the FCSError and CollisionEvent
signals were not asserted. If an implementation
can not obtain a count of frames as seen by
the repeater itself, this counter may be
implemented as the summation of the values of the
rptrMonitorPortReadableFrames counters for all of
the ports in the repeater.

This statistic provides one of the parameters
necessary for obtaining the packet error rate.
The approximate minimum time for rollover of this
counter is 80 hours in a 10Mb/s repeater."

::= { rptrMonEntry 3 }

rptrMonTotalErrors OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The total number of errors which have occurred on
all of the ports in this repeater. The errors
included in this count are the same as those listed
for the rptrMonitorPortTotalErrors counter. If an
implementation can not obtain a count of these
errors as seen by the repeater itself, this counter
may be implemented as the summation of the values of the
rptrMonitorPortTotalErrors counters for all of
the ports in the repeater."

::= { rptrMonEntry 4 }

rptrMonTotalOctets OBJECT-TYPE
SYNTAX Counter32  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
"The total number of octets contained in the valid frames that have been received on the ports in this group. If an implementation can not obtain a count of octets as seen by the repeater itself, this counter may be the summation of the values of the rptrMonitorPortReadableOctets counters for all of the ports in the group.

This statistic provides an indicator of the total data transferred. The approximate minimum time for rollover of this counter in a 10Mb/s repeater is 58 minutes divided by the number of ports in the repeater.

For 100Mb/s repeaters processing traffic at a maximum rate, this counter can roll over in less than 6 minutes divided by the number of ports in the repeater. Since that amount of time could be less than a management station’s poll cycle time, in order to avoid a loss of information a management station is advised to also poll the rptrMonUpper32TotalOctets object, or to use the 64-bit counter defined by rptrMonHCTotalOctets instead of the two 32-bit counters."

::= { rptrMonEntry 5 }

rptrMon100Table OBJECT-TYPE  
SYNTAX SEQUENCE OF RptrMon100Entry  
MAX-ACCESS not-accessible  
STATUS current  
DESCRIPTION  
"A table of additional information about each 100Mb/s repeater, augmenting the entries in the rptrMonTable. Entries exist in this table only for 100Mb/s repeaters.

The columnar object rptrInfoLastChange is used to indicate possible discontinuities of counter type columnar objects in this table."

::= { rptrMonitorAllRptrInfo 2 }

rptrMon100Entry OBJECT-TYPE  
SYNTAX RptrMon100Entry  
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "An entry in the table, containing information about a single 100Mbps repeater."
INDEX { rptrInfoId }
::= { rptrMon100Table 1 }

RptrMon100Entry ::= SEQUENCE {
  rptrMonUpper32TotalOctets
    Counter32,
  rptrMonHCTotalOctets
    Counter64
}

rptrMonUpper32TotalOctets OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION "The total number of octets contained in the valid frames that have been received on the ports in this repeater, modulo 2**32. That is, it contains the upper 32 bits of a 64-bit counter, of which the lower 32 bits are contained in the rptrMonTotalOctets object. If an implementation can not obtain a count of octets as seen by the repeater itself, the 64-bit value may be the summation of the values of the rptrMonitorPortReadableOctets counters combined with the corresponding rptrMonitorPortUpper32Octets counters for all of the ports in the repeater.

This statistic provides an indicator of the total data transferred within the repeater.

This two-counter mechanism is provided for those network management protocols that do not support 64-bit counters (e.g. SNMP V1) and are used to manage a repeater type of 100Mb/s.

Conformance clauses for this MIB are defined such that implementation of this object is not required in a system which does not support 100Mb/s. However, systems with mixed 10 and 100Mb/s ports may implement this object across all ports, including 10Mb/s. If this object is implemented, it must be according to the definition in the first
rptrMonHCTotalOctets OBJECT-TYPE
SYNTAX Counter64
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The total number of octets contained in the valid frames that have been received on the ports in this group. If a implementation can not obtain a count of octets as seen by the repeater itself, this counter may be the summation of the values of the rptrMonitorPortReadableOctets counters for all of the ports in the group.

This statistic provides an indicator of the total data transferred.

This counter is a 64-bit (high-capacity) version of rptrMonUpper32Total10Octets and rptrMonTotalOctets. It should be used by network management protocols which support 64-bit counters (e.g. SNMPv2).

Conformance clauses for this MIB are defined such that implementation of this object is not required in a system which does not support 100Mb/s. However, systems with mixed 10 and 100Mb/s ports may implement this object across all ports, including 10Mb/s. If this object is implemented, it must be according to the definition in the first paragraph of this description; that is, the value of this object MUST be a valid count."

::= { rptrMon100Entry 2 }

--
-- The Repeater Address Search Table
--
-- This table provides an active address tracking capability which can be also used to collect the necessary information for mapping the topology of a network. Note that an NMS is required to have read-write access to the table in order to access this function. Section 4, "Topology Mapping", contains a description of an algorithm which can make use of this table, in combination with the
-- forwarding databases of managed bridges/switches
-- in the network, to map network topology.

rpbrAddrSearchTable OBJECT-TYPE
SYNTAX     SEQUENCE OF RpbrAddrSearchEntry
MAX-ACCESS not-accessible
STATUS     current
DESCRIPTION
 "This table contains one entry per repeater in the
system. It defines objects which allow a network
management application to instruct an agent to watch
for a given MAC address and report which port it
was seen on. Only one address search can be in
progress on each repeater at any one time. Before
starting an address search, a management application
should obtain ‘ownership’ of the entry in
rpbrAddrSearchTable for the repeater that is to
perform the search. This is accomplished with the
rpbrAddrSearchLock and rpbrAddrSearchStatus as
follows:

try_again:
    get(rpbrAddrSearchLock, rpbrAddrSearchStatus)
    while (rpbrAddrSearchStatus != notInUse)
    {
        /* Loop waiting for objects to be available*/
        short delay
        get(rpbrAddrSearchLock, rpbrAddrSearchStatus)
    }

    /* Try to claim map objects */
    lock_value = rpbrAddrSearchLock
    if ( set(rpbrAddrSearchLock = lock_value,
        rpbrAddrSearchStatus = inUse,
        rpbrAddrSearchOwner = 'my-IP-address)
        == FAILURE)
        /* Another manager got the lock */
        goto try_again

