

Audio Video Transport Group
Internet-Draft
Expires September 6, 2000

Mark Baugher
Intel Corp.
Irina Suconick
VideoServer Corp.
Bill Strahm
Intel Corp.
March 6, 2000

Real-Time Transport Protocol
Management Information Base
<[draft-ietf-avt-rtp-mib-13.txt](#)>

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of [section 10 of RFC2026](#). 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. Internet-Drafts may be updated, replaced, or made obsolete by other documents at any time. It is not appropriate to use Internet-Drafts as reference material or to cite them other than as a ``working draft'' or ``work in progress.''
To learn the current status of any Internet-Draft, please check the lid-abstracts.txt listing contained in the Internet-Drafts Shadow Directories on ftp.ietf.org, nic.nordu.net, venera.isi.edu, or munnari.oz.au.

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

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

Copyright Notice

Copyright (C) The Internet Society (2000). All Rights Reserved.

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 [[RFC1889](#)].

Internet Draft RTP MIB

March 2000

Table of Contents

1	The Network Management Framework	3
2	Overview	4
2.1	Components	4
2.2	Applicability of the MIB to RTP System Implementations	5
2.3	The Structure of the RTP MIB	5
3	Definitions	7
4	Security Issues	31
5	Acknowledgements	32
6	Intellectual Property	32
7	References	33
8	Authors Addresses	35
9	Full Copyright Statement	35

1. The SNMP Management Framework

The SNMP Management Framework presently consists of five major components:

- o An overall architecture, described in [RFC 2571](#) [[RFC2571](#)].
- o Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIV1 and described in STD 16, [RFC 1155](#) [[RFC1155](#)], STD 16, [RFC 1212](#) [[RFC1212](#)] and [RFC 1215](#) [[RFC1215](#)]. The second version, called SMIV2, is described in STD 58, [RFC 2578](#) [[RFC2578](#)], [RFC 2579](#) [[RFC2579](#)] and [RFC 2580](#) [[RFC2580](#)].
- o Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in STD 15, [RFC 1157](#) [[RFC1157](#)]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in [RFC 1901](#) [[RFC1901](#)] and [RFC 1906](#) [[RFC1906](#)]. The third version of the message protocol is called SNMPv3 and described in [RFC 1906](#) [[RFC1906](#)], [RFC 2572](#) [[RFC2572](#)] and [RFC 2574](#) [[RFC2574](#)].

- o Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in STD 15, [RFC 1157](#) [[RFC1157](#)]. A second set of protocol operations and associated PDU formats is described in [RFC 1905](#) [[RFC1905](#)].
- o A set of fundamental applications described in [RFC 2573](#) [[RFC2573](#)] and the view-based access control mechanism described in [RFC 2575](#) [[RFC2575](#)].

A more detailed introduction to the current SNMP Management Framework can be found in [RFC 2570](#) [[RFC2570](#)].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the mechanisms defined in the SMI.

This memo specifies a MIB module that is compliant to the SMIV2. A MIB conforming to the SMIV1 can be produced through the appropriate translations. The resulting translated MIB must be semantically equivalent, except where objects or events are omitted because no translation is possible (use of Counter64). Some machine readable information in SMIV2 will be converted into textual descriptions in SMIV1 during the translation process. However, this loss of machine readable information is not considered to change the semantics of the MIB.

[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

[[RFC1889](#)]. 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 \[RFC1889\]](#).

[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 \[RFC1889\]](#). The sender is also a source of RTCP Sender Report packets as specified in [section 6 of \[RFC1889\]](#).

[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 \[RFC1889\]](#).

Bagher, Strahm, Suconick

Expires September 6, 2000

[Page 4]

Internet Draft RTP MIB

March 2000

[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 \[RFC1889\]](#). Use of the RTP MIB for RTP Translators and Mixers, as defined in [section 7 of \[RFC1889\]](#), 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 manager 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`.

