

Management Information Base for Telephony Routing over IP (TRIP)

Status of this Memo

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

Copyright Notice

Copyright (C) The Internet Society (2004).

Abstract

This memo defines a portion of the Management Information Base (MIB) module for use with network management protocols in the Internet community. In particular, it describes a set of managed objects that are used to manage Telephony Routing over IP (TRIP) devices.

Table of Contents

1. The Internet-Standard Management Framework	2
2. Introduction	2
3. Conventions used in this document.	2
4. Overview	2
5. Structure of TRIP MIB.	2
5.1. Textual Conventions.	3
6. Definitions.	4
6.1. TRIP Textual Conventions	4
6.2. TRIP MIB	7
7. Security Considerations.	48
8. References	50
8.1. Normative References	50
8.2. Informative References	51
9. Acknowledgments.	51
10. Authors' Addresses	52
11. Full Copyright Statement	53

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 module objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in this MIB module 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. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes a set of managed objects that are used to schedule management operations periodically or at specified dates and times. Since TRIP [RFC3219] is modeled after the Border Gateway Protocol (BGP-4) [RFC1771], the managed objects for TRIP are also modeled after RFC1657 - Definitions of Managed Objects for the Fourth Version of the Border Gateway Protocol (BGP-4) using SMIV2 [RFC1657].

3. Conventions used in this document

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 BCP 14, RFC 2119 [RFC2119].

4. Overview

This MIB module provides managed objects for TRIP devices defined in Telephony Routing over IP [RFC3219]. TRIP is an inter-domain application-layer control protocol that exchanges information between TRIP location servers (LS) to provide efficient IP telephony routing.

5. Structure of TRIP MIB

This MIB module utilizes the framework described in RFC 2788 [RFC2788] for management of multiple instances of TRIP from a single entity. The Network Services Monitoring MIB module applTable will be populated with entries corresponding to each TRIP Location Server

in the system. Each TRIP Location Server will then have an `applIndex` associated with it. The value assigned to `applIndex` will represent the distinct instance of TRIP.

The TRIP MIB module contains the following groups of objects with each group as part of the management of a singular TRIP entity. Each group covers a section of functionality of TRIP:

- o The `tripConfigGroup` contains the common configuration objects applicable to all TRIP applications referenced by the `applIndex`.
- o The `tripPeerTableConfigGroup` contains the configuration objects applicable to all TRIP peers of the Location Server referenced by the `applIndex`.
- o The `tripRouteGroup` contains the configuration objects related to the routes of all TRIPs of this Location Server.
- o The `tripItadTopologyGroup` contains information about the topology of the TRIP ITADs concerning this Location Server.
- o The `tripPeerTableStatsGroup` contains the statistical objects applicable to all TRIP peers of the Location Server referenced by the `applIndex`.
- o The `tripNotificationGroup` contains notifications that the TRIP application can generate.
- o The `tripNotifObjectGroup` contains the objects needed by one or more of the notifications.

5.1. Textual Conventions

The data types `TripItad` and `TripId` are used as textual conventions in this document. A TRIP ITAD (IP Telephony Administrative Domain) is described in [RFC3219]. A TRIP ID is used as a distinct identifier for a TRIP Location Server. A `TripAppProtocol` is used to identify an application protocol. A `TripAddressFamily` is used to define an address family. `TripCommunityId` is used as a distinct identifier for a TRIP community. `TripProtocolVersion` depicts the version number of the TRIP protocol. `TripSendReceiveMode` describes the operational mode of the TRIP application.

6. Definitions

6.1. TRIP Textual Conventions

TRIP-TC-MIB DEFINITIONS ::= BEGIN

IMPORTS

MODULE-IDENTITY,
Unsigned32,
Integer32,
mib-2
FROM SNMPv2-SMI -- [RFC2578]

TEXTUAL-CONVENTION
FROM SNMPv2-TC; -- [RFC2579]

tripTC MODULE-IDENTITY

LAST-UPDATED "200409020000Z" -- Sep 02, 2004
ORGANIZATION "IETF IPTel Working Group.
Mailing list: iptel@lists.bell-labs.com"

CONTACT-INFO

"Co-editor David Zinman
postal: 265 Ridley Blvd.
Toronto ON, M5M 4N8
Canada
email: dzinman@rogers.com
phone: +1 416 433 4298

Co-editor: David Walker
Sedna Wireless Inc.
postal: 495 March Road, Suite 500
Ottawa, ON K2K 3G1
Canada
email: david.walker@sedna-wireless.com
phone: +1 613 878 8142

Co-editor Jianping Jiang
Syndesis Limited
postal: 30 Fulton Way
Richmond Hill, ON L4B 1J5
Canada

email: jjjiang@syndesis.com
phone: +1 905 886-7818 x2515
"

DESCRIPTION

"Initial version of TRIP (Telephony Routing Over IP)
MIB Textual Conventions module used by other

TRIP-related MIB Modules.

Copyright (C) The Internet Society (2004). This version of this MIB module is part of RFC 3872, see the RFC itself for full legal notices."

REVISION "200409020000Z" -- Sep 02, 2004

DESCRIPTION

"The initial version, Published as RFC 3872."