    /* I have the lock */
    set (rpbrAddrSearchAddress = <search target>)

    wait for rpbrAddrSearchState to change from none

    if (rpbrAddrSearchState == single)
        get (rpbrAddrSearchGroup, rpbrAddrSearchPort)
/* release the lock, making sure not to overwrite anyone else’s lock */
set (rptrAddrSearchLock = lock_value+1,
     rptrAddrSearchStatus = notInUse,
     rptrAddrSearchOwner = '')

A management station first retrieves the values of the appropriate instances of the rptrAddrSearchLock and rptrAddrSearchStatus objects, periodically repeating the retrieval if necessary, until the value of rptrAddrSearchStatus is ‘notInUse’. The management station then tries to set the same instance of the rptrAddrSearchLock object to the value it just retrieved, the same instance of the rptrAddrSearchStatus object to ‘inUse’, and the corresponding instance of rptrAddrSearchOwner to a value indicating itself. If the set operation succeeds, then the management station has obtained ownership of the rptrAddrSearchEntry, and the value of rptrAddrSearchLock is incremented by the agent (as per the semantics of TestAndIncr). Failure of the set operation indicates that some other manager has obtained ownership of the rptrAddrSearchEntry.

Once ownership is obtained, the management station can proceed with the search operation. Note that the agent will reset rptrAddrSearchStatus to ‘notInUse’ if it has been in the ‘inUse’ state for an abnormally long period of time, to prevent a misbehaving manager from permanently locking the entry. It is suggested that this timeout period be between one and five minutes.

When the management station has completed its search operation, it should free the entry by setting the instance of the rptrAddrSearchLock object to the previous value + 1, the instance of the rptrAddrSearchStatus to ‘notInUse’, and the instance of rptrAddrSearchOwner to a zero length string. This is done to prevent overwriting another station’s lock.

::= { rptrAddrTrackRptrInfo 1 }

rptrAddrSearchEntry OBJECT-TYPE
SYNTAX     RptrAddrSearchEntry
MAX-ACCESS not-accessible
STATUS     current
DESCRIPTION  

"An entry containing objects for invoking an address search on a repeater."

INDEX ( { rptrInfoId } ::= { rptrAddrSearchTable 1 } )

RptrAddrSearchEntry ::= SEQUENCE {
    rptrAddrSearchLock     TestAndIncr,
    rptrAddrSearchStatus   INTEGER,
    rptrAddrSearchAddress  MacAddress,
    rptrAddrSearchState    INTEGER,
    rptrAddrSearchGroup    Integer32,
    rptrAddrSearchPort     Integer32,
    rptrAddrSearchOwner    OwnerString
}

rptrAddrSearchLock OBJECT-TYPE
SYNTAX       TestAndIncr
MAX-ACCESS   read-write
STATUS       current
DESCRIPTION
    "This object is used by a management station as an advisory lock for this rptrAddrSearchEntry."
 ::= { rptrAddrSearchEntry 1 }

rptrAddrSearchStatus  OBJECT-TYPE
SYNTAX     INTEGER {
    notInUse(1),
    inUse(2)
}
MAX-ACCESS read-write
STATUS     current
DESCRIPTION
    "This object is used to indicate that some management station is currently using this rptrAddrSearchEntry. Cooperating managers should set this object to 'notInUse' when they are finished using this entry. The agent will automatically set the value of this object to 'notInUse' if it has been set to 'inUse' for an unusually long period of time."
 ::= { rptrAddrSearchEntry 2 }

rptrAddrSearchAddress OBJECT-TYPE
SYNTAX       MacAddress
MAX-ACCESS   read-write
STATUS       current
DESCRIPTION
"This object is used to search for a specified MAC address. When this object is set, an address search begins. This automatically sets the corresponding instance of the rptrAddrSearchState object to 'none' and the corresponding instances of the rptrAddrSearchGroup and rptrAddrSearchPort objects to 0.

When a valid frame is received by this repeater with a source MAC address which matches the current value of rptrAddrSearchAddress, the agent will update the corresponding instances of rptrAddrSearchState, rptrAddrSearchGroup and rptrAddrSearchPort to reflect the current status of the search, and the group and port on which the frame was seen."

::= { rptrAddrSearchEntry 3 }

rptrAddrSearchState OBJECT-TYPE
SYNTAX INTEGER {
  none(1),
  single(2),
  multiple(3)
}
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The current state of the MAC address search on this repeater. This object is initialized to 'none' when the corresponding instance of rptrAddrSearchAddress is set. If the agent detects the address on exactly one port, it will set this object to 'single', and set the corresponding instances of rptrAddrSearchGroup and rptrAddrSearchPort to reflect the group and port on which the address was heard. If the agent detects the address on more than one port, it will set this object to 'multiple'."

::= { rptrAddrSearchEntry 4 }

rptrAddrSearchGroup OBJECT-TYPE
SYNTAX Integer32 (0..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The group from which an error-free frame whose source address is equal to the corresponding instance of rptrAddrSearchAddress has been received. The value of this object is undefined when the corresponding instance of rptrAddrSearchState is
equal to ‘none’ or ‘multiple’.

::= { rptrAddrSearchEntry 5 }

rptrAddrSearchPort OBJECT-TYPE
SYNTAX Integer32 (0..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The port rom which an error-free frame whose
source address is equal to the corresponding instance
of rptrAddrSearchAddress has been received. The
value of this object is undefined when the
corresponding instance of rptrAddrSearchState is
equal to ‘none’ or ‘multiple’.

::= { rptrAddrSearchEntry 6 }

rptrAddrSearchOwner OBJECT-TYPE
SYNTAX OwnerString
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"The entity which currently has ‘ownership’ of this
rptrAddrSearchEntry."

::= { rptrAddrSearchEntry 7 }

--
-- The Port Address Tracking Table
--
-- This table provides a way for a network management
-- application to passively gather information (using
-- read-only privileges) about which network addresses
-- are connected to which ports of a repeater.
--

rptrAddrTrackTable OBJECT-TYPE
SYNTAX SEQUENCE OF RptrAddrTrackEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Table of address mapping information about the
ports."

::= { rptrAddrTrackPortInfo 1 }

rptrAddrTrackEntry OBJECT-TYPE
SYNTAX RptrAddrTrackEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"An entry in the table, containing address mapping information about a single port."