[2.3](#) 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
RowStatus, TAddress,	
TDomain, TestAndIncr,	
TimeStamp, TruthValue	FROM SNMPv2-TC
OBJECT-GROUP, MODULE-COMPLIANCE	FROM SNMPv2-CONF
Utf8String	FROM SYSAPPL-MIB
InterfaceIndex	FROM IF-MIB;

rtpMIB MODULE-IDENTITY

LAST-UPDATED "200012070000Z" -- 12 July 2000

ORGANIZATION

"IETF AVT Working Group

Email: rem-conf@es.net"

CONTACT-INFO

"Mark Baugher

Postal: Intel Corporation

2111 NE 25th Avenue

Hillsboro, OR 97124

United States

Tel: +1 503 466 8406

Email: mbaugher@passededge.com

Bill Strahm

Postal: Intel Corporation

2111 NE 25th Avenue

Hillsboro, OR 97124

United States

Tel: +1 503 264 4632

Email: bill.strahm@intel.com

Irina Suconick

Postal: Ennovate Networks

60 Codman Hill Rd.,

Boxboro, Ma 01719

Tel: +1 781-505-2155

Email: irina@ennovatenetworks.com"

Internet Draft RTP MIB

March 2000

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 1889](#), 'RTP: A Transport Protocol for Real-Time Applications,' [section 3.0](#), for definitions."

REVISION "200012070000Z" -- 12 July 2000

DESCRIPTION "Initial version of this MIB.

Published as RFC xxx." -- RFC-Editor assigns xxx

::= { mib-2 xxx } -- to be assigned by IANA

--

-- OBJECTS

--

rtpMIBObjects OBJECT IDENTIFIER ::= { rtpMIB 1 }

rtpConformance OBJECT IDENTIFIER ::= { rtpMIB 2 }

--

-- SESSION NEW INDEX

--

rtpSessionNewIndex OBJECT-TYPE

SYNTAX TestAndIncr

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"This object is used to assign values to rtpSessionIndex as described in 'Textual Conventions for SMIV2'. 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."

::= { rtpMIBObjects 1 }

--

-- SESSION INVERSE TABLE

--

rtpSessionInverseTable OBJECT-TYPE

SYNTAX SEQUENCE OF RtpSessionInverseEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Maps rtpSessionDomain, rtpSessionRemAddr, and rtpSessionLocAddr TAddress pairs to one or more rtpSessionIndex values, each describing a row in the rtpSessionTable. This makes it possible to retrieve the row(s) in the rtpSessionTable corresponding to a given session without having to walk the entire (potentially large) table."

::= { rtpMIBObjects 2 }

rtpSessionInverseEntry OBJECT-TYPE

SYNTAX RtpSessionInverseEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Each entry corresponds to exactly one entry in the rtpSessionTable - the entry containing the tuple, rtpSessionDomain, rtpSessionRemAddr, rtpSessionLocAddr and rtpSessionIndex."

INDEX { rtpSessionDomain, rtpSessionRemAddr, rtpSessionLocAddr, rtpSessionIndex }

::= { rtpSessionInverseTable 1 }

RtpSessionInverseEntry ::= SEQUENCE {

 rtpSessionInverseStartTime TimeStamp
}

rtpSessionInverseStartTime OBJECT-TYPE

SYNTAX TimeStamp

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of SysUpTime at the time that this row was created."

::= { rtpSessionInverseEntry 1 }

--

-- SESSION TABLE

--

rtpSessionTable OBJECT-TYPE

SYNTAX SEQUENCE OF RtpSessionEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"There's one entry in rtpSessionTable for each RTP session on which packets are being sent, received, and/or monitored."