::= { mib-2 115 }

--

-- Textual Conventions

--

TripItad ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"The values for identifying the IP Telephony Administrative Domain (ITAD)."

SYNTAX Unsigned32 (0..4294967295)

TripId ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"The TRIP Identifier uniquely identifies a LS within its ITAD. It is a 4 octet unsigned integer that may, but not necessarily, represent the IPv4 address of a Location Server. Where bytes 1-4 of the Unsigned32 represent 1-4 bytes of the IPv4 address in network-byte order. For an IPv6 network, TripId will not represent the IPv6 address."

SYNTAX Unsigned32 (0..4294967295)

TripAddressFamily ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"A type of address for a TRIP route. Address families defined within this MIB module are:

Code	Address Family
1	Decimal Routing Numbers
2	PentaDecimal Routing Numbers
3	E.164 Numbers
255	An other type of address family"

SYNTAX INTEGER

{ decimal(1), pentadecimal(2), e164(3), other(255) }

```
TripAppProtocol ::= TEXTUAL-CONVENTION
    STATUS current
    DESCRIPTION
        "The application protocol used for communication with TRIP
        Location Servers. Protocols defined in this MIB Module
        are:

        Code                Protocol
        1                    SIP
        2                    H.323-H.225.0-Q.931
        3                    H.323-H.225.0-RAS
        4                    H.323-H.225.0-Annex-G
        255                  An other type of application protocol"

    SYNTAX INTEGER
        { sip(1), q931(2), ras(3), annexG(4), other(255) }

TripCommunityId ::= TEXTUAL-CONVENTION
    STATUS current
    DESCRIPTION
        "The range of legal values for a TRIP Community
        Identifier."
    SYNTAX Unsigned32 (0..4294967295)

TripProtocolVersion ::= TEXTUAL-CONVENTION
    STATUS current
    DESCRIPTION
        "The version number of the TRIP protocol."
    SYNTAX Integer32 (1..255)

TripSendReceiveMode ::= TEXTUAL-CONVENTION
    STATUS current
    DESCRIPTION
        "The operational mode of the TRIP application. Possible
        values are:
        1 - Send Receive mode
        2 - Send only mode
        3 - Receive Only mode"
    SYNTAX INTEGER { sendReceive(1), sendOnly(2), receiveOnly(3) }

END
```

6.2. TRIP MIB

```
TRIP-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```
    MODULE-IDENTITY,  
    OBJECT-TYPE,  
    NOTIFICATION-TYPE,  
    Unsigned32,  
    Integer32,  
    Counter32,  
    mib-2  
        FROM SNMPv2-SMI                -- [RFC2578]
```

```
    DateAndTime,  
    TimeInterval,  
    TruthValue,  
    TimeStamp,  
    StorageType,  
    RowStatus  
        FROM SNMPv2-TC                -- [RFC2579]
```

```
    OBJECT-GROUP,  
    MODULE-COMPLIANCE,  
    NOTIFICATION-GROUP  
        FROM SNMPv2-CONF                -- [RFC2580]
```

```
    InetAddressType,  
    InetAddress,  
    InetPortNumber  
        FROM INET-ADDRESS-MIB          -- [RFC3291]
```

```
    applIndex,  
    applRFC2788Group  
        FROM NETWORK-SERVICES-MIB      -- [RFC2788]
```

```
    TripItad,  
    TripId,  
    TripAppProtocol,  
    TripAddressFamily,  
    TripCommunityId,  
    TripProtocolVersion,  
    TripSendReceiveMode  
        FROM TRIP-TC-MIB;                -- [RFC3872]
```

```
tripMIB MODULE-IDENTITY
```

```
    LAST-UPDATED "200409020000Z" -- Sep 02, 2004  
    ORGANIZATION "IETF IPTel Working Group.
```

Mailing list: iptel@lists.bell-labs.com"

CONTACT-INFO

"Co-editor David Zinman
 postal: 265 Ridley Blvd.
 Toronto ON, M5M 4N8
 Canada
 email: dzinman@rogers.com
 phone: +1 416 433 4298

Co-editor: David Walker
 Sedna Wireless Inc.
 postal: 495 March Road, Suite 500
 Ottawa, ON K2K 3G1
 Canada
 email: david.walker@sedna-wireless.com
 phone: +1 613 878 8142

Co-editor Jianping Jiang
 Synthesis Limited
 postal: 30 Fulton Way
 Richmond Hill, ON L4B 1J5
 Canada
 email: jjjiang@synthesis.com
 phone: +1 905 886-7818 x2515
 "

DESCRIPTION

"The MIB module describing Telephony Routing over IP (TRIP). TRIP is a policy driven inter-administrative domain protocol for advertising the reachability of telephony destinations between location servers (LS), and for advertising attributes of the routes to those destinations.

Copyright (C) The Internet Society (2004). This version of this MIB module is part of RFC 3872, see the RFC itself for full legal notices."

REVISION "200409020000Z" -- Sep 02, 2004

DESCRIPTION

"The initial version, Published as RFC 3872."