INDEX { rptrAddrTrackGroupIndex, rptrAddrTrackPortIndex } ::= { rptrAddrTrackTable 1 }

RptrAddrTrackEntry ::= SEQUENCE {
    rptrAddrTrackGroupIndex INTEGER,
    rptrAddrTrackPortIndex INTEGER,
    rptrAddrTrackLastSourceAddress -- DEPRECATED OBJECT MacAddress,
    rptrAddrTrackSourceAddrChanges Counter32,
    rptrAddrTrackNewLastSrcAddress OptMacAddr,
    rptrAddrTrackCapacity Integer32
}

rptrAddrTrackGroupIndex OBJECT-TYPE
SYNTAX INTEGER (1..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object identifies the group containing the port for which this entry contains information."
 ::= { rptrAddrTrackEntry 1 }

rptrAddrTrackPortIndex OBJECT-TYPE
SYNTAX INTEGER (1..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object identifies the port within the group for which this entry contains information."
REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.1, aPortID."
 ::= { rptrAddrTrackEntry 2 }

rptrAddrTrackLastSourceAddress OBJECT-TYPE
SYNTAX MacAddress
MAX-ACCESS read-only
STATUS deprecated
DESCRIPTION
"********* THIS OBJECT IS DEPRECATED **********
This object is the SourceAddress of the last readable frame (i.e., counted by rptrMonitorPortReadableFrames) received by this port.

This object has been deprecated because its value is undefined when no frames have been observed on this port. The replacement object is rptrAddrTrackNewLastSrcAddress.

REFERENCE

"[IEEE 802.3 Mgt], 30.4.3.1.18, aLastSourceAddress."

::= { rptrAddrTrackEntry 3 }

rptrAddrTrackSourceAddrChanges OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This counter is incremented by one for each time that the rptrAddrTrackLastSourceAddress attribute for this port has changed.

This may indicate whether a link is connected to a single DTE or another multi-user segment.

A discontinuity may occur in the value when the value of object rptrMonitorPortLastChange changes.

The approximate minimum time for rollover of this counter is 81 hours in a 10Mb/s repeater."

REFERENCE

"[IEEE 802.3 Mgt], 30.4.3.1.19, aSourceAddressChanges."

::= { rptrAddrTrackEntry 4 }

rptrAddrTrackNewLastSrcAddress OBJECT-TYPE
SYNTAX OptMacAddr
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object is the SourceAddress of the last readable frame (i.e., counted by rptrMonitorPortReadableFrames) received by this port. If no frames have been received by this port since the agent began monitoring the port activity, the agent shall return a string of length zero."

REFERENCE

"[IEEE 802.3 Mgt], 30.4.3.1.18, aLastSourceAddress."
::= { rptrAddrTrackEntry 5 }

rptrAddrTrackCapacity OBJECT-TYPE
SYNTAX      Integer32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION  "The maximum number of addresses that can be
detected on this port. This value indicates
the maximum number of entries in the
rptrExtAddrTrackTable relative to this port.

If this object has the value of 1, the agent
implements only the LastSourceAddress mechanism
described by RFC 1368 or RFC 1516."
::= { rptrAddrTrackEntry 6 }

-- Table for multiple addresses per port

rptrExtAddrTrackTable OBJECT-TYPE
SYNTAX      SEQUENCE OF RptrExtAddrTrackEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION  "A table to extend the address tracking table (i.e.,
rptrAddrTrackTable) with a list of source MAC
addresses that were recently received on each port.
The number of ports is the same as the number
of entries in table rptrPortTable. The number of
entries in this table depends on the agent/repeater
implementation and the number of different
addresses received on each port.

The first entry for each port contains
the same MAC address that is given by the
rptrAddrTrackNewLastSrcAddress for that port.

Entries in this table for a particular port are
retained when that port is switched from one
repeater to another.

The ordering of MAC addresses listed for a
particular port is implementation dependent."
::= { rptrAddrTrackPortInfo 2 }

rptrExtAddrTrackEntry OBJECT-TYPE
SYNTAX      RptrExtAddrTrackEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
"A row in the table of extended address tracking
information for ports. Entries can not be directly
created or deleted via SNMP operations."
INDEX       { rptrAddrTrackGroupIndex,
rptrAddrTrackPortIndex,
rptrExtAddrTrackMacIndex }
::= { rptrExtAddrTrackTable 1 }

RptrExtAddrTrackEntry ::= SEQUENCE {
  rptrExtAddrTrackMacIndex Integer32,
rptrExtAddrTrackSourceAddress MacAddress
}

rptrExtAddrTrackMacIndex OBJECT-TYPE
SYNTAX      Integer32 (1..2147483647)
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"The index of a source MAC address seen on
the port.

The ordering of MAC addresses listed for a
particular port is implementation dependent.

There is no implied relationship between a
particular index and a particular MAC
address. The index for a particular MAC
address may change without notice."
::= { rptrExtAddrTrackEntry 1 }

rptrExtAddrTrackSourceAddress OBJECT-TYPE
SYNTAX      MacAddress
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
"The source MAC address from a readable frame
(i.e., counted by rptrMonitorPortReadableFrames)
recently received by the port."
REFERENCE
"[IEEE 802.3 Mgt], 30.4.3.1.18, aLastSourceAddress."
::= { rptrExtAddrTrackEntry 2 }

-- The Repeater Top "N" Port Group
-- The Repeater Top N Port group is used to prepare reports that
-- describe a list of ports ordered by one of the statistics in the
-- Repeater Monitor Port Table. The statistic chosen by the
-- management station is sampled over a management
-- station-specified time interval, making the report rate based.
-- The management station also specifies the number of ports that
-- are reported.
--
-- The rptrTopNPortControlTable is used to initiate the generation
-- of a report. The management station may select the parameters
-- of such a report, such as which repeater, which statistic, how
-- many ports, and the start & stop times of the sampling. When
-- the report is prepared, entries are created in the
-- rptrTopNPortTable associated with the relevant
-- rptrTopNControlEntry. These entries are static for
-- each report after it has been prepared.
--
-- Note that counter discontinuities may appear in some
-- implementations if ports’ assignment to repeaters changes
-- during the collection of data for a Top "N" report.
-- A management application could read the corresponding
-- rptrMonitorPortLastChange timestamp in order to check
-- whether a discontinuity occurred.

