Real-time Transport Protocol (RTP) MIB Version 2
draft-ietf-avt-mib-rtp-bis-01

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on 22 December 2006.

Copyright Notice

Copyright (C) The Internet Society (2006).

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 Real-Time Transport Protocol (RTP) systems (RFC3550) and is a proposed replacement for RFC 2959 - the RTP MIB.
1. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIv2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

2. Overview

An "RTP System" may be a host end-system that runs an application program that sends or receives RTP data packets, or it may be an intermediate-system that forwards RTP packets. RTP Control Protocol (RTCP) packets are sent by senders and receivers to convey information about RTP packet transmission and reception [RFC3550]. RTP monitors may collect RTCP information on senders and receivers to and from an RTP host or intermediate-system.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

2.1 Components

The RTP MIB is structured around "Session," "Receiver" and "Sender" conceptual abstractions.

2.1.1 An "RTP Session" is the "...association of participants communicating with RTP. For each participant, the session is defined by a particular pair of destination transport addresses (one network address plus a port pair for RTP and RTCP). The destination transport addresses may be common for all participants, as in the
case of IP multicast, or may be different for each, as in the case of individual unicast addresses plus a common port pair," as defined in section 3 of [RFC3550].

2.1.2 A "Sender" is identified within an RTP session by a 32-bit numeric "Synchronization Source," or "SSRC", value and is "...the source of a stream of RTP packets" as defined in section 3 of [RFC3550]. The sender is also a source of RTCP Sender Report packets as specified in section 6 of [RFC3550].

2.1.3 A "Receiver" of a "stream of RTP packets" can be a unicast or multicast Receiver as described in 2.1.1, above. An RTP Receiver has an SSRC value that is unique to the session. An RTP Receiver is a source of RTCP Receiver Reports as specified in section 6 of [RFC3550].

2.2 Applicability of the MIB to RTP System Implementations

The RTP MIB may be used in two types of RTP implementations, RTP Host Systems (end systems) and RTP Monitors, see section 3 of [RFC3550]. Use of the RTP MIB for RTP Translators and Mixers, as defined in section 7 of [RFC3550], is for further study.

2.2.1 RTP host Systems are end-systems that may use the RTP MIB to collect RTP session and stream data that the host is sending or receiving; these data may be used by a network manager to detect and diagnose faults that occur over the lifetime of an RTP session as in a "help-desk" scenario.

2.2.2 RTP Monitors of multicast RTP sessions may be third-party or may be located in the RTP host. RTP Monitors may use the RTP MIB to collect RTP session and stream statistical data; these data may be used by a network manager for capacity planning and other network-management purposes. An RTP Monitor may use the RTP MIB to collect data to permit a network manager to detect and diagnose faults in RTP sessions or to permit a network manger to configure its operation.

2.2.3 Many host systems will want to keep track of streams beyond what they are sending and receiving. In a host monitor system, a host agent would use RTP data from the host to maintain data about streams it is sending and receiving, and RTCP data to collect data about other hosts in the session. For example, an agent for an RTP host that is sending a stream would use data from its RTP system to maintain the rtpSenderTable, but it may want to maintain a rtpRcvrTable for endpoints that are receiving its stream. To do this the RTP agent will collect RTCP data from the receivers of its stream to build the rtpRcvrTable. A host monitor system MUST set the rtpSessionMonitor object to ‘true(1)’, but it does not have to accept management operations that create and destroy rows in its rtpSessionTable.

2.2.4 The RTCP XR MIB provides extended data related to the performance of Voice over IP streams. The RTP-MIBV2 and RTCP XR
MIBs have been designed to be used together to support the management
of Voice over IP systems.

2.3 The Structure of the RTP MIB

There are six tables in the RTP MIB. The rtpSessionTable contains
objects that describe active sessions at the host, or monitor. The
rtpSenderTable contains information about senders to the RTP session.
The rtpRcvrTable contains information about receivers of RTP session
data. The rtpSessionInverseTable, rtpSenderInverseTable, and
rtpRcvrInverseTable contain information to efficiently find indexes
into the rtpSessionTable, rtpSenderTable, and rtpRcvrTable,
respectively.

The reverse lookup tables (rtpSessionInverseTable,
rtpSenderInverseTable, and rtpRcvrInverseTable) are optional tables
to help management applications efficiently access conceptual rows in
other tables. Implementors of this MIB SHOULD implement these tables
for multicast RTP sessions when table indexes (rtpSessionIndex of
rtpSessionTable, rtpSenderSSRC of rtpSenderTable, and the SSRC pair
in the rtpRcvrTable) are not available from other MIBs. Otherwise,
the management application may be forced to perform expensive tree
walks through large numbers of sessions, senders, or receivers.

For any particular RTP session, the rtpSessionMonitor object
indicates whether remote senders or receivers to the RTP session are
to be monitored. If rtpSessionMonitor is true(1) then senders and
receivers to the session MUST be monitored with entries in the
rtpSenderTable and rtpRcvrTable. RTP sessions are monitored by the
RTP agent that updates rtpSenderTable and rtpRcvrTable objects with
information from RTCP reports from remote senders or remote receivers
respectively.

rtpSessionNewIndex is a global object that permits a network-
management application to obtain a unique index for conceptual row
creation in the rtpSessionTable. In this way the SNMP Set operation
MAY be used to configure a monitor.