::= { rtpMIBObjects 3 }

rtpSessionEntry OBJECT-TYPE

SYNTAX RtpSessionEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"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. Rows created by an RTP agent MUST be deleted when the session is over and there are no rtpRcvrEntry and no rtpSenderEntry for this session. 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)'."

```
INDEX { rtpSessionIndex }
 ::= { rtpSessionTable 1 }
```

```
RtpSessionEntry ::= SEQUENCE {
    rtpSessionIndex      Integer32,
    rtpSessionDomain     TDomain,
    rtpSessionRemAddr    TAddress,
    rtpSessionLocAddr    TAddress,
    rtpSessionIfIndex    InterfaceIndex,
    rtpSessionSenderJoins Counter32,
    rtpSessionReceiverJoins Counter32,
    rtpSessionByes       Counter32,
    rtpSessionStartTime  TimeStamp,
    rtpSessionMonitor    TruthValue,
    rtpSessionRowStatus  RowStatus
}
```

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 1 }
```

rtpSessionDomain OBJECT-TYPE

SYNTAX TDomain
MAX-ACCESS read-create
STATUS current

DESCRIPTION

"The transport-layer protocol used for sending or receiving the stream of RTP data packets on this session.

Cannot be changed if rtpSessionRowStatus is 'active'."

::= { rtpSessionEntry 2 }

rtpSessionRemAddr OBJECT-TYPE

SYNTAX TAddress
MAX-ACCESS read-create
STATUS current

DESCRIPTION

"The address to which RTP packets are sent by the RTP system.

In an IP multicast RTP session, this is the single address used by all senders and receivers of RTP session data. In a unicast RTP session this is the unicast address of the remote RTP system.

'The destination address pair 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 network address pairs.' See [RFC 1889](#), 'RTP: A Transport Protocol for Real-Time Applications,' sec. 3. The transport service is identified by rtpSessionDomain. For snmpUDPDDomain, this is an IP address and even-numbered UDP Port with the RTCP being sent on the next higher odd-numbered port, see [RFC 1889](#), sec. 5."

::= { rtpSessionEntry 3 }

rtpSessionLocAddr OBJECT-TYPE

SYNTAX TAddress
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"The local address used by the RTP system. In an IP multicast RTP session, rtpSessionRemAddr will be the same IP multicast address as rtpSessionLocAddr. In a unicast RTP session, rtpSessionRemAddr and rtpSessionLocAddr will have different unicast addresses. See [RFC 1889](#), 'RTP: A Transport Protocol for Real-Time Applications,' sec. 3. The transport service is identified by rtpSessionDomain. For snmpUDPDomain, this is an IP address and even-numbered UDP Port with the RTCP being sent on the next higher odd-numbered port, see [RFC 1889](#), sec. 5."

::= { rtpSessionEntry 4 }

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 SMIPv2'). 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 5 }

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 1889](#), '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 6 }

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 1889](#), 'RTP: A Transport Protocol for Real-Time Applications,' sec. 6.6) or session timeout may be counted twice."

::= { rtpSessionEntry 7 }

rtpSessionByes OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"A count of RTCP BYE (see [RFC 1889](#), 'RTP: A Transport Protocol for Real-Time Applications,' sec. 6.6) messages received by this entity."

::= { rtpSessionEntry 8 }

rtpSessionStartTime OBJECT-TYPE

SYNTAX TimeStamp
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"The value of SysUpTime at the time that this row was

```

        created."
 ::= { rtpSessionEntry 9 }

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 10 }

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 11 }

```

```

--
-- SENDER INVERSE TABLE
--
rtpSenderInverseTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF RtpSenderInverseEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Maps rtpSenderAddr, rtpSessionIndex, to the rtpSenderSSRC
        index of the rtpSenderTable. This table allows management
        applications to find entries sorted by rtpSenderAddr 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."
    ::= { rtpMIBObjects 4 }

rtpSenderInverseEntry OBJECT-TYPE
    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 { rtpSessionDomain, rtpSenderAddr, rtpSessionIndex,
            rtpSenderSSRC }
    ::= { rtpSenderInverseTable 1 }

RtpSenderInverseEntry ::= SEQUENCE {
    rtpSenderInverseStartTime    TimeStamp
}

rtpSenderInverseStartTime OBJECT-TYPE
    SYNTAX          TimeStamp
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The value of SysUpTime at the time that this row was
        created."
    ::= { rtpSenderInverseEntry 1 }

```