rptrTopNPortControlTable OBJECT-TYPE
SYNTAX SEQUENCE OF RptrTopNPortControlEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "A table of control records for reports on the top ‘N’
ports for the rate of a selected counter. The number
of entries depends on the configuration of the agent.
The maximum number of entries is implementation
dependent."
::= { rptrTopNPortInfo 1 }

rptrTopNPortControlEntry OBJECT-TYPE
SYNTAX RptrTopNPortControlEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "A set of parameters that control the creation of a
report of the top N ports according to several metrics."
INDEX { rptrTopNPortControlIndex }
::= { rptrTopNPortControlTable 1 }

RptrTopNPortControlEntry ::= SEQUENCE {
rptrTopNPortControlIndex
  Integer32,
  rptrTopNPortRepeaterId
  Integer32,
  rptrTopNPortRateBase
  INTEGER,
  rptrTopNPortTimeRemaining
  Integer32,
  rptrTopNPortDuration
  Integer32,
  rptrTopNPortRequestedSize
  Integer32,
  rptrTopNPortGrantedSize
  Integer32,
  rptrTopNPortStartTime
  TimeStamp,
  rptrTopNPortOwner
  OwnerString,
  rptrTopNPortRowStatus
  RowStatus
}

rptrTopNPortControlIndex OBJECT-TYPE
  SYNTAX   Integer32 (1 .. 65535)
  MAX-ACCESS read-only
  STATUS   current
  DESCRIPTION
    "An index that uniquely identifies an entry in the
    rptrTopNPortControl table. Each such entry defines
    one top N report prepared for a repeater or system."
  ::= { rptrTopNPortControlEntry 1 }

rptrTopNPortRepeaterId OBJECT-TYPE
  SYNTAX   Integer32 (0..2147483647)
  MAX-ACCESS read-create
  STATUS   current
  DESCRIPTION
    "Identifies the repeater for which a top N report will
    be prepared (see rptrInfoId). If the value of this
    object is positive, only ports assigned to this repeater
    will be used to form the list in which to order the
    Top N table. If this value is zero, all ports will be
    eligible for inclusion on the list.

    The value of this object may not be modified if the
    associated rptrTopNPortRowStatus object is equal to
    active(1)."
If, for a particular row in this table, the repeater specified by the value of this object goes away (is removed from the rptrInfoTable) while the associated rptrTopNPortRowStatus object is equal to active(1), the row in this table is preserved by the agent but the value of rptrTopNPortRowStatus is changed to notInService(2), and the agent may time out the row if appropriate. If the specified repeater comes back (reappears in the rptrInfoTable) before the row has been timed out, the management station must set the value of the rptrTopNPortRowStatus object back to active(1) if desired (the agent doesn’t do this automatically).

::= { rptrTopNPortControlEntry 2 }

rptrTopNPortRateBase OBJECT-TYPE
SYNTAX INTEGER {
  readableFrames(1),
  readableOctets(2),
  fcsErrors(3),
  alignmentErrors(4),
  frameTooLongs(5),
  shortEvents(6),
  runts(7),
  collisions(8),
  lateEvents(9),
  veryLongEvents(10),
  dataRateMismatches(11),
  autoPartitions(12),
  totalErrors(13),
  isolates(14),
  symbolErrors(15)
}
MAX-ACCESS read-create
STATUS current
DESCRIPTION "The monitored variable, which the rptrTopNPortRate variable is based upon.

The value of this object may not be modified if the associated rptrTopNPortRowStatus object has a value of active(1)."
::= { rptrTopNPortControlEntry 3 }

rptrTopNPortTimeRemaining OBJECT-TYPE
SYNTAX Integer32 (0..2147483647)
MAX-ACCESS read-create
STATUS current
DESCRIPTION

"The number of seconds left in the report currently being collected. When this object is modified by the management station, a new collection is started, possibly aborting a currently running report. The new value is used as the requested duration of this report, which is loaded into the associated rptrTopNPortDuration object.

When this object is set to a non-zero value, any associated rptrTopNPortEntries shall be made inaccessible by the agent. While the value of this object is non-zero, it decrements by one per second until it reaches zero. During this time, all associated rptrTopNPortEntries shall remain inaccessible. At the time that this object decrements to zero, the report is made accessible in the rptrTopNPortTable. Thus, the rptrTopNPort table needs to be created only at the end of the collection interval.

If the value of this object is set to zero while the associated report is running, the running report is aborted and no associated rptrTopNPortEntries are created."

DEFVAL { 0 }
::= { rptrTopNPortControlEntry 4 }

rptrTopNPortDuration OBJECT-TYPE
SYNTAX      Integer32 (0..2147483647)
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION

"The number of seconds that this report has collected during the last sampling interval, or if this report is currently being collected, the number of seconds that this report is being collected during this sampling interval.

When the associated rptrTopNPortTimeRemaining object is set, this object shall be set by the agent to the same value and shall not be modified until the next time the rptrTopNPortTimeRemaining is set.

This value shall be zero if no reports have been requested for this rptrTopNPortControlEntry."
::= { rptrTopNPortControlEntry 5 }

rptrTopNPortRequestedSize OBJECT-TYPE
SYNTAX      Integer32
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
 "The maximum number of repeater ports requested
 for the Top N Table.

When this object is created or modified, the
agent should set rptrTopNPortGrantedSize as close
to this object as is possible for the particular
implementation and available resources."
DEFVAL { 10 }
::= { rptrTopNPortControlEntry 6 }

rptrTopNPortGrantedSize OBJECT-TYPE
SYNTAX      Integer32 (0..65535)
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
 "The maximum number of repeater ports in the
 top N table.

When the associated rptrTopNPortRequestedSize object is
created or modified, the agent should set this object as
closely to the requested value as is possible for the
particular implementation and available resources. The
agent must not lower this value except as a result of a
set to the associated rptrTopNPortRequestedSize object."
::= { rptrTopNPortControlEntry 7 }

rptrTopNPortStartTime OBJECT-TYPE
SYNTAX      TimeStamp
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
 "The value of sysUpTime when this top N report was
 last started. In other words, this is the time that
the associated rptrTopNPortTimeRemaining object was
modified to start the requested report.