3. Definitions

RTP-MIB DEFINITIONS ::= BEGIN
IMPORTS
    Counter32, Counter64, Gauge32, mib-2, Integer32,
    MODULE-IDENTITY,
    OBJECT-TYPE, Unsigned32 FROM SNMPv2-SMI
    InetAddressType, InetAddress, InetPortNumber FROM INET-ADDRESS-MIB
    RowStatus, TestAndIncr, TruthValue, DateAndTime FROM SNMPv2-TC
    OBJECT-GROUP, MODULE-COMPLIANCE FROM SNMPv2-CONF
    Utf8String FROM SYSAPPL-MIB
    InterfaceIndex FROM IF-MIB;

Clark Expires December 2006
rtpMIBV2 MODULE-IDENTITY
   LAST-UPDATED "200602260000Z" -- 26 February 2006
   ORGANIZATION
      "IETF AVT Working Group
         Email: avt@ietf.org"
   CONTACT-INFO
      "Alan Clark
         Telchemy
         3360 Martins Farm Rd
         Suwanee, GA 20024
         United States
         Email: alan@telchemy.com
      Amy Pendleton
         Nortel
         2380 Performance Drive
         Richardson, TX 75081
         Email: aspen@nortel.com"
   DESCRIPTION
      "The managed objects of RTP systems. The MIB is
      structured around three types of information.
      1. General information about RTP sessions such
         as the session address.
      2. Information about RTP streams being sent to
         an RTP session by a particular sender.
      3. Information about RTP streams received on an
         RTP session by a particular receiver from a
         particular sender.
      There are two types of RTP Systems, RTP hosts and
      RTP monitors. As described below, certain objects
      are unique to a particular type of RTP System. An
      RTP host may also function as an RTP monitor.
      Refer to RFC 3550, 'RTP: A Transport Protocol for
      Real-Time Applications,' section 3.0, for definitions."
   REVISION     "200602260000Z" -- 26 February 2006
   DESCRIPTION  "Version 2 of this MIB.
      Published as draft-ietf-avt-mib-rtp-bis-01"
::= { mib-2 nnn }

-- OBJECTS
--

rtpMIBV2Objects OBJECT IDENTIFIER ::= { rtpMIBV2 1 }
rtpConformance OBJECT IDENTIFIER ::= { rtpMIBV2 2 }

-- SESSION NEW INDEX
--

rtpSessionNewIndex OBJECT-TYPE
   SYNTAX          TestAndIncr
   MAX-ACCESS      read-write
This object is used to assign values to rtpSessionIndex as described in 'Textual Conventions for SMIPv2'. For an RTP system that supports the creation of rows, the network manager would read the object, and then write the value back in the Set that creates a new instance of rtpSessionEntry. If the Set fails with the code 'inconsistentValue,' then the process must be repeated; If the Set succeeds, then the object is incremented, and the new instance is created according to the manager’s directions. However, if the RTP agent is not acting as a monitor, only the RTP agent may create conceptual rows in the RTP session table.

::= { rtpMIBV2Objects 1 }

-- SESSION INVERSE TABLE
--
rtpSessionInverseTable OBJECT-TYPE
SYNTAX SEQUENCE OF RtpSessionInverseEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "Maps source and destination address to or more rtpSessionIndex values describing rows in the rtpSessionTable. This allows rows to be retrieved in the rtpSessionTable corresponding to a given session without having to walk the entire (potentially large) table."
::= { rtpMIBV2Objects 2 }

RtpSessionInverseEntry OBJECT-TYPE
SYNTAX RtpSessionInverseEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "Each entry corresponds to exactly one entry in the rtpSessionTable."
INDEX { rtpSessionSourceIPaddress, rtpSessionSourceRTPport, rtpSessionDestIPaddress, rtpSessionDestRTPport, rtpSessionCallState, rtpSessionIndex }
::= { rtpSessionInverseTable 1 }

RtpSessionInverseEntry ::= SEQUENCE {
    rtpSessionInverseStartTime     DateAndTime
}

rtpSessionInverseStartTime OBJECT-TYPE
SYNTAX DateAndTime
MAX-ACCESS read-only
STATUS current
DESCRIPTION "The local time at which this row was
There’s one entry in rtpSessionTable for each RTP session on which packets are being sent, received, and/or monitored.

```plaintext
::= { rtpMIBV2Objects 3 }
```

Data in rtpSessionTable uniquely identify an RTP session. A host RTP agent MUST create a read-only row for each session to which packets are being sent or received. Rows MUST be created by the RTP Agent at the start of a session when one or more senders or receivers are observed. An RTP session SHOULD be monitored to create management information on all RTP streams being sent or received when the rtpSessionMonitor has the TruthValue of ‘true(1)’. An RTP monitor SHOULD permit row creation with the side effect of causing the RTP System to join the multicast session for the purposes of gathering management information (additional conceptual rows are created in the rtpRcvrTable and rtpSenderTable). Thus, rtpSessionTable rows SHOULD be created for RTP session monitoring purposes. Rows created by a management application SHOULD be deleted via SNMP operations by management applications. Rows created by management operations are deleted by management operations by setting rtpSessionRowStatus to ‘destroy(6)’.

```plaintext
INDEX { rtpSessionCallState, rtpSessionIndex }
::= { rtpSessionTable 1 }
```

```plaintext
RtpSessionEntry ::= SEQUENCE {
    rtpSessionCallState              INTEGER,
    rtpSessionIndex                  Integer32,
    rtpSessionSessionIdentifier      OCTET STRING,
    rtpSessionStartTime              DateAndTime,
    rtpSessionStopTime               DateAndTime,
    rtpSessionSourceIPtype           InetAddressType,
    rtpSessionSourceIPaddress        InetAddress,
    rtpSessionSourceRTPport          InetPortNumber,
    rtpSessionSourceRTCPport         InetPortNumber,
    rtpSessionDestIPtype             InetAddressType,
    rtpSessionDestIPaddress          InetAddress,
    rtpSessionDestIPport             InetPortNumber
}
```
rtpSessionDestRTPport           InetPortNumber,
rtpSessionDestRTCPport          InetPortNumber,
rtpSessionSrcIdenType           INTEGER,
rtpSessionSrcIdentifier         OCTET STRING,
rtpSessionDestIdenType          INTEGER,
rtpSessionDestIdentifier        OCTET STRING,
rtpSessionIfIndex               InterfaceIndex,
rtpSessionMonitor               TruthValue,
rtpSessionSenderJoins           Counter32,
rtpSessionReceiverJoins         Counter32,
rtpSessionByes                  Counter32,
rtpSessionRowStatus             RowStatus,
rtpSessionMaxNumEntries         Integer32
}

rtpSessionCallState OBJECT-TYPE
SYNTAX INTEGER { active(1),
                 completed(2)
                 }
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Index for this session within the Session ID table. The value of this parameter shall be 2 if the session is complete or inactive and 1 if the session is still active."
::= { rtpSessionEntry 1 }

rtpSessionIndex OBJECT-TYPE
SYNTAX          Integer32 (1..2147483647)
MAX-ACCESS      not-accessible
STATUS          current
DESCRIPTION
"The index of the conceptual row which is for SNMP purposes only and has no relation to any protocol value. There is no requirement that these rows are created or maintained sequentially."
::= { rtpSessionEntry 2 }

rtpSessionSessionIdentifier OBJECT-TYPE
SYNTAX            OCTET STRING (SIZE(0..128))
MAX-ACCESS        read-only
STATUS            current
DESCRIPTION
"Unique identifier for this session. A billing record correlation identifier should be used if available, otherwise an identifier such as SSRC can be used."
::= { rtpSessionEntry 3 }

rtpSessionStartTime OBJECT-TYPE
SYNTAX            DateAndTime
MAX-ACCESS        read-only
STATUS            current
DESCRIPTION
"Call start time for this call. If the start time is not known then this represents the earliest known time associated with the call."
::= { rtpSessionEntry 4 }

rtpSessionStopTime OBJECT-TYPE
SYNTAX DateAndTime
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Call stop time for this call. If the call is still active then this shall have the value 0. If the call is complete but the time is unknown then this shall have the value of the latest time associated with the call."
::= { rtpSessionEntry 5 }

rtpSessionSourceIPtype OBJECT-TYPE
SYNTAX InetAddressType
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"IP address type for the originating IP endpoint for this RTP stream."
::= { rtpSessionEntry 6 }

rtpSessionSourceIPAddress OBJECT-TYPE
SYNTAX InetAddress
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"IP address for the originating IP endpoint for this RTP stream."
::= { rtpSessionEntry 7 }

rtpSessionSourceRTPport OBJECT-TYPE
SYNTAX InetPortNumber
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Source UDP port for RTP. A value of 0 indicates an unknown port number."
::= { rtpSessionEntry 8 }

Clark Expires December 2006 [Page 9]
rtpSessionSourceRTCPport OBJECT-TYPE
SYNTAX InetPortNumber
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Source UDP port for RTCP. A value of 0 indicates
an unknown port number."
 ::= { rtpSessionEntry 9 }

rtpSessionDestIPtype OBJECT-TYPE
SYNTAX InetAddressType
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Destination IP address type for this session."
 ::= { rtpSessionEntry 10 }

rtpSessionDestIPaddress OBJECT-TYPE
SYNTAX InetAddress
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Destination IP address for this session."
 ::= { rtpSessionEntry 11 }

rtpSessionDestRTPport OBJECT-TYPE
SYNTAX InetPortNumber
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Destination UDP port for RTP. A value of 0 indicates
an unknown port number."
 ::= { rtpSessionEntry 12 }

rtpSessionDestRTCPport OBJECT-TYPE
SYNTAX InetPortNumber
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Destination UDP port for RTCP. A value of 0 indicates
an unknown port number."
 ::= { rtpSessionEntry 13 }
rtpSessionSrcIdenType OBJECT-TYPE
   SYNTAX INTEGER {dialedNumber (1),
                   urlID (2),
                   other (3) }
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "Defines the type of address in parameter
       rtpSessionSourceIdentifier"
   ::= { rtpSessionEntry 14 }

rtpSessionSrcIdentifier OBJECT-TYPE
   SYNTAX OCTET STRING (SIZE(0..128))
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "Alternate identifier to the IP address.  This can be E.164,
       DN, or URL."
   ::= { rtpSessionEntry 15 }

rtpSessionDestIdenType OBJECT-TYPE
   SYNTAX INTEGER {dialedNumber (1),
                   urlID (2),
                   other (3) }
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "Defines the type of address in parameter
       rtpSessionDestIdentifier."
   ::= { rtpSessionEntry 16 }

rtpSessionDestIdentifier OBJECT-TYPE
   SYNTAX OCTET STRING (SIZE(0..128))
   MAX-ACCESS read-only
   STATUS current
   DESCRIPTION
      "Alternate identifier to the IP address.  This can be E.164,
       DN, or URL."
   ::= { rtpSessionEntry 17 }

rtpSessionIfIndex OBJECT-TYPE
   SYNTAX InterfaceIndex
   MAX-ACCESS read-create
   STATUS current
   DESCRIPTION
      "The ifIndex value is set to the corresponding value
       from IF-MIB (See RFC 2233, ‘The Interfaces Group MIB using
       SMIv2’).  This is the interface that the RTP stream is being sent
to or received from, or in the case of an RTP Monitor the
interface that RTCP packets will be received on.  Cannot be
changed if rtpSessionRowStatus is ‘active’."
   ::= { rtpSessionEntry 18 }

Clark                   Expires December 2006                [Page 11]
rtpSessionMonitor OBJECT-TYPE
SYNTAX          TruthValue
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
  "Boolean, Set to ‘true(1)’ if remote senders or receivers in
addition to the local RTP System are to be monitored using RTCP.
RTP Monitors MUST initialize to ‘true(1)’ and RTP Hosts SHOULD
initialize this ‘false(2)’. Note that because ‘host monitor’
systems are receiving RTCP from their remote participants they
MUST set this value to ‘true(1)’.”
::= { rtpSessionEntry 19 }

rtpSessionSenderJoins OBJECT-TYPE
SYNTAX          Counter32
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
  "The number of senders that have been observed to have
joined the session since this conceptual row was created
(rtpSessionStartTime). A sender ‘joins’ an RTP
session by sending to it. Senders that leave and then
re-join following an RTCP BYE (see RFC 3550, ‘RTP: A
Transport Protocol for Real-Time Applications,’ sec. 6.6)
or session timeout may be counted twice. Every time a new
RTP sender is detected either using RTP or RTCP, this counter
is incremented.”
::= { rtpSessionEntry 20 }

rtpSessionReceiverJoins OBJECT-TYPE
SYNTAX          Counter32
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
  "The number of receivers that have been observed to have
joined this session since this conceptual row was created
(rtpSessionStartTime). A receiver ‘joins’ an RTP
session by sending RTCP Receiver Reports to the session.
Receivers that leave and then re-join following an RTCP BYE
(see RFC 3550, ‘RTP: A Transport Protocol for Real-Time
Applications,’ sec. 6.6) or session timeout may be counted
twice."
::= { rtpSessionEntry 21 }

rtpSessionByes OBJECT-TYPE
SYNTAX          Counter32
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
  "A count of RTCP BYE (see RFC 3550, ‘RTP: A Transport
Protocol for Real-Time Applications,’ sec. 6.6) messages
received by this entity.”
::= { rtpSessionEntry 22 }
rtpSessionRowStatus OBJECT-TYPE
SYNTAX RowStatus
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Value of ‘active’ when RTP or RTCP messages are being
sent or received by an RTP System. A newly-created
conceptual row must have the all read-create objects
initialized before becoming ‘active’. A conceptual row that is in the ‘notReady’ or ‘notInService’
state MAY be removed after 5 minutes."
 ::= { rtpSessionEntry 23 }

rtpSessionMaxNumEntries OBJECT-TYPE
SYNTAX Integer32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The maximum number of entries that can be supported
in this table."
 ::= { rtpSessionEntry 24 }

-- SENDER INVERSE TABLE
--
rtpSenderInverseTable OBJECT-TYPE
SYNTAX SEQUENCE OF RtpSenderInverseEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Maps rtpSenderIPAddress, rtpSessionIndex, to the rtpSenderSSRC
index of the rtpSenderTable. This table allows management
applications to find entries sorted by Sender IP address rather than sorted by rtpSessionIndex. Given the rtpSessionDomain and rtpSenderAddr, a set of rtpSessionIndex and rtpSenderSSRC values can be returned from a tree walk. When rtpSessionIndex is specified in the SNMP Get-Next operations, one or more rtpSenderSSRC values may be returned.

```plaintext
::= { rtpMIBV2Objects 4 }
```

**rtpSenderInverseEntry**

SYNTAX RtpSenderInverseEntry
MAX-ACCESS not-accessible
STATUS current

DESCRIPTION
"Each entry corresponds to exactly one entry in the rtpSenderTable - the entry containing the index pair, rtpSessionIndex, rtpSenderSSRC."

INDEX { rtpSenderIPaddress, rtpSenderRTPport, rtpSessionCallState, rtpSessionIndex, rtpSenderSSRC }

```plaintext
 ::= { rtpSenderInverseTable 1 }
```

**RtpSenderInverseEntry** ::= SEQUENCE {
  rtpSenderInverseStartTime     DateAndTime
}

**rtpSenderInverseStartTime**

SYNTAX DateAndTime
MAX-ACCESS read-only
STATUS current

DESCRIPTION
"The time at which this row was created."

```plaintext
 ::= { rtpSenderInverseEntry 1 }
```

-- SENDERS TABLE --

**rtpSenderTable**

SYNTAX SEQUENCE OF RtpSenderEntry
MAX-ACCESS not-accessible
STATUS current

DESCRIPTION
"Table of information about a sender or senders to an RTP Session. RTP sending hosts MUST have an entry in this table for each stream being sent. RTP receiving hosts MAY have an entry in this table for each sending stream being received by this host. RTP monitors MUST create an entry for each observed sender to a multicast RTP Session as a side-effect when a
conceptual row in the rtpSessionTable is made ‘active’ by a manager.

::= { rtpMIBV2Objects 5 }

rtpSenderEntry OBJECT-TYPE
SYNTAX          RtpSenderEntry
MAX-ACCESS      not-accessible
STATUS          current
DESCRIPTION      "Each entry contains information from a single RTP Sender
Synchronization Source (SSRC, see RFC 3550 ‘RTP: A Transport
Protocol for Real-Time Applications’ sec.6). The session is
identified to the the SNMP entity by rtpSessionIndex.
Rows are removed by the RTP agent when a BYE is received
from the sender or when the sender times out (see RFC
3550, Sec. 6.2.1) or when the rtpSessionEntry is deleted."
INDEX { rtpSessionCallState, rtpSessionIndex, rtpSenderSSRC }
::= { rtpSenderTable 1 }

RtpSenderEntry ::= SEQUENCE {
  rtpSenderSSRC              Unsigned32,
  rtpSenderCNAME             Utf8String,
  rtpSenderIPtype            InetAddressType,
  rtpSenderIPaddress         InetAddress,
  rtpSenderRTPport           InetPortNumber,
  rtpSenderRTCPport          InetPortNumber,
  rtpSenderPackets           Counter64,
  rtpSenderOctets            Counter64,
  rtpSenderTool              Utf8String,
  rtpSenderSRs               Counter32,
  rtpSenderSRTime            DateAndTime,
  rtpSenderPT                Integer32,
  rtpSenderStartTime         DateAndTime
}

rtpSenderSSRC OBJECT-TYPE
SYNTAX          Unsigned32
MAX-ACCESS      not-accessible
STATUS          current
DESCRIPTION      "The RTP SSRC, or synchronization source identifier of the
sender. The RTP session address plus an SSRC uniquely
identify a sender to an RTP session (see RFC 3550, ‘RTP: A
Transport Protocol for Real-Time Applications’ sec.3)."
::= { rtpSenderEntry 1 }

rtpSenderCNAME OBJECT-TYPE
SYNTAX          Utf8String
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION      "The RTP canonical name of the sender."
::= { rtpSenderEntry 2 }

Clark                   Expires December 2006                [Page 15]
rtpSenderIPtype OBJECT-TYPE
  SYNTAX InetAddressType
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
  "IP address type for the originating IP endpoint for this RTP stream."
  ::= { rtpSenderEntry 3 }

rtpSenderIPaddress OBJECT-TYPE
  SYNTAX InetAddress
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
  "IP address for the originating IP endpoint for this RTP stream."
  ::= { rtpSenderEntry 4 }

rtpSenderRTPport OBJECT-TYPE
  SYNTAX InetPortNumber
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
  "Source UDP port for RTP. A value of 0 indicates an unknown port number."
  ::= { rtpSenderEntry 5 }

rtpSenderRTCPport OBJECT-TYPE
  SYNTAX InetPortNumber
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
  "Source UDP port for RTCP. A value of 0 indicates an unknown port number."
  ::= { rtpSenderEntry 6 }

rtpSenderPackets OBJECT-TYPE
  SYNTAX Counter64
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
  "Count of RTP packets sent by this sender, or observed by an RTP monitor, since rtpSenderStartTime."
  ::= { rtpSenderEntry 7 }

rtpSenderOctets OBJECT-TYPE
  SYNTAX Counter64
  MAX-ACCESS read-only
  STATUS current
  DESCRIPTION
  "Count of non-header RTP octets sent by this sender, or observed..."
by an RTP monitor, since rtpSenderStartTime.

::= { rtpSenderEntry 8 }

rtpSenderTool OBJECT-TYPE
SYNTAX     Utf8String (SIZE(0..127))
MAX-ACCESS read-only
STATUS     current
DESCRIPTION "Name of the application program source of the stream."

::= { rtpSenderEntry 9 }

rtpSenderSRs OBJECT-TYPE
SYNTAX     Counter32
MAX-ACCESS read-only
STATUS     current
DESCRIPTION "A count of the number of RTCP Sender Reports that have
been sent from this sender, or observed if the RTP entity
is a monitor, since rtpSenderStartTime."

::= { rtpSenderEntry 10 }

rtpSenderSRTime OBJECT-TYPE
SYNTAX     DateAndTime
MAX-ACCESS read-only
STATUS     current
DESCRIPTION "rtpSenderSRTime is the time at which
the last SR was received from this sender, in the case of a
monitor or receiving host. Or sent by this sender, in the
case of a sending host."

::= { rtpSenderEntry 11 }

rtpSenderPT OBJECT-TYPE
SYNTAX     Integer32(0..127)
MAX-ACCESS read-only
STATUS     current
DESCRIPTION "Payload type from the RTP header of the most recently received
RTP Packet (see RFC 3550, ‘RTP: A Transport Protocol for
Real-Time Applications’ sec. 5)."

::= { rtpSenderEntry 12 }

rtpSenderStartTime OBJECT-TYPE
SYNTAX     DateAndTime
MAX-ACCESS read-only
STATUS     current
DESCRIPTION "The time at which this row was
created."

::= { rtpSenderEntry 13 }

Clark                   Expires December 2006                [Page 17]
-- RECEIVER INVERSE TABLE

rtpRcvrInverseTable OBJECT-TYPE
SYNTAX        SEQUENCE OF RtpRcvrInverseEntry
MAX-ACCESS    not-accessible
STATUS        current
DESCRIPTION   "Maps rtpRcvrIPaddress and rtpSessionIndex to the rtpRcvrSRCSSRC and rtpRcvrSSRC indexes of the rtpRcvrTable. This table allows management applications to find entries by rtpRcvrIPaddress rather than by rtpSessionIndex. Given rtpSessionDomain and rtpRcvrIPaddress, a set of rtpSessionIndex, rtpRcvrSRCSSRC, and rtpRcvrSSRC values can be returned from a tree walk. When rtpSessionIndex is specified in SNMP Get-Next operations, one or more rtpRcvrSRCSSRC and rtpRcvrSSRC pairs may be returned."
 ::= { rtpMIBV2Objects 6 }

RtpRcvrInverseEntry OBJECT-TYPE
SYNTAX        RtpRcvrInverseEntry
MAX-ACCESS    not-accessible
STATUS        current
DESCRIPTION   "Each entry corresponds to exactly one entry in the rtpRcvrTable - the entry containing the index pair, rtpSessionIndex, rtpRcvrSSRC."
INDEX { rtpRcvrIPaddress, rtpRcvrRTPport, rtpSessionCallState, rtpSessionIndex, rtpRcvrSRCSSRC, rtpRcvrSSRC }
 ::= { rtpRcvrInverseTable 1 }

RtpRcvrInverseEntry ::= SEQUENCE {
    rtpRcvrInverseStartTime DateAndTime
}

rtpRcvrInverseStartTime OBJECT-TYPE
SYNTAX        DateAndTime
MAX-ACCESS    read-only
STATUS        current
DESCRIPTION   "The time at which this row was created."
 ::= { rtpRcvrInverseEntry 1 }

--

-- RECEIVERS TABLE

rtpRcvrTable OBJECT-TYPE
SYNTAX        SEQUENCE OF RtpRcvrEntry
MAX-ACCESS    not-accessible

Clark                   Expires December 2006                [Page 18]
STATUS current
DESCRIPTION "Table of information about a receiver or receivers of RTP session data. RTP hosts that receive RTP session packets MUST create an entry in this table for that receiver/sender pair. RTP hosts that send RTP session packets MAY create an entry in this table for each receiver to their stream using RTCP feedback from the RTP group. RTP monitors create an entry for each observed RTP session receiver as a side effect when a conceptual row in the rtpSessionTable is made ‘active’ by a manager."
 ::= { rtpMIBV2Objects 7 }
rtpRcvrEntry OBJECT-TYPE
SYNTAX     RtpRcvrEntry
MAX-ACCESS not-accessible
STATUS     current
DESCRIPTION "Each entry contains information from a single RTP Synchronization Source that is receiving packets from the sender identified by rtpRcvrSRCSSRC (SSRC, see RFC 3550, ‘RTP: A Transport Protocol for Real-Time Applications’ sec.6). The session is identified to the the RTP Agent entity by rtpSessionIndex. Rows are removed by the RTP agent when a BYE is received from the sender or when the sender times out (see RFC 3550) or when the rtpSessionEntry is deleted."
INDEX { rtpSessionCallState, rtpSessionIndex, rtpRcvrSRCSSRC, rtpRcvrSSRC }
 ::= { rtpRcvrTable 1 }
RtpRcvrEntry ::= SEQUENCE {
  rtpRcvrSRCSSRC        Unsigned32,
  rtpRcvrSSRC           Unsigned32,
  rtpRcvrCNAME          Utf8String,
  rtpRcvrIPtype         InetAddressType,
  rtpRcvrIPaddress      InetAddress,
  rtpRcvrRTPport        InetPortNumber,
  rtpRcvrRTCPport       InetPortNumber,
  rtpRcvrRTT            Gauge32,
  rtpRcvrLostPackets    Counter64,
  rtpRcvrJitter         Gauge32,
  rtpRcvrTool           Utf8String,
  rtpRcvrRRs            Counter32,
  rtpRcvrRRTime         DateAndTime,
  rtpRcvrPT             Integer32,
  rtpRcvrRTT            DateAndTime,
  rtpRcvrStart         DateAndTime
}

rtpRcvrSRCSSRC OBJECT-TYPE
SYNTAX     Unsigned32
MAX-ACCESS not-accessible
 STATUS: current
DESCRIPTION:
"The RTP SSRC, or synchronization source identifier of the
sender. The RTP session address plus an SSRC uniquely
identify a sender or receiver of an RTP stream (see RFC
3550, ‘RTP: A Transport Protocol for Real-Time
Applications’ sec.3)."
::= { rtpRcvrEntry 1 }

rtpRcvrSSRC OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION:
"The RTP SSRC, or synchronization source identifier of the
receiver. The RTP session address plus an SSRC uniquely
identify a receiver of an RTP stream (see RFC 3550, ‘RTP: A
Transport Protocol for Real-Time Applications’ sec.3)."
::= { rtpRcvrEntry 2 }

rtpRcvrCNAME OBJECT-TYPE
SYNTAX Utf8String
MAX-ACCESS read-only
STATUS current
DESCRIPTION:
"The RTP canonical name of the receiver."
::= { rtpRcvrEntry 3 }

rtpRcvrIPtype OBJECT-TYPE
SYNTAX InetAddressType
MAX-ACCESS read-only
STATUS current
DESCRIPTION:
"Destination IP address type for this session."
::= { rtpRcvrEntry 4 }

rtpRcvrIPaddress OBJECT-TYPE
SYNTAX InetAddress
MAX-ACCESS read-only
STATUS current
DESCRIPTION:
"Destination IP address for this session."
::= { rtpRcvrEntry 5 }

rtpRcvrRTPport OBJECT-TYPE
SYNTAX InetPortNumber
MAX-ACCESS read-only
STATUS current
DESCRIPTION:
"Destination UDP port for RTP. A value of 0 indicates
an unknown port number."
::= { rtpRcvrEntry 6 }
rtpRcvrRTCPport OBJECT-TYPE
SYNTAX  InetPortNumber
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Destination UDP port for RTCP. A value of 0 indicates an unknown port number."
::= { rtpRcvrEntry 7 }

rtpRcvrRTT OBJECT-TYPE
SYNTAX       Gauge32
MAX-ACCESS   read-only
STATUS       current
DESCRIPTION
"The round trip time measurement taken by the source of the RTP stream based on the algorithm described on sec. 6 of RFC 3550, 'RTP: A Transport Protocol for Real-Time Applications.' This algorithm can produce meaningful results when the RTP agent has the same clock as the stream sender (when the RTP monitor is also the sending host for the particular receiver). Otherwise, the entity should return 'noSuchInstance' in response to queries against rtpRcvrRTT."
::= { rtpRcvrEntry 8 }

rtpRcvrLostPackets OBJECT-TYPE
SYNTAX          Counter64
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
"A count of RTP packets lost as observed by this receiver since rtpRcvrStartTime."
::= { rtpRcvrEntry 9 }

rtpRcvrJitter OBJECT-TYPE
SYNTAX          Gauge32
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
"An estimate of delay variation as observed by this receiver. (see RFC 3550, 'RTP: A Transport Protocol for Real-Time Applications' sec.6.3.1 and A.8)."
::= { rtpRcvrEntry 10 }

rtpRcvrTool OBJECT-TYPE
SYNTAX          Utf8String (SIZE(0..127))
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
"Name of the application program source of the stream."
::= { rtpRcvrEntry 11 }
draft-ietf-avt-mib-rtp-bis-01

rtpRcvrRRs  OBJECT-TYPE
SYNTAX      Counter32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
  "A count of the number of RTCP Receiver Reports that have
  been sent from this receiver, or observed if the RTP entity
  is a monitor, since rtpRcvrStartTime."
 ::= { rtpRcvrEntry 12 }

rtpRcvrRRTime  OBJECT-TYPE
SYNTAX       DateAndTime
MAX-ACCESS   read-only
STATUS       current
DESCRIPTION
  "rtpRcvrRRTime is the time at which the last RTCP Receiver Report
  was received from this receiver, in the case of a monitor or RR
  receiver (the RTP Sender). It is the time at which the last
  RR was sent by this receiver in the case of an RTP receiver
  sending the RR."
 ::= { rtpRcvrEntry 13 }

rtpRcvrPT  OBJECT-TYPE
SYNTAX      Integer32(0..127)
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
  "Static or dynamic payload type from the RTP header (see
  RFC 3550, ‘RTP: A Transport Protocol for Real-Time
  Applications’ sec. 5)."
 ::= { rtpRcvrEntry 14 }

rtpRcvrPackets  OBJECT-TYPE
SYNTAX     Counter64
MAX-ACCESS read-only
STATUS     current
DESCRIPTION
  "Count of RTP packets received by this RTP host receiver
  since rtpRcvrStartTime."
 ::= { rtpRcvrEntry 15 }

rtpRcvrOctets  OBJECT-TYPE
SYNTAX    Counter64
MAX-ACCESS read-only
STATUS    current
DESCRIPTION
  "Count of non-header RTP octets received by this receiving RTP
  host since rtpRcvrStartTime."
 ::= { rtpRcvrEntry 16 }

Clark                    Expires December 2006                [Page 22]
rtpRcvrStartTime OBJECT-TYPE
SYNTAX DateAndTime
MAX-ACCESS read-only
STATUS current
DESCRIPTION "The time at which this row was created."
::= { rtpRcvrEntry 17 }

-- MODULE GROUPS

-- There are two types of RTP Systems, RTP hosts and RTP Monitors.
-- Thus there are three kinds of objects: 1) Objects common to both
-- kinds of systems, 2) Objects unique to RTP Hosts and 3) Objects
-- unique to RTP Monitors. There is a fourth group, 4) Objects that
-- SHOULD be implemented by Multicast hosts and RTP Monitors

rtpGroups OBJECT IDENTIFIER ::= { rtpConformance 1 }
rtpSystemGroup OBJECT-GROUP
OBJECTS
{
  rtpSessionSessionIdentifier,
  rtpSessionStartTime,
  rtpSessionStopTime,
  rtpSessionDestIPType,
  rtpSessionDestIPaddress,
  rtpSessionDestRTPport,
  rtpSessionDestRTCPport,
  rtpSessionSrceIdenType,
  rtpSessionSrceIdentifier,
  rtpSessionDestIdenType,
  rtpSessionDestIdentifier,
  rtpSessionIfIndex,
  rtpSessionSenderJoins,
  rtpSessionReceiverJoins,
  rtpSessionByes,
  rtpSessionMonitor,
  rtpSessionMaxNumEntries,
  rtpSenderCNAME,
  rtpSenderIPType,
  rtpSenderIPaddress,
  rtpSenderRTPport,
  rtpSenderRTCPport,
  rtpSenderPackets,
  rtpSenderOctets,
  rtpSenderTool,
  rtpSenderSRs,
  rtpSenderSRTIME,
  rtpSenderStartTime,
  rtpRcvrCNAME,
  rtpRcvrIPType,
  rtpRcvrIPaddress,
  rtpRcvrRTPport,
  rtpRcvrRTCPport,
  rtpRcvrLostPackets,
rtpRcvrJitter,  
rtpRcvrTool,  
rtpRcvrRRs,  
rtpRcvrRRTime,  
rtpRcvrStartTime
}

STATUS          current
DESCRIPTION
"Objects available to all RTP Systems."
 ::= { rtpGroups 1 }

rtpHostGroup    OBJECT-GROUP
    OBJECTS     {
        rtpSessionSourceIPtype, 
        rtpSessionSourceIPaddress, 
        rtpSessionSourceRTPport, 
        rtpSessionSourceRTCPport, 
        rtpSenderPT, 
        rtpRcvrPT, 
        rtpRcvrRTT, 
        rtpRcvrOctets, 
        rtpRcvrPackets
    }

STATUS          current
DESCRIPTION
"Objects that are available to RTP Host systems, but may not be available to RTP Monitor systems."
 ::= { rtpGroups 2 }

rtpMonitorGroup OBJECT-GROUP
    OBJECTS     {
        rtpSessionNewIndex, 
        rtpSessionRowStatus
    }

STATUS          current
DESCRIPTION
"Objects used to create rows in the RTP Session Table. These objects are not needed if the system does not create rows."
 ::= { rtpGroups 3 }

rtpInverseGroup OBJECT-GROUP
    OBJECTS     {
        rtpSessionInverseStartTime, 
        rtpSenderInverseStartTime, 
        rtpRcvrInverseStartTime
    }

STATUS          current
DESCRIPTION
"Objects used in the Inverse Lookup Tables."
 ::= { rtpGroups 4 }

-- Compliance
--
rtpCompliances OBJECT IDENTIFIER ::= { rtpConformance 2 }

rtpHostCompliance MODULE-COMPLIANCE
  STATUS current
  DESCRIPTION "Host implementations MUST comply."
  MODULE RTP-MIB
  MANDATORY-GROUPS {
    rtpSystemGroup, rtpHostGroup
  }
  GROUP rtpMonitorGroup
  DESCRIPTION "Host systems may optionally support row creation and deletion. This would allow an RTP Host system to act as an RTP Monitor."
  GROUP rtpInverseGroup
  DESCRIPTION "Multicast RTP Systems SHOULD implement the optional tables."

OBJECT rtpSessionNewIndex
  MIN-ACCESS not-accessible
  DESCRIPTION "RTP system implementations support of row creation and deletion is OPTIONAL so implementation of this object is OPTIONAL."

OBJECT rtpSessionDestIPtype
  MIN-ACCESS read-only
  DESCRIPTION "Row creation and deletion is OPTIONAL so read-create access to this object is OPTIONAL."

OBJECT rtpSessionDestIPAddress
  MIN-ACCESS read-only
  DESCRIPTION "Row creation and deletion is OPTIONAL so read-create access to this object is OPTIONAL."

OBJECT rtpSessionDestRTPport
  MIN-ACCESS read-only
  DESCRIPTION "Row creation and deletion is OPTIONAL so read-create access to this object is OPTIONAL."

OBJECT rtpSessionDestRTCPport
  MIN-ACCESS read-only
  DESCRIPTION "Row creation and deletion is OPTIONAL so read-create access to this object is OPTIONAL."

OBJECT rtpSessionIfIndex
  MIN-ACCESS read-only
  DESCRIPTION "Row creation and deletion is OPTIONAL so read-create access to this object is OPTIONAL."

OBJECT rtpSessionRowStatus
  MIN-ACCESS not-accessible
DESCRIPTION
"Row creation and deletion is OPTIONAL so read-create access to this object is OPTIONAL."

OBJECT rtpSessionInverseStartTime
MIN-ACCESS not-accessible
DESCRIPTION
"Multicast RTP Systems SHOULD implement the optional tables."

OBJECT rtpSenderInverseStartTime
MIN-ACCESS not-accessible
DESCRIPTION
"Multicast RTP Systems SHOULD implement the optional tables."

OBJECT rtpRcvrInverseStartTime
MIN-ACCESS not-accessible
DESCRIPTION
"Multicast RTP Systems SHOULD implement the optional tables."

::= { rtpCompliances 1 }
OBJECT rtpSessionSourceRTPport
MIN-ACCESS not-accessible
DESCRIPTION
"RTP monitor sourcing of RTP or RTCP data packets
is OPTIONAL and implementation of this object is
OPTIONAL."

OBJECT rtpSessionSourceRTCPport
MIN-ACCESS not-accessible
DESCRIPTION
"RTP monitor sourcing of RTP or RTCP data packets
is OPTIONAL and implementation of this object is
OPTIONAL."

OBJECT rtpRcvrPT
MIN-ACCESS not-accessible
DESCRIPTION
"RTP monitor systems may not support
retrieval of the RTP Payload Type from the RTP
header (and may receive RTCP messages only). When
queried for the payload type information."

OBJECT rtpSenderPT
MIN-ACCESS not-accessible
DESCRIPTION
"RTP monitor systems may not support
retrieval of the RTP Payload Type from the RTP
header (and may receive RTCP messages only). When
queried for the payload type information."

OBJECT rtpRcvrOctets
MIN-ACCESS not-accessible
DESCRIPTION
"RTP monitor systems may receive only the RTCP messages
and not the RTP messages that contain the octet count
of the RTP message. Thus implementation of this
object is OPTIONAL"

OBJECT rtpRcvrPackets
MIN-ACCESS not-accessible
DESCRIPTION
"RTP monitor systems may receive only the RTCP messages
and not the RTP messages that contain the octet count
of the RTP message. Thus implementation of this
object is OPTIONAL."

OBJECT rtpSessionIfIndex
MIN-ACCESS read-only
DESCRIPTION
"Row creation and deletion is OPTIONAL so
read-create access to this object is OPTIONAL."

OBJECT rtpSessionInverseStartTime
MIN-ACCESS not-accessible
DESCRIPTION
"Multicast RTP Systems SHOULD implement the optional
tables."
OBJECT  rtpSenderInverseStartTime
MIN-ACCESS not-accessible
DESCRIPTION
"Multicast RTP Systems SHOULD implement the optional tables."

OBJECT  rtprcvrInverseStartTime
MIN-ACCESS not-accessible
DESCRIPTION
"Multicast RTP Systems SHOULD implement the optional tables."

::= { rtpCompliances 2 }

END
4. Security Considerations

In most cases, MIBs are not themselves security risks; if SNMP security is operating as intended, the use of a MIB to view information about a system, or to change some parameter at the system, is a tool, not a threat. However, there are a number of management objects defined in this MIB that have a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations.

None of the read-only objects in this MIB reports a password, though some SDES [RFC3550] items such as the CNAME [RFC3550], the canonical name, may be deemed sensitive depending on the security policies of a particular enterprise. If access to these objects is not limited by an appropriate access control policy, these objects can provide an attacker with information about a system’s configuration and the services that that system is providing. Some enterprises view their network and system configurations, as well as information about usage and performance, as corporate assets; such enterprises may wish to restrict SNMP access to most of the objects in the MIB. This MIB supports read-write operations against rtpSessionNewIndex which has the side effect of creating an entry in the rtpSessionTable when it is written to. Five objects in rtpSessionEntry have read-create access: rtpSessionDomain, rtpSessionRemAddr, rtpSessionIfIndex, rtpSessionRowStatus, and rtpSessionIfAddr identify an RTP session to be monitored on a particular interface. The values of these objects are not to be changed once created, and initialization of these objects affects only the monitoring of an RTP session and not the operation of an RTP session on any host end-system. Since write operations to rtpSessionNewIndex and the five objects in rtpSessionEntry affect the operation of the monitor, write access to these objects should be subject to access control.

Confidentiality of RTP and RTCP data packets is defined in section 9 of the RTP specification [RFC3550]. Encryption may be performed on RTP packets, RTCP packets, or both. Encryption of RTCP packets may pose a problem for third-party monitors though "For RTCP, it is allowed to split a compound RTCP packet into two lower-layer packets, one to be encrypted and one to be sent in the clear. For example, SDES information might be encrypted while reception reports were sent in the clear to accommodate third-party monitors [RFC3550]."

SNMPv1 by itself is not a secure environment. Even if the network...
itself is secure (for example by using IPSec), there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB. It is recommended that the implementers consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model [RFC 2574] and the View-based Access Control Model [RFC 2575] is recommended. It is then a customer/user responsibility to ensure that the SNMP entity giving access to an instance of this MIB, is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

5. IANA Considerations
   TBD

6. Acknowledgements
   The authors wish to thank Brian Park for his contributions in reviewing this MIB.

7. Intellectual Property
   The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

8. References
9. Informative References


10. Authors' Addresses

Alan Clark
Telchemy Incorporated
2905 Premiere Parkway, Suite 280
Duluth, Georgia 30097
U.S.A.
Email: alan@telchemy.com

Amy Pendleton
Nortel
2380 Performance Drive
Richardson, Texas 75081
U.S.A.
Email: aspen@nortel.com

Full Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET
ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgement

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