```
--
```

```
-- SENDERS TABLE
```

```
--
```

```
rtpSenderTable OBJECT-TYPE
```

```
    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."
```

```
    ::= { rtpMIBObjects 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 1889 '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
        1889, Sec. 6.2.1) or when the rtpSessionEntry is deleted."
```

```
    INDEX { rtpSessionIndex, rtpSenderSSRC }
```

```
    ::= { rtpSenderTable 1 }
```

```
RtpSenderEntry ::= SEQUENCE {
```

```
    rtpSenderSSRC          Unsigned32,
```

```
    rtpSenderCNAME         Utf8String,
```

```
    rtpSenderAddr          TAddress,
```

```
    rtpSenderPackets       Counter64,
```

```
    rtpSenderOctets        Counter64,
```

```
    rtpSenderTool          Utf8String,
```

```
    rtpSenderSRs           Counter32,
```

```
    rtpSenderSRTime        TimeStamp,
```

```
    rtpSenderPT            INTEGER,
```

```
    rtpSenderStartTime     TimeStamp
```

```
 }
```

Internet Draft RTP MIB

March 2000

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 1889](#), '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 }

rtpSenderAddr OBJECT-TYPE

SYNTAX TAddress
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"The unicast transport source address of the sender. In the case of an RTP Monitor this address is the address that the sender is using to send its RTCP Sender Reports."

::= { rtpSenderEntry 3 }

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 4 }

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 5 }

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 6 }

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 7 }

rtpSenderSRTime OBJECT-TYPE

SYNTAX TimeStamp

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"rtpSenderSRTime is the value of SysUpTime at the time that 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 8 }

rtpSenderPT OBJECT-TYPE

SYNTAX INTEGER (0..127)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Payload type from the RTP header of the most recently received

RTP Packet (see [RFC 1889](#), 'RTP: A Transport Protocol for Real-Time Applications' sec. 5)."
 ::= { rtpSenderEntry 9 }

rtpSenderStartTime OBJECT-TYPE

SYNTAX TimeStamp

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of SysUpTime at the time that this row was created."

::= { rtpSenderEntry 10 }

--

-- RECEIVER INVERSE TABLE

--

rtpRcvrInverseTable OBJECT-TYPE

SYNTAX SEQUENCE OF RtpRcvrInverseEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"Maps rtpRcvrAddr and rtpSessionIndex to the rtpRcvrSRCSSRC and rtpRcvrSSRC indexes of the rtpRcvrTable. This table allows management applications to find entries sorted by rtpRcvrAddr rather than by rtpSessionIndex. Given rtpSessionDomain and rtpRcvrAddr, 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."

::= { rtpMIBObjects 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 { rtpSessionDomain, rtpRcvrAddr, rtpSessionIndex,
 rtpRcvrSRCSSRC, rtpRcvrSSRC }

```

 ::= { rtpRcvrInverseTable 1 }

RtpRcvrInverseEntry ::= SEQUENCE {
    rtpRcvrInverseStartTime    TimeStamp
}

rtpRcvrInverseStartTime OBJECT-TYPE
    SYNTAX                    TimeStamp
    MAX-ACCESS                read-only
    STATUS                    current
    DESCRIPTION
        "The value of SysUpTime at the time that this row was
        created."
 ::= { rtpRcvrInverseEntry 1 }

```

```

--
-- RECEIVERS TABLE
--
rtpRcvrTable OBJECT-TYPE
    SYNTAX                    SEQUENCE OF RtpRcvrEntry
    MAX-ACCESS                not-accessible
    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."
 ::= { rtpMIBObjects 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 1889](#), '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 1889](#), Sec. 6.2.1) or when the rtpSessionEntry is deleted."