If the report has not yet been started, the value
of this object is zero."
::= { rptrTopNPortControlEntry 8 }

rptrTopNPortOwner OBJECT-TYPE
SYNTAX      OwnerString
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION          "The entity that configured this entry and is
                     using the resources assigned to it."
::= { rptrTopNPortControlEntry 9 }

rptrTopNPortRowStatus OBJECT-TYPE
SYNTAX      RowStatus
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION          "The status of this row.
If the value of this object is not equal to
active(1), all associated entries in the
rptrTopNPortTable shall be deleted by the
agent."
::= { rptrTopNPortControlEntry 10 }

-- Top "N" reports

rptrTopNPortTable OBJECT-TYPE
SYNTAX      SEQUENCE OF RptrTopNPortEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION          "A table of reports for the top 'N' ports based on
setting of associated control table entries. The
maximum number of entries depends on the number
of entries in table rptrTopNPortControlTable and
the value of object rptrTopNPortGrantedSize for
each entry.

For each entry in the rptrTopNPortControlTable,
repeater ports with the highest value of
rptrTopNPortRate shall be placed in this table
in decreasing order of that rate until there is
no more room or until there are no more ports."
::= { rptrTopNPortInfo 2 }

rptrTopNPortEntry OBJECT-TYPE
SYNTAX      RptrTopNPortEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
"A set of statistics for a repeater port that is part of a top N report."

INDEX  
\{ rptrTopNPortControlIndex,  
  rptrTopNPortIndex \}  
::= \{ rptrTopNPortTable 1 \}

\[RptrTopNPortEntry ::= SEQUENCE \{
  rptrTopNPortIndex
  Integer32,
  rptrTopNPortGroupIndex
  Integer32,
  rptrTopNPortPortIndex
  Integer32,
  rptrTopNPortRate
  Gauge32
\} \]

\[rptrTopNPortIndex OBJECT-TYPE\]
SYNTAX Integer32 (1..65535)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"An index that uniquely identifies an entry in the rptrTopNPort table among those in the same report. This index is between 1 and N, where N is the number of entries in this report. Increasing values of rptrTopNPortIndex shall be assigned to entries with decreasing values of rptrTopNPortRate until index N is assigned to the entry with the lowest value of rptrTopNPortRate or there are no more rptrTopNPortEntries.

No ports are included in a report where their value of rptrTopNPortRate would be zero."
::= \{ rptrTopNPortEntry 1 \}

\[rptrTopNPortGroupIndex OBJECT-TYPE\]
SYNTAX Integer32 (1..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object identifies the group containing the port for this entry. (See also object type rptrGroupIndex.)"
::= \{ rptrTopNPortEntry 2 \}

\[rptrTopNPortPortIndex OBJECT-TYPE\]
SYNTAX Integer32 (1..2147483647)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The index of the repeater port.  
(See object type rptrPortIndex.)"
::= { rptrTopNPortEntry 3 }

rptrTopNPortRate OBJECT-TYPE
SYNTAX Gauge32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The amount of change in the selected variable 
during this sampling interval for the identified 
port.  The selected variable is that port’s 
instance of the object selected by 
rptrTopNPortRateBase."
::= { rptrTopNPortEntry 4 }

-- Notifications for use by Repeaters

rptrHealth NOTIFICATION-TYPE
OBJECTS { rptrOperStatus }
STATUS deprecated
DESCRIPTION
********** THIS OBJECT IS DEPRECATED **********

In a system containing a single managed repeater, 
the rptrHealth notification conveys information 
related to the operational status of the repeater. 
It is sent either when the value of 
rptrOperStatus changes, or upon completion of a 
non-disruptive test.

The rptrHealth notification must contain the 
rptrOperStatus object. The agent may optionally 
include the rptrHealthText object in the varBind 
list. See the rptrOperStatus and rptrHealthText 
objects for descriptions of the information that 
is sent.

The agent must throttle the generation of 
consecutive rptrHealth traps so that there is at 
least a five-second gap between traps of this 
type. When traps are throttled, they are dropped, 
not queued for sending at a future time. (Note
that 'generating' a trap means sending to all configured recipients.)"

REFERENCE
"[IEEE 802.3 Mgt], 30.4.1.3.1, nRepeaterHealth notification."
::= { snmpDot3RptrMgt 0 1 }

rptrGroupChange NOTIFICATION-TYPE
OBJECTS     { rptrGroupIndex }
STATUS      deprecated
DESCRIPTION
"********* THIS OBJECT IS DEPRECATED **********

In a system containing a single managed repeater, this notification is sent when a change occurs in the group structure of the repeater. This occurs only when a group is logically or physically removed from or added to a repeater. The varBind list contains the identifier of the group that was removed or added.

The agent must throttle the generation of consecutive rptrGroupChange traps for the same group so that there is at least a five-second gap between traps of this type. When traps are throttled, they are dropped, not queued for sending at a future time. (Note that 'generating' a trap means sending to all configured recipients.)"

REFERENCE
"[IEEE 802.3 Mgt], 30.4.1.3.3, nGroupMapChange notification."
::= { snmpDot3RptrMgt 0 2 }

rptrResetEvent NOTIFICATION-TYPE
OBJECTS     { rptrOperStatus }
STATUS      deprecated
DESCRIPTION
"********* THIS OBJECT IS DEPRECATED **********

In a system containing a single managed repeater-unit, the rptrResetEvent notification conveys information related to the operational status of the repeater. This trap is sent on completion of a repeater reset action. A repeater reset action is defined as an a transition to the START state of Fig 9-2 in section 9 [IEEE 802.3 Std], when triggered by a management command (e.g., an SNMP Set on the
The agent must throttle the generation of consecutive rptrResetEvent traps so that there is at least a five-second gap between traps of this type. When traps are throttled, they are dropped, not queued for sending at a future time. (Note that ‘generating’ a trap means sending to all configured recipients.)

The rptrResetEvent trap is not sent when the agent restarts and sends an SNMP coldStart or warmStart trap. However, it is recommended that a repeater agent send the rptrOperStatus object as an optional object with its coldStart and warmStart trap PDUs.

The rptrOperStatus object must be included in the varbind list sent with this trap. The agent may optionally include the rptrHealthText object as well."

REFERENCE
"[IEEE 802.3 Mgt], 30.4.1.3.2, nRepeaterReset notification."
::= { snmpDot3RptrMgt 0 3 }

-- Notifications for repeaters in a multiple-repeater implementation.
-- An implementation may send either the single-repeater OR
-- multiple-repeater version of these notifications (1 or 4; 2 or 5)
-- but not both.

rptrInfoHealth NOTIFICATION-TYPE
OBJECTS     { rptrInfoOperStatus }
STATUS      current
DESCRIPTION
"In a system containing multiple managed repeaters, the rptrInfoHealth notification conveys information related to the operational status of a repeater. It is sent either when the value of rptrInfoOperStatus changes, or upon completion of a non-disruptive test.

The agent must throttle the generation of consecutive rptrInfoHealth notifications for the same repeater so that there is at least a five-second gap between notifications of this type. When notifications are throttled, they are dropped, not queued for sending at a future time. (Note
that 'generating' a notification means sending to all configured recipients.)" 
REFERENCE
"[IEEE 802.3 Mgt], 30.4.1.3.1, nRepeaterHealth notification."
 ::= { snmpDot3RptrMgt 0 4 }

rptrInfoResetEvent NOTIFICATION-TYPE
OBJECTS { rptrInfoOperStatus }
STATUS current
DESCRIPTION
"In a system containing multiple managed repeaters, the rptrInfoResetEvent notification conveys information related to the operational status of a repeater. This notification is sent on completion of a repeater reset action. A repeater reset action is defined as a transition to the START state of Fig 9-2 in section 9 of [IEEE 802.3 Std], when triggered by a management command (e.g., an SNMP Set on the rptrInfoReset object).

The agent must throttle the generation of consecutive rptrInfoResetEvent notifications for a single repeater so that there is at least a five-second gap between notifications of this type. When notifications are throttled, they are dropped, not queued for sending at a future time. (Note that 'generating' a notification means sending to all configured recipients.)

The rptrInfoResetEvent is not sent when the agent restarts and sends an SNMP coldStart or warmStart trap. However, it is recommended that a repeater agent send the rptrInfoOperStatus object as an optional object with its coldStart and warmStart trap PDUs."
REFERENCE
"[IEEE 802.3 Mgt], 30.4.1.3.2, nRepeaterReset notification."
 ::= { snmpDot3RptrMgt 0 5 }

-- Conformance information

snmpRptrModConf
OBJECT IDENTIFIER ::= { snmpRptrMod 1 }
snmpRptrModCompls
   OBJECT IDENTIFIER ::= { snmpRptrModConf 1 }

snmpRptrModObjGrps
   OBJECT IDENTIFIER ::= { snmpRptrModConf 2 }

snmpRptrModNotGrps
   OBJECT IDENTIFIER ::= { snmpRptrModConf 3 }

-- Object groups

snmpRptrGrpBasic1516 OBJECT-GROUP
   OBJECTS     { rptrGroupCapacity,
                     rptrOperStatus,
                     rptrHealthText,
                     rptrReset,
                     rptrNonDisruptTest,
                     rptrTotalPartitionedPorts,
                     rptrGroupIndex,
                     rptrGroupDescr,
                     rptrGroupObjectID,
                     rptrGroupOperStatus,
                     rptrGroupLastOperStatusChange,
                     rptrGroupPortCapacity,
                     rptrPortGroupIndex,
                     rptrPortIndex,
                     rptrPortAdminStatus,
                     rptrPortAutoPartitionState,
                     rptrPortOperStatus }  
   STATUS      deprecated
   DESCRIPTION
"********* THIS GROUP IS DEPRECATED **********

Basic group from RFCs 1368 and 1516.

NOTE: this object group is DEPRECATED and replaced
with snmpRptrGrpBasic."
   ::= { snmpRptrModObjGrps 1 }

snmpRptrGrpMonitor1516 OBJECT-GROUP
   OBJECTS     { rptrMonitorTransmitCollisions,
                     rptrMonitorGroupIndex,
                     rptrMonitorGroupTotalFrames,
                     rptrMonitorGroupTotalOctets,
                     rptrMonitorGroupTotalErrors,}
rptrMonitorPortGroupIndex,
rptrMonitorPortIndex,
rptrMonitorPortReadableFrames,
rptrMonitorPortReadableOctets,
rptrMonitorPortFCSErrors,
rptrMonitorPortAlignmentErrors,
rptrMonitorPortFrameTooLongs,
rptrMonitorPortShortEvents,
rptrMonitorPortRunts,
rptrMonitorPortCollisions,
rptrMonitorPortLateEvents,
rptrMonitorPortVeryLongEvents,
rptrMonitorPortDataRateMismatches,
rptrMonitorPortAutoPartitions,
rptrMonitorPortTotalErrors }

STATUS deprecated
DESCRIPTION
"******** THIS GROUP IS DEPRECATED ********

Monitor group from RFCs 1368 and 1516.

NOTE: this object group is DEPRECATED and replaced
with snmpRptrGrpMonitor."

::= { snmpRptrModObjGrps 2 }

snmpRptrGrpAddrTrack1368 OBJECT-GROUP
OBJECTS
  { rptrAddrTrackGroupIndex, rptrAddrTrackPortIndex, rptrAddrTrackLastSourceAddress, rptrAddrTrackSourceAddrChanges }

STATUS obsolete
DESCRIPTION
"Address tracking group from RFC 1368.

NOTE: this object group is OBSOLETE and replaced
with snmpRptrGrpAddrTrack1516."

::= { snmpRptrModObjGrps 3 }

snmpRptrGrpAddrTrack1516 OBJECT-GROUP
OBJECTS
  { rptrAddrTrackGroupIndex, rptrAddrTrackPortIndex, rptrAddrTrackLastSourceAddress, rptrAddrTrackSourceAddrChanges, rptrAddrTrackNewLastSrcAddress }

STATUS deprecated
DESCRIPTION
"******** THIS GROUP IS DEPRECATED ********

---
de Graaf, et. al. Standards Track [Page 68]
Address tracking group from RFC 1516.

NOTE: this object group is DEPRECATED and replaced with snmpRptrGrpAddrTrack.