::= { mib-2 116 }

tripMIBNotifications OBJECT IDENTIFIER ::= { tripMIB 0 }
 tripMIBObjects OBJECT IDENTIFIER ::= { tripMIB 1 }
 tripMIBConformance OBJECT IDENTIFIER ::= { tripMIB 2 }
 tripMIBNotifObjects OBJECT IDENTIFIER ::= { tripMIB 3 }

```

tripMIBCompliances    OBJECT IDENTIFIER ::=
                        { tripMIBConformance 1 }
tripMIBGroups          OBJECT IDENTIFIER ::=
                        { tripMIBConformance 2 }

--
-- tripCfgTable
--
tripCfgTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF TripCfgEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This table contains the common configuration objects
        applicable to all TRIP applications referenced by the
        applIndex. Each row represents those objects for a
        particular TRIP LS present in this system. The
        instances of TRIP LS's are uniquely identified by the
        applIndex. The objects in this table SHOULD be
        nonVolatile and survive a reboot."
    ::= { tripMIBObjects 1 }

tripCfgEntry OBJECT-TYPE
    SYNTAX      TripCfgEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A row of common configuration."
    INDEX { applIndex }
    ::= { tripCfgTable 1 }

TripCfgEntry ::=
    SEQUENCE {
        tripCfgProtocolVersion      TripProtocolVersion,
        tripCfgItad                  TripItad,
        tripCfgIdentifier             TripId,
        tripCfgAdminStatus            INTEGER,
        tripCfgOperStatus             INTEGER,
        tripCfgAddrIAddrType          InetAddressType,
        tripCfgAddr                   InetAddress,
        tripCfgPort                    InetPortNumber,
        tripCfgMinItadOriginationInterval Unsigned32,
        tripCfgMinRouteAdvertisementInterval Unsigned32,
        tripCfgMaxPurgeTime            Unsigned32,
        tripCfgDisableTime             Unsigned32,
        tripCfgSendReceiveMode         TripSendReceiveMode,
        tripCfgStorage                 StorageType
    }

```

```
tripCfgProtocolVersion    OBJECT-TYPE
    SYNTAX      TripProtocolVersion
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This object will reflect the version of TRIP
        supported by this system.  It follows the same
        format as TRIP version information contained
        in the TRIP messages generated by this TRIP entity."
    REFERENCE
        "RFC 3219, section 4.2."
    ::= { tripCfgEntry 1 }

tripCfgItad    OBJECT-TYPE
    SYNTAX      TripItad
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The Internet Telephony Administrative domain (ITAD)
        of this LS."
    ::= { tripCfgEntry 2 }

tripCfgIdentifier    OBJECT-TYPE
    SYNTAX      TripId
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The object that identifies this TRIP Client."
    ::= { tripCfgEntry 3 }

tripCfgAdminStatus    OBJECT-TYPE
    SYNTAX      INTEGER {
                    up(1),
                    down(2)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The desired TRIP state.

        up(1)   : Set the application to normal operation.

        down(2): Set the application to a state where it will
                  not process TRIP messages.

        Setting this object should be reflected in
        tripCfgOperStatus.  If an unknown error occurs
        tripCfgOperStatus will return unknown(0)."
```

```
::= { tripCfgEntry 4 }
```

```
tripCfgOperStatus OBJECT-TYPE
```

```
SYNTAX      INTEGER {
                unknown(0),
                up(1),
                down(2),
                faulty(3)
            }
```

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

```
STATUS      current
```

```
DESCRIPTION
```

```
"The current operational state of the TRIP protocol.
```

```
    unknown(0): The operating status of the application is
                  unknown.
```

```
    up(1):      The application is operating normally, and
                  is ready to process (receive and issue) TRIP
                  requests and responses.
```

```
    down(2):    The application is currently not processing
                  TRIP messages. This occurs if the TRIP
                  application is in an initialization state or
                  if tripCfgAdminStatus is set to down(2).
```

```
    faulty(3):  The application is not operating normally due
                  to a fault in the system.
```

```
    If tripCfgAdminStatus is down(2) then tripOperStatus SHOULD
    be down(2). If tripAdminStatus is changed to up(1) then
    tripOperStatus SHOULD change to up(1) if there is no
    fault that prevents the TRIP protocol from moving to the
    up(1) state."
```

```
::= { tripCfgEntry 5 }
```

```
tripCfgAddrIAddrType OBJECT-TYPE
```

```
SYNTAX      InetAddressType
```

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

```
STATUS      current
```

```
DESCRIPTION
```

```
"The type of Inet Address of the tripAddr."
```

```
REFERENCE
```

```
"RFC 3291, section 3."
```

```
::= { tripCfgEntry 6 }
```

```
tripCfgAddr OBJECT-TYPE
```

```
SYNTAX      InetAddress
```

MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The network address of the local LS that the peer connects to. The type of address depends on the object tripCfgAddrIAddrType. The type of this address is determined by the value of the tripCfgAddrIAddrType object."
REFERENCE
 "RFC 3291, section 3."
 ::= { tripCfgEntry 7 }

tripCfgPort OBJECT-TYPE
SYNTAX InetPortNumber
MAX-ACCESS read-write
STATUS current
DESCRIPTION
 "The local tcp/udp port on the local LS that the peer connects to."
 ::= { tripCfgEntry 8 }

tripCfgMinItadOriginationInterval OBJECT-TYPE
SYNTAX Unsigned32 (1..2147483647)
UNITS "Seconds"
MAX-ACCESS read-write
STATUS current
DESCRIPTION
 "The minimum amount of time that MUST elapse between advertisement of the update message that reports changes within the LS's own ITAD."
DEFVAL { 30 }
 ::= { tripCfgEntry 9 }

tripCfgMinRouteAdvertisementInterval OBJECT-TYPE
SYNTAX Unsigned32 (1..2147483647)
UNITS "Seconds"
MAX-ACCESS read-write
STATUS current
DESCRIPTION
 "Specifies minimal interval between successive advertisements to a particular destination from an LS."
DEFVAL { 30 }
 ::= { tripCfgEntry 10 }

tripCfgMaxPurgeTime OBJECT-TYPE
SYNTAX Unsigned32 (1..2147483647)
UNITS "Seconds"
MAX-ACCESS read-write

```

STATUS      current
DESCRIPTION
    "Indicates the interval that the LS MUST maintain routes
    marked as withdrawn in its database."
DEFVAL { 10 }
 ::= { tripCfgEntry 11 }

```

```

tripCfgDisableTime OBJECT-TYPE
SYNTAX      Unsigned32 (1..2147483647)
UNITS       "Seconds"
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION
    "Indicates the interval that the TRIP module of the
    LS MUST be disabled while routes originated by this
    LS with high sequence numbers can be removed."
DEFVAL { 180 }
 ::= { tripCfgEntry 12 }

```

```

tripCfgSendReceiveMode OBJECT-TYPE
SYNTAX      TripSendReceiveMode
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "The operational mode of the TRIP entity running on this
    system."
 ::= { tripCfgEntry 13 }

```

```

tripCfgStorage OBJECT-TYPE
SYNTAX      StorageType
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION
    "The storage type for this conceptual row. Conceptual rows
    having the value 'permanent' need not allow write-access
    to any columnar objects in the row."
DEFVAL { nonVolatile }
 ::= { tripCfgEntry 14 }

```

```

--
-- TripRouteTypeTable
--

```

```

tripRouteTypeTable OBJECT-TYPE
SYNTAX      SEQUENCE OF TripRouteTypeEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION

```

"The TRIP peer Route Type table contains one entry per supported protocol - address family pair. The objects in this table are volatile and are refreshed after a reboot."
 ::= { tripMIBObjects 2 }

tripRouteTypeEntry OBJECT-TYPE

SYNTAX TripRouteTypeEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An entry containing information about the route type that a particular TRIP entity supports. Each entry represents information about either the local or a remote LS peer. The object tripRouteTypePeer is used to distinguish this. In the case of a local LS, the address/port information will reflect the values configured in tripCfgTable. In the case of a remote peer, the address/port information will reflect the values of an entry in the tripPeerTable.

Implementation need to be aware that if the size of tripRouteTypeAddr exceeds 111 sub-IDs, then OIDs of column instances in this table will have more than 128 sub-IDs and cannot be accessed using SNMPv1, SNMPv2c, or snmpv3."

INDEX { applIndex,
 tripRouteTypeAddrInetType,
 tripRouteTypeAddr,
 tripRouteTypePort,
 tripRouteTypeProtocolId,
 tripRouteTypeAddrFamilyId }
 ::= { tripRouteTypeTable 1 }

TripRouteTypeEntry ::= SEQUENCE {

tripRouteTypeAddrInetType	InetAddressType,
tripRouteTypeAddr	InetAddress,
tripRouteTypePort	InetPortNumber,
tripRouteTypeProtocolId	TripAppProtocol,
tripRouteTypeAddrFamilyId	TripAddressFamily,
tripRouteTypePeer	INTEGER

}

tripRouteTypeAddrInetType OBJECT-TYPE

SYNTAX InetAddressType

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The type of Inet Address of the tripRouteTypeAddr."

REFERENCE

```
        "RFC 3291, section 3."
 ::= { tripRouteTypeEntry 1 }

tripRouteTypeAddr OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "The network address of this entry's TRIP peer LS. The
         type of this address is determined by the value of the
         tripRouteTypeAddrInetType object."
    REFERENCE
        "RFC 3291, section 3."
 ::= { tripRouteTypeEntry 2 }

tripRouteTypePort OBJECT-TYPE
    SYNTAX      InetPortNumber
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "The port for the TCP connection between this and
         an associated TRIP peer."
 ::= { tripRouteTypeEntry 3 }

tripRouteTypeProtocolId OBJECT-TYPE
    SYNTAX      TripAppProtocol
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "The object identifier of a protocol that the associated
         peer is using."
 ::= { tripRouteTypeEntry 4 }

tripRouteTypeAddrFamilyId OBJECT-TYPE
    SYNTAX      TripAddressFamily
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "The object identifier of an address family that the
         associated peer belongs to."
 ::= { tripRouteTypeEntry 5 }

tripRouteTypePeer OBJECT-TYPE
    SYNTAX      INTEGER { local(1), remote(2) }
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "This object identifies whether this entry is
```

```

        associated with a 'local' or 'remote' LS peer."
 ::= { tripRouteTypeEntry 6 }

--
-- tripSupportedCommunityTable
--

tripSupportedCommunityTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF TripSupportedCommunityEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The list of TRIP communities that this LS supports. A
        TRIP community is a group of destinations that share
        common properties.

        The TRIP Supported Communities entry is used to group
        destinations so that the routing decision can be based
        on the identity of the group."
    REFERENCE
        "RFC 3219, section 5.9"
 ::= { tripMIBObjects 3 }

tripSupportedCommunityEntry OBJECT-TYPE
    SYNTAX      TripSupportedCommunityEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Entry containing information about a community. A TRIP
        community is a group of destinations that share some
        common property. This attribute is used so that routing
        decisions can be based on the identity of the group."
    INDEX { applIndex, tripSupportedCommunityId }
 ::= { tripSupportedCommunityTable 1 }

TripSupportedCommunityEntry ::= SEQUENCE {
    tripSupportedCommunityId      TripCommunityId,
    tripSupportedCommunityItad     TripItad,
    tripSupportedCommunityStorage  StorageType,
    tripSupportedCommunityRowStatus RowStatus
}

tripSupportedCommunityId OBJECT-TYPE
    SYNTAX      TripCommunityId
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The identifier of the supported Community."