INDEX { rtpSessionIndex, rtpRcvrSRCSSRC, rtpRcvrSSRC }
 ::= { rtpRcvrTable 1 }

```
RtpRcvrEntry ::= SEQUENCE {  
    rtpRcvrSRCSSRC      Unsigned32,  
    rtpRcvrSSRC         Unsigned32,  
    rtpRcvrCNAME        Utf8String,  
    rtpRcvrAddr         TAddress,  
    rtpRcvrRTT          Gauge32,  
    rtpRcvrLostPackets  Counter64,  
    rtpRcvrJitter       Gauge32,  
    rtpRcvrTool         Utf8String,  
    rtpRcvrRRs          Counter32,  
    rtpRcvrRRTime       TimeStamp,  
    rtpRcvrPT           INTEGER,  
    rtpRcvrPackets      Counter64,  
    rtpRcvrOctets       Counter64,  
    rtpRcvrStartTime    TimeStamp  
}
```

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 1889](#), '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 1889](#), '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 }

Baugher, Strahm, Suconick Expires September 6, 2000 [Page 21]

Internet Draft RTP MIB

March 2000

rtpRcvrAddr OBJECT-TYPE
 SYNTAX TAddress
 MAX-ACCESS read-only
 STATUS current
 DESCRIPTION
 "The unicast transport address on which the receiver is receiving RTP packets and/or RTCP Receiver Reports."
 ::= { rtpRcvrEntry 4 }

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 1889](#), '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 5 }

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 6 }

rtpRcvrJitter OBJECT-TYPE

SYNTAX Gauge32
 MAX-ACCESS read-only
 STATUS current
 DESCRIPTION
 "An estimate of delay variation as observed by this receiver. (see [RFC 1889](#), 'RTP: A Transport Protocol for Real-Time Applications' sec.6.3.1 and A.8)."
 ::= { rtpRcvrEntry 7 }

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 8 }

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 9 }

rtpRcvrRRTime OBJECT-TYPE

SYNTAX TimeStamp
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"rtpRcvrRRTime is the value of SysUpTime at the time that 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 value of SysUpTime at the time that the last RR was sent by this receiver in the case of an RTP receiver sending the RR."

::= { rtpRcvrEntry 10 }

rtpRcvrPT OBJECT-TYPE

SYNTAX INTEGER (0..127)
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"Static or dynamic payload type from the RTP header (see [RFC 1889](#), 'RTP: A Transport Protocol for Real-Time Applications' sec. 5)."

::= { rtpRcvrEntry 11 }

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 12 }

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 13 }

rtpRcvrStartTime OBJECT-TYPE

SYNTAX TimeStamp

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of SysUpTime at the time that this row was
created."

::= { rtpRcvrEntry 14 }

Internet Draft RTP MIB

March 2000

```
--  
-- 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  
        {  
            rtpSessionDomain,  
            rtpSessionRemAddr,  
            rtpSessionIfIndex,  
            rtpSessionSenderJoins,  
            rtpSessionReceiverJoins,  
            rtpSessionStartTime,  
            rtpSessionByes,  
            rtpSessionMonitor,  
            rtpSenderCNAME,  
            rtpSenderAddr,  
            rtpSenderPackets,  
            rtpSenderOctets,  
            rtpSenderTool,  
            rtpSenderSRs,  
            rtpSenderSRTime,  
            rtpSenderStartTime,  
            rtpRcvrCNAME,  
            rtpRcvrAddr,  
            rtpRcvrLostPackets,  
            rtpRcvrJitter,  
            rtpRcvrTool,  
            rtpRcvrRRs,  
            rtpRcvrRRTime,  
            rtpRcvrStartTime  
        }  
    STATUS      current  
    DESCRIPTION  
        "Objects available to all RTP Systems."  
    ::= { rtpGroups 1 }
```

```
rtpHostGroup OBJECT-GROUP
  OBJECTS {
    rtpSessionLocAddr,
    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 rtpSessionDomain
        MIN-ACCESS read-only
```

```

        DESCRIPTION
        "RTP system implementation support of
        row creation and deletion is OPTIONAL.  When
        it is not supported so write access is
        OPTIONAL."
OBJECT   rtpSessionRemAddr
        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 }

```

rtpMonitorCompliance MODULE-COMPLIANCE

STATUS current

DESCRIPTION

"Monitor implementations must comply. RTP Monitors are not required to support creation or deletion."