::= { snmpRptrModObjGrps 4 }

snmpRptrGrpBasic OBJECT-GROUP
OBJECTS { rptrGroupIndex,
           rptrGroupObjectID,
           rptrGroupOperStatus,
           rptrGroupPortCapacity,
           rptrPortGroupIndex,
           rptrPortIndex,
           rptrPortAdminStatus,
           rptrPortAutoPartitionState,
           rptrPortOperStatus,
           rptrPortRptrId,
           rptrInfoId,
           rptrInfoRptrType,
           rptrInfoOperStatus,
           rptrInfoReset,
           rptrInfoPartitionedPorts,
           rptrInfoLastChange }
STATUS    current
DESCRIPTION "Basic group for a system with one or more repeater-units in multi-segment (post-RFC 1516) version of the MIB module."
::= { snmpRptrModObjGrps 5 }

snmpRptrGrpMonitor OBJECT-GROUP
OBJECTS { rptrMonitorPortGroupIndex,
           rptrMonitorPortIndex,
           rptrMonitorPortReadableFrames,
           rptrMonitorPortReadableOctets,
           rptrMonitorPortFCSErrors,
           rptrMonitorPortAlignmentErrors,
           rptrMonitorPortFrameTooLongs,
           rptrMonitorPortShortEvents,
           rptrMonitorPortRuntts,
           rptrMonitorPortCollisions,
           rptrMonitorPortLateEvents,
           rptrMonitorPortVeryLongEvents,
           rptrMonitorPortDataRateMismatches,
           rptrMonitorPortAutoPartitions,
           rptrMonitorPortTotalErrors,
rptrMonitorPortLastChange,

rptrMonTxCollisions,
rptrMonTotalFrames,
rptrMonTotalErrors,
rptrMonTotalOctets }

STATUS current

DESCRIPTION "Monitor group for a system with one or more
repeater-units in multi-segment (post-RFC 1516)
version of the MIB module."

::= { snmpRptrModObjGrps 6 }

snmpRptrGrpMonitor100 OBJECT-GROUP

OBJECTS { rptrMonitorPortIsolates,
rptrMonitorPortSymbolErrors,
rptrMonitorPortUpper32Octets,

rptrMonUpper32TotalOctets }

STATUS current

DESCRIPTION "Monitor group for 100Mb/s ports and repeaters
in a system with one or more repeater-units in
multi-segment (post-RFC 1516) version of the MIB
module. Systems which support Counter64 should
also implement snmpRptrGrpMonitor100w64."

::= { snmpRptrModObjGrps 7 }

snmpRptrGrpMonitor100w64 OBJECT-GROUP

OBJECTS { rptrMonitorPortHCReadableOctets,
rptrMonHCTotalOctets }

STATUS current

DESCRIPTION "Monitor group for 100Mb/s ports and repeaters in a
system with one or more repeater-units and support
for Counter64."

::= { snmpRptrModObjGrps 8 }

snmpRptrGrpAddrTrack OBJECT-GROUP

OBJECTS { rptrAddrTrackGroupIndex,
rptrAddrTrackPortIndex,
rptrAddrTrackSourceAddrChanges,
rptrAddrTrackNewLastSrcAddress,
rptrAddrTrackCapacity }

STATUS current

DESCRIPTION "Passive address tracking group for post-RFC 1516
version of the MIB module."
::= { snmpRptrModObjGrps 9 }

snmpRptrGrpExtAddrTrack OBJECT-GROUP
OBJECTS { rptrxAddrTrackMacIndex,
            rptrxAddrTrackSourceAddress }
STATUS current
DESCRIPTION
"Extended passive address tracking group for
a system with one or more repeater-units in
post-RFC 1516 version of the MIB module."
::= { snmpRptrModObjGrps 10 }

snmpRptrGrpRptrAddrSearch OBJECT-GROUP
OBJECTS { rptrxAddrSearchLock,
            rptrxAddrSearchStatus,
            rptrxAddrSearchAddress,
            rptrxAddrSearchState,
            rptrxAddrSearchGroup,
            rptrxAddrSearchPort,
            rptrxAddrSearchOwner }
STATUS current
DESCRIPTION
"Active MAC address search group and topology
mapping support for repeaters."
::= { snmpRptrModObjGrps 11 }

snmpRptrGrpTopNPort OBJECT-GROUP
OBJECTS { rptrxTopNPortControlIndex,
           rptrxTopNPortRepeaterId,
           rptrxTopNPortRateBase,
           rptrxTopNPortTimeRemaining,
           rptrxTopNPortDuration,
           rptrxTopNPortRequestedSize,
           rptrxTopNPortGrantedSize,
           rptrxTopNPortStartTime,
           rptrxTopNPortOwner,
           rptrxTopNPortRowStatus,
           rptrxTopNPortIndex,
           rptrxTopNPortGroupId,
           rptrxTopNPortPorth }
STATUS current
DESCRIPTION
"Top 'N' group for repeater ports."
::= { snmpRptrModObjGrps 12 }

-- Compliances
snmpRptrModComplRFC1368 MODULE-COMPLIANCE
STATUS obsolete
DESCRIPTION
"Compliance for RFC 1368.

NOTE: this module compliance is OBSOLETE and replaced by snmpRptrModComplRFC1516."

MODULE -- this module
MANDATORY-GROUPS { snmpRptrGrpBasic1516 }

GROUP snmpRptrGrpMonitor1516
DESCRIPTION
"Implementation of this optional group is recommended for systems which have the instrumentation to do performance monitoring."

GROUP snmpRptrGrpAddrTrack1516
DESCRIPTION
"Implementation of this group is recommended for systems which have the necessary instrumentation."

::= { snmpRptrModCompl 1 }
::= { snmpRptrModCompls 2 }

snmpRptrModCompl MODULE-COMPLIANCE
STATUS current
DESCRIPTION
"Compliance for the multi-segment version of the MIB module for a system with one or more repeater-units."

MODULE -- this module
MANDATORY-GROUPS { snmpRptrGrpBasic,
    snmpRptrGrpMonitor,
    snmpRptrGrpAddrTrack }

GROUP snmpRptrGrpMonitor100
DESCRIPTION
"Implementation of this group is mandatory for managed systems which contain 100Mb/s repeaters."

GROUP snmpRptrGrpMonitor100w64
DESCRIPTION
"Implementation of this group is mandatory for managed systems which contain 100Mb/s repeaters and which can support Counter64."

GROUP snmpRptrGrpExtAddrTrack
DESCRIPTION
"Implementation of this group is recommended for systems which have the necessary instrumentation to track MAC addresses of multiple DTEs attached to a single repeater port."

GROUP snmpRptrGrpRptrAddrSearch
DESCRIPTION
"Implementation of this group is recommended for systems which allow read-write access and which have the necessary instrumentation to search all incoming data streams for a particular MAC address."

GROUP snmpRptrGrpTopNPort
DESCRIPTION
"Implementation of this group is recommended for systems which have
the necessary resources to support TopN statistics reporting."

::= { snmpRptrModCompls 3 }

END
4. Topology Mapping

The network mapping algorithm presented below takes information available from network devices such as repeaters, bridges, and switches, and creates a representation of the physical topology of the network.

Networking devices connect to the network via one or more ports. Through these ports, the device is capable of hearing network packets sent by other devices. By looking the source address in the packet, and identifying which port the packet was heard on, the device can provide information to a Network Management System about the location of an address in the network, relative to that device. For devices such as bridges and switches, the association of address to port can be retrieved via the forwarding data base part of the Bridge MIB. For repeaters, the rptrAddrSearchTable may be used to perform the association.

Given this information, it would be possible for the NMS to create a topology of the network which represents the physical relationships of the devices in the networks. The following is an example of how this might be done:

Assume the network:

```
    d1         d4         d7
   / \        /  \       /  \  
  d2  d3    d5           d6
```

The discovery process would first determine the existence of the network devices and nodes in the network. In the above example, the network devices discovered would be:

```
d1,d2,d3,d4,d5,d6,d7
```

From this list of discovered devices, select (arbitrarily or via some heuristic) a device as the starting point. From that device, determine where all other devices are located in the network with respect to the selected device.
For example, if d1 is the selected device, the network in relation to d1 would look like:

```
  / \ 
 d1 / | \   
    /  |  \  
   d2 d3 d4,d5,d6,d7
```

So d1 sees d2 on one port, d3 on another port, and d4, d5, and d6 on the third port. In other words, using the rptrAddrSearchTable (if d1 is a repeater) or the Forwarding Database (if it is a bridge or a switch), d1 has located d2 on one port, d1 has located d3 on another port, and finally, d1 has located d4, d5, d6, and d7 on yet another port.

After the first step of the algorithm is accomplished, the next and final step is a recursive one. Go to each of these temporary ‘segments’ (e.g., the segment connecting d1 and d2, or the segment connecting d1 and d3, or the segment connecting d1, d4, d5, d6, and d7) and determine which of these devices really belongs in that segment.

As new segments are created due to this process, the recursive algorithm visits them, and performs the exact same process.

In the example, the segments connecting d1 and d2, and connecting d1 and d3, require no further scrutiny, since there are only two nodes in those segments. However, the segment connecting d1, d4, d5, d6, and d7 may prove to be one or more segments, so we will investigate it.

The purpose of this step is to determine which devices are really connected to this segment, and which are actually connected downstream. This is done by giving each of the child devices in the segment (d4, d5, d6, and d7) a chance to eliminate each of the others from the segment.

A device eliminates another device by showing that it hears the parent device (in this case, d1) on one port, and the other device on another port (different from the port on which it heard the parent). If this is true, then it must mean that that device is _between_ the parent device and the device which is being eliminated.
In the example, we can see that device d4 can eliminate both d5 and d6, but nobody can eliminate d4 and d7, because everybody hears them on the same port that they hear the parent device (d1). So the resulting topology looks like:

```
   d1
  / | \     
 /  |  \    
 d2 d3 d4,d7
   |     
 d5,d6
```

Next the algorithm visits the next segment, which is the one connecting d4, d5, and d6. Using the process stated above, d5 can eliminate d6, since it hears d4 on a different port from where it hears d6. Finally, the topology looks like:

```
   d1
  / | \     
 /  |  \    
 d2 d3 d4,d7
   |     
 d5     
   |     
 d6
```

This is actually the topology shown at the beginning of the description.

With this information about how the network devices are connected, it is a relatively simple extension to then place nodes such as workstations and PCs in the network. This can be done by placing the node into a segment, then allowing the network devices to show that the node is really not part of that segment.

This elimination can be done because the devices know what port connects them to the segment on which the node is temporarily placed. If they actually hear the node on a different port than that which connects the device to the segment, then the node must be downstream, and so it is moved onto the downstream segment. Then that segment is evaluated, and so forth. Eventually, no device can show that the node is connected downstream, and so it must be attached to that segment.
For example, assume the network:

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>d1</td>
<td>d4</td>
<td>d7</td>
</tr>
</tbody>
</table>
```

```
/ \ |
/  \ |
| d2 d3 d5 |
```

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>e1</td>
<td>d1</td>
<td>d4</td>
</tr>
</tbody>
</table>
```

```
/ \ |
/  \ |
| d2 d3 d5 |
```

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>d1</td>
<td>d4</td>
<td>d7</td>
</tr>
</tbody>
</table>
```

```
/ \ |
/  \ |
| d2 d3, e1 d5 |
```

In this network, we are trying to place el where it belongs. We begin by placing it arbitrarily into a segment:

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>e1</td>
<td>d1</td>
<td>d4</td>
</tr>
</tbody>
</table>
```

```
/ \ |
/  \ |
| d2 d3 d5 |
```

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>d1</td>
<td>d4</td>
<td>d7</td>
</tr>
</tbody>
</table>
```

```
/ \ |
/  \ |
| d2 d3, e1 d5 |
```

In the above case, we would give d1, d4, and d7 a chance to show that el is not really on that segment. d4 and d7 hear el on the same port which connects them to that segment, so they cannot eliminate el from the segment. However, d1 will hear el on a different port, so we move el down onto the segment which is connected by that port. This yields the following:
Now we give everyone in that segment (besides that parent device, d1) a chance to eliminate e1. Only d3 can try, and it succeeds, so we place e1 on segment which is connected by the port on which d3 heard e1. There is no segment there (yet), so we create one, and end up with the following:

```
|            |       |
|            |       |
| d1         | d4    | d7  |
/ \         |
/  \        |
d2   d3     | d5    |
/            |
| e1         | d6    |
```

which is the correct position.

5. Acknowledgements

This document was produced by the IETF Hub MIB Working Group, whose efforts were greatly advanced by the contributions of the following people:

Chuck Black
John Flick
Jeff Johnson
Leon Leong
Mike Lui
Dave Perkins
Geoff Thompson
Maurice Turcotte
Paul Woodruff
6. References


7. Security Considerations

Security issues are not discussed in this memo.

8. Authors’ Addresses

Kathryn de Graaf
3Com Corporation
118 Turnpike Rd.
Southborough, MA 01772 USA

Phone: (508)229-1627
Fax: (508)490-5882
EMail: kdegraaf@isd.3com.com

Dan Romascanu
Madge Networks (Israel) Ltd.
Atidim Technology Park, Bldg. 3
Tel Aviv 61131, Israel

Phone: 972-3-6458414, 6458458
Fax: 972-3-6487146
EMail: dromasca@madge.com