```

```

        ::= { tripSupportedCommunityEntry 1 }

tripSupportedCommunityItad OBJECT-TYPE
    SYNTAX      TripItad
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        "The ITAD of the community."
    ::= { tripSupportedCommunityEntry 2 }

tripSupportedCommunityStorage OBJECT-TYPE
    SYNTAX      StorageType
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        "The storage type for this conceptual row. Conceptual
        rows having the value 'permanent' need not allow write-
        access to any columnar objects in the row. It is not a
        requirement that this storage be non volatile."
    DEFVAL { nonVolatile }
    ::= { tripSupportedCommunityEntry 3 }

tripSupportedCommunityRowStatus OBJECT-TYPE
    SYNTAX      RowStatus
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        "The row status of the entry. This object is REQUIRED
        to create or delete rows by a manager. A value for
        tripSupportedCommunityItad MUST be set for row creation
        to be successful. If the instance already exists for a
        particular applIndex, the row create operation will
        fail.

        The value of this object has no effect on whether
        other objects in this conceptual row can be modified."
    ::= { tripSupportedCommunityEntry 4 }

--
-- TripPeerTable
--
tripPeerTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF TripPeerEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "The TRIP peer table. This table contains one entry per
        TRIP peer, and information about the connection with

```

```

        the peer."
 ::= { tripMIBObjects 4 }

tripPeerEntry OBJECT-TYPE
    SYNTAX      TripPeerEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Entry containing information about the connection with
        a TRIP peer.

        Implementation need to be aware that if the size of
        tripPeerRemoteAddr exceeds 113 sub-IDs, then OIDs of
        column instances in this table will have more than 128
        sub-IDs and cannot be accessed using SNMPv1, SNMPv2c, or
        snmpv3."
    INDEX { applIndex,
            tripPeerRemoteAddrInetType,
            tripPeerRemoteAddr,
            tripPeerRemotePort }
    ::= {tripPeerTable 1}

TripPeerEntry ::= SEQUENCE {
    tripPeerRemoteAddrInetType      InetAddressType,
    tripPeerRemoteAddr              InetAddress,
    tripPeerRemotePort              InetPortNumber,
    tripPeerIdentifier              TripId,
    tripPeerState                   INTEGER,
    tripPeerAdminStatus             INTEGER,
    tripPeerNegotiatedVersion       TripProtocolVersion,
    tripPeerSendReceiveMode         TripSendReceiveMode,
    tripPeerRemoteItad              TripItad,
    tripPeerConnectRetryInterval    Unsigned32,
    tripPeerMaxRetryInterval        Unsigned32,
    tripPeerHoldTime                Unsigned32,
    tripPeerKeepAlive               Unsigned32,
    tripPeerHoldTimeConfigured      Unsigned32,
    tripPeerKeepAliveConfigured     Unsigned32,
    tripPeerMaxPurgeTime            Unsigned32,
    tripPeerDisableTime             Unsigned32,
    tripPeerLearned                 TruthValue,
    tripPeerStorage                 StorageType,
    tripPeerRowStatus               RowStatus
}

tripPeerRemoteAddrInetType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  not-accessible

```

STATUS current
DESCRIPTION
"The type of Inet Address of the tripPeerRemoteAddr."
REFERENCE
"RFC 3291, section 3."
::= { tripPeerEntry 1 }

tripPeerRemoteAddr OBJECT-TYPE
SYNTAX InetAddress
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"The IP address of this entry's TRIP peer LS. The type of
this address is determined by the value of the
tripPeerRemoteAddrInetType object."
REFERENCE
"RFC 3291, section 3."
::= { tripPeerEntry 2 }

tripPeerRemotePort OBJECT-TYPE
SYNTAX InetPortNumber
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"The remote port for the TCP connection between the
TRIP peers."
::= { tripPeerEntry 3 }

tripPeerIdentifier OBJECT-TYPE
SYNTAX TripId
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"TRIP identifier of the peer."
::= { tripPeerEntry 4 }

tripPeerState OBJECT-TYPE
SYNTAX INTEGER {
idle(1),
connect(2),
active(3),
openSent(4),
openConfirm(5),
established(6)
}
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"TRIP Peer Finite State Machine state.

idle(1) : The initial state. Local LS refuses all incoming connections. No application resources are allocated to processing information about the remote peer.

connect(2) : Local LS waiting for a transport protocol connection to be completed to the peer, and is listening for inbound transport connections from the peer.

active(3) : Local LS is listening for an inbound connection from the peer, but is not in the process of initiating a connection to the remote peer.

openSent(4) : Local LS has sent an OPEN message to its peer and is waiting for an OPEN message from the remote peer.

openConfirm(5): Local LS has sent an OPEN message to the remote peer, received an OPEN message from the remote peer, and sent a KEEPALIVE message in response to the OPEN. The local LS is now waiting for a KEEPALIVE message or a NOTIFICATION message in response to its OPEN message.

established(6): LS can exchange UPDATE, NOTIFICATION, and KEEPALIVE messages with its peer."

::= { tripPeerEntry 5 }

tripPeerAdminStatus OBJECT-TYPE

SYNTAX INTEGER {
up(1),
down(2)
}

MAX-ACCESS read-create
STATUS current

DESCRIPTION

"This object is used to affect the TRIP connection state.

up(1) : Allow a connection with the peer LS.

down(2) : disconnect the connection from the peer LS and do not allow any further connections to this

peer.

If this value is set to down(2) then tripPeerState will have the value of idle(1)."

DEFVAL { up }
::= { tripPeerEntry 6 }

tripPeerNegotiatedVersion OBJECT-TYPE

SYNTAX TripProtocolVersion

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The negotiated version of TRIP running between this local entity and this peer."

::= { tripPeerEntry 7 }

tripPeerSendReceiveMode OBJECT-TYPE

SYNTAX TripSendReceiveMode

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The operational mode of this peer."

::= { tripPeerEntry 8 }

tripPeerRemoteItad OBJECT-TYPE

SYNTAX TripItad

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The Internet Telephony Administrative domain of this peer."

::= { tripPeerEntry 9 }

tripPeerConnectRetryInterval OBJECT-TYPE

SYNTAX Unsigned32 (0..2147483647)

UNITS "Seconds"

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"Specifies the initial amount of time that will elapse between connection retry. This value SHOULD double after each attempt up to the value of tripPeerMaxRetryInterval. This value MUST always be less than or equal to the value of tripPeerMaxRetryInterval. Attempts to set this value higher than the max retry will not be allowed."

DEFVAL { 120 }
::= { tripPeerEntry 10 }

tripPeerMaxRetryInterval OBJECT-TYPE

SYNTAX Unsigned32 (0..2147483647)

UNITS "Seconds"

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"Specifies the maximum amount of time that will elapse between connection retries. Once the value of tripPeerConnectRetryInterval has reached this value, no more retries will be attempted. Attempts to set this value lower than the retry interval SHOULD not be allowed."

DEFVAL { 360 }

::= { tripPeerEntry 11 }

tripPeerHoldTime OBJECT-TYPE

SYNTAX Unsigned32 (1..2147483647)

UNITS "Seconds"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The time interval in seconds for the hold timer that is established with the peer. The value of this object is the smaller of the values in tripPeerHoldTimeConfigured and the hold time received in the open message."

::= { tripPeerEntry 12 }

tripPeerKeepAlive OBJECT-TYPE

SYNTAX Unsigned32 (1..2147483647)

UNITS "Seconds"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Specifies the amount of time that MUST elapse between keep alive messages. This value is negotiated with the remote when a connection is established."

::= { tripPeerEntry 13 }

tripPeerHoldTimeConfigured OBJECT-TYPE

SYNTAX Unsigned32 (0 | 3..65535)

UNITS "Seconds"

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"Specifies the maximum time that MAY elapse between the receipt of successive keepalive or update message. A value of 0 means that keepalive or update messages will not be

```
        sent."
    DEFVAL { 240 }
    ::= { tripPeerEntry 14 }

tripPeerKeepAliveConfigured OBJECT-TYPE
    SYNTAX      Unsigned32 (1..2147483647)
    UNITS       "Seconds"
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        "Specifies the amount of time that MUST elapse between
         keep alive messages."
    DEFVAL { 30 }
    ::= { tripPeerEntry 15 }

tripPeerMaxPurgeTime OBJECT-TYPE
    SYNTAX      Unsigned32 (1..65535)
    UNITS       "Seconds"
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        "Indicates the interval that the LS MUST maintain routes
         marked as withdrawn in its database."
    DEFVAL { 10 }
    ::= { tripPeerEntry 16 }

tripPeerDisableTime OBJECT-TYPE
    SYNTAX      Unsigned32 (1..65535)
    UNITS       "Seconds"
    MAX-ACCESS   read-create
    STATUS      current
    DESCRIPTION
        "Indicate the interval that the TRIP module of the remote
         peer LS MUST be disabled while routes originated by the
         local LS with high sequence numbers can be removed."
    DEFVAL { 180 }
    ::= { tripPeerEntry 17 }

tripPeerLearned OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "Indicates whether this entry was learned or
         configured."
    DEFVAL { false }
    ::= { tripPeerEntry 18 }
```

```

tripPeerStorage OBJECT-TYPE
    SYNTAX      StorageType
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION
        "The storage type for this conceptual row.  Conceptual
        rows having the value 'permanent' need not allow write-
        access to any columnar objects in the row.  It is not a
        requirement that this storage be non volatile."
    DEFVAL { nonVolatile }
    ::= { tripPeerEntry 19 }

tripPeerRowStatus OBJECT-TYPE
    SYNTAX      RowStatus
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION
        "The row status of the entry.  This object is REQUIRED to
        create or delete rows remotely by a manager.  If the
        instance already exists for a particular applIndex, the
        row create operation will fail.

        The value of this object has no effect on whether
        other objects in this conceptual row can be modified.

        Entries in this table can be learned by the TRIP
        application, or provisioned through this table."
    ::= { tripPeerEntry 20 }

--
-- TripPeerStatisticsTable
--

tripPeerStatisticsTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF TripPeerStatisticsEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
        "The TRIP peer stats table.  This table contains one
        entry per remote TRIP peer, and statistics related to the
        connection with the remote peer.  The objects in this
        table are volatile."
    ::= { tripMIBObjects 5 }

tripPeerStatisticsEntry OBJECT-TYPE
    SYNTAX      TripPeerStatisticsEntry
    MAX-ACCESS   not-accessible
    STATUS       current

```

DESCRIPTION

"Entry containing information about the connection with a TRIP peer."

AUGMENTS { tripPeerEntry }
 ::= { tripPeerStatisticsTable 1 }

```

TripPeerStatisticsEntry ::= SEQUENCE {
    tripPeerInUpdates          Counter32,
    tripPeerOutUpdates         Counter32,
    tripPeerInTotalMessages    Counter32,
    tripPeerOutTotalMessages   Counter32,
    tripPeerFsmEstablishedTransitions Counter32,
    tripPeerFsmEstablishedTime DateAndTime,
    tripPeerInUpdateElapsedTime TimeInterval,
    tripPeerStateChangeTime    TimeStamp
}

```

tripPeerInUpdates OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of TRIP update messages received from this remote peer since the last restart of this location server."

::= { tripPeerStatisticsEntry 1 }

tripPeerOutUpdates OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of TRIP update messages sent to this remote peer since the last restart of this LS."

::= { tripPeerStatisticsEntry 2 }

tripPeerInTotalMessages OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The total number of TRIP messages received from the remote peer on this connection since the last restart of this LS."

::= { tripPeerStatisticsEntry 3 }

tripPeerOutTotalMessages OBJECT-TYPE

SYNTAX Counter32

```

MAX-ACCESS    read-only
STATUS        current
DESCRIPTION
    "The total number of outgoing TRIP messages sent to the
    remote peer since the last restart of this LS."
 ::= { tripPeerStatisticsEntry 4 }

```

tripPeerFsmEstablishedTransitions OBJECT-TYPE

```

SYNTAX         Counter32
MAX-ACCESS    read-only
STATUS        current
DESCRIPTION
    "The number of times the remote peer has transitioned
    into the established state since the last restart of this
    LS."
 ::= { tripPeerStatisticsEntry 5 }

```

tripPeerFsmEstablishedTime OBJECT-TYPE

```

SYNTAX         DateAndTime
MAX-ACCESS    read-only
STATUS        current
DESCRIPTION
    "Indicates the time and date that this remote peer entered
    the 'established' state."
 ::= { tripPeerStatisticsEntry 6 }

```

tripPeerInUpdateElapsedTime OBJECT-TYPE

```

SYNTAX         TimeInterval
MAX-ACCESS    read-only
STATUS        current
DESCRIPTION
    "Elapsed time in hundredths of seconds since the last
    TRIP update message was received from this remote peer."
 ::= { tripPeerStatisticsEntry 7 }

```

tripPeerStateChangeTime OBJECT-TYPE

```

SYNTAX         TimeStamp
MAX-ACCESS    read-only
STATUS        current
DESCRIPTION
    "The value of sysUpTime when the last state change of
    tripPeerState took place."
 ::= { tripPeerStatisticsEntry 8 }

```

```

-- TRIP Received Route Table.  This table contains
-- all routes from all sources.  Each entry consists
-- of a route and its associated path attributes.

```

```

tripRouteTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF TripRouteEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The TRIP route table containing information about
        reachable routes that are to be added to service by the
        receiving LS. The objects in this table are volatile
        and are refreshed when this LS rediscovers its route
        table."
    ::= { tripMIBObjects 6 }

tripRouteEntry OBJECT-TYPE
    SYNTAX      TripRouteEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Information about a route to a called destination."
    INDEX { applIndex,
            tripRouteAppProtocol,
            tripRouteAddressFamily,
            tripRouteAddress,
            tripRoutePeer
          }
    ::= { tripRouteTable 1 }

TripRouteEntry ::= SEQUENCE {
    tripRouteAppProtocol          TripAppProtocol,
    tripRouteAddressFamily        TripAddressFamily,
    tripRouteAddress              OCTET STRING,
    tripRoutePeer                 TripId,
    tripRouteTRIBMask             BITS,
    tripRouteAddressSequenceNumber Unsigned32,
    tripRouteAddressOriginatorId  TripId,
    tripRouteNextHopServerIAddrType InetAddressType,
    tripRouteNextHopServer        InetAddress,
    tripRouteNextHopServerPort    InetPortNumber,
    tripRouteNextHopServerItad     TripItad,
    tripRouteMultiExitDisc        Unsigned32,
    tripRouteLocalPref            Unsigned32,
    tripRouteAdvertisementPath     OCTET STRING,
    tripRouteRoutedPath           OCTET STRING,
    tripRouteAtomicAggregate      TruthValue,
    tripRouteUnknown              OCTET STRING,
    tripRouteWithdrawn            TruthValue,
    tripRouteConverted            TruthValue,
    tripRouteReceivedTime         TimeStamp
}

```

```
tripRouteAppProtocol OBJECT-TYPE
    SYNTAX      TripAppProtocol
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The protocol for which this entry of the routing table
         is maintained."
    ::= { tripRouteEntry 1 }

tripRouteAddressFamily OBJECT-TYPE
    SYNTAX      TripAddressFamily
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Specifies the type of address for the destination
         route."
    ::= { tripRouteEntry 2 }

tripRouteAddress OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(1..105))
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This is the address (prefix) of the family type given
         by Address Family of the destination. It is the prefix
         of addresses reachable from this gateway via the next
         hop server. The SIZE value of 105 has been assigned due
         to the sub identifier of object types length limitation
         as defined in SMIV2."
    REFERENCE
        "RFC 3219, section 5.1.1.1."
    ::= { tripRouteEntry 3 }

tripRoutePeer OBJECT-TYPE
    SYNTAX      TripId
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The identifier of the peer where the route information
         was learned."
    ::= { tripRouteEntry 4 }

tripRouteTRIBMask OBJECT-TYPE
    SYNTAX      BITS {
        adjTribIns(0),
        extTrib(1),
        locTrib(2),
        adjTribOut(3)
```

```

    }
MAX-ACCESS    read-only
STATUS        current
DESCRIPTION
    "Indicates which Telephony Routing Information Base (TRIB)
    this entry belongs to. This is
    a bit-map of possible types. If the bit has a value of
    1, then the entry is a member of the corresponding TRIB
    type. If the bit has a value of 0 then the entry is not
    a member of the TRIP type. The various bit positions
    are:

```

0	adjTribIns	The entry is of type adj-TRIBs-ins, stores routing information that has been learned from inbound UPDATE messages.
1	extTrib	The entry is of type ext-TRIB, the best route for a given destination.
2	locTrib	The entry is of type loc-TRIB contains the local TRIP routing information that the LS has selected.
3	adjTribOut	The entry is of type adj-TRIBs-out, stores the information that the local LS has selected for advertisement to its external peers."

REFERENCE

"RFC 3291, section 3.5."

```
::= { tripRouteEntry 5 }
```

tripRouteAddressSequenceNumber OBJECT-TYPE

SYNTAX Unsigned32 (1..2147483647)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Indicates the version of the destination route originated by the LS identified by tripRouteAddressOriginatorId intra-domain attribute."

```
::= { tripRouteEntry 6 }
```

tripRouteAddressOriginatorId OBJECT-TYPE

SYNTAX TripId

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This is an intra-domain attribute indicating the internal LS that originated the route into the ITAD."

```
::= { tripRouteEntry 7 }
```

```
tripRouteNextHopServerIAAddrType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "The type of Inet Address of the tripRouteNextHopServer."
    REFERENCE
        "RFC 3291, section 3."
    ::= { tripRouteEntry 8 }

tripRouteNextHopServer OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "Indicates the next hop that messages of a given protocol
        destined for tripRouteAddress SHOULD be sent to. The type
        of this address is determined by the value of the
        tripRouteNextHopServerIAAddrType object."
    ::= { tripRouteEntry 9 }

tripRouteNextHopServerPort OBJECT-TYPE
    SYNTAX      InetPortNumber
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "The port of the next hop server that this route
        will use."
    ::= { tripRouteEntry 10 }

tripRouteNextHopServerItad OBJECT-TYPE
    SYNTAX      TripItad
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "Indicates the domain of the next hop."
    ::= { tripRouteEntry 11 }

tripRouteMultiExitDisc OBJECT-TYPE
    SYNTAX      Unsigned32 (0..4294967295)
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "The Multiple Exit Discriminator allows an LS to
        discriminate between, and indicate preference for,
        otherwise similar routes to a neighbouring domain.
        A higher value represents a more preferred routing
        object."
```

REFERENCE

"RFC 3219, section 5.8"

::= { tripRouteEntry 12 }

tripRouteLocalPref OBJECT-TYPE

SYNTAX Unsigned32 (0..4294967295)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Indicated the local LS's degree of preference for an advertised route destination."

REFERENCE

"RFC 3219, section 4.3.4.7"

::= { tripRouteEntry 13 }

tripRouteAdvertisementPath OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(4..252))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Identifies the sequence of domains through which this advertisement has passed."

This object is probably best represented as sequence of TripItads. For SMI compatibility, though, it is represented as an OCTET STRING. This object is a sequence of ITADs where each set of 4 octets corresponds to a TRIP ITAD in network byte order."

REFERENCE

"RFC 3219, section 4.3.4.4"

::= { tripRouteEntry 14 }

tripRouteRoutedPath OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(4..252))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Identifies the ITADs through which messages sent using this route would pass. These