MODULE RTP-MIB

MANDATORY-GROUPS {
 rtpSystemGroup,
 rtpMonitorGroup
}

GROUP rtpHostGroup

DESCRIPTION

"Monitor implementations may not have access to values in the rtpHostGroup."

GROUP rtpInverseGroup

DESCRIPTION

"Multicast RTP Systems SHOULD implement the optional tables."

OBJECT rtpSessionLocAddr
 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 Issues

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 [[RFC1889](#)] items such as the CNAME [[RFC1889](#)], 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 the appropriate access control policy.

Confidentiality of RTP and RTCP data packets is defined in [section 9](#) of the RTP specification [[RFC1889](#)]. 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 [[RFC1889](#)]."

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](#) [[RFC2574](#)] and the View-based Access Control Model [RFC 2575](#) [[RFC2575](#)] 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.](#) Acknowledgements

The authors wish to thank Bert Wijnen and the participants from the ITU SG-16 management effort for their helpful comments. Alan Batie and Bill Lewis from Intel also contributed greatly to the RTP MIB through their review of various drafts of the MIB and their work on the implementation of an SNMP RTP Monitor. Stan Naudus from 3Com and John Du from Intel contributed to the original RTP MIB design and co-authored the original RTP MIB draft documents; much of their work remains in the current RTP MIB. Bill Fenner provided solid feedback that improved the quality of the final document.

[6.](#) Intellectual Property

The IETF takes no position regarding the validity or scope of any intellectual property 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; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in [BCP-11](#). Copies of claims of rights made available for publication 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 implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

7. References

- [RFC1889] H. Shulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for real-time applications," [RFC 1889](#).
- [RFC2571] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", [RFC 2571](#), April 1999
- [RFC1155] Rose, M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, [RFC 1155](#), May 1990
- [RFC1212] Rose, M., and K. McCloghrie, "Concise MIB Definitions", STD 16, [RFC 1212](#), March 1991
- [RFC1215] M. Rose, "A Convention for Defining Traps for use with the SNMP", [RFC 1215](#), March 1991
- [RFC2578] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, [RFC 2578](#), April 1999
- [RFC2579] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Textual Conventions for SMIv2", STD 58, [RFC 2579](#), April 1999
- [RFC2580] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Conformance Statements for SMIv2", STD 58, [RFC 2580](#), April 1999
- [RFC1157] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", STD 15, [RFC 1157](#), May 1990.
- [RFC1901] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Introduction to Community-based SNMPv2", [RFC 1901](#), January 1996.
- [RFC1906] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", [RFC 1906](#), January 1996.
- [RFC2572] Case, J., Harrington D., Presuhn R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network

- [RFC2574] Blumenthal, U., and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", [RFC 2574](#), April 1999
- [RFC1905] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", [RFC 1905](#), January 1996.
- [RFC2573] Levi, D., Meyer, P., and B. Stewart, "SNMPv3 Applications", [RFC 2573](#), April 1999
- [RFC2575] Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", [RFC 2575](#), April 1999
- [RFC2570] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", [RFC 2570](#), April 1999

Internet Draft RTP MIB

March 2000

[8.](#) Authors Addresses

Mark Baugher
Intel Corporation
[2111](#) N.E.25th Avenue
Hillsboro, Oregon 97124
U.S.A.

Email: mbaugher@passedge.com

Bill Strahm
Intel Corporation
[2111](#) N.E.25th Avenue
Hillsboro, Oregon 97124
U.S.A.

Email: Bill.Strahm@intel.com

Irina Suconick
Ennovate Networks
[60](#) Codman Hill Rd.,
Boxboro, Ma 01719
U.S.A.

Email: irina@ennovatenetworks.com

[9.](#) Full Copyright Statement

Copyright (C) The Internet Society (2000). All Rights Reserved.

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

copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

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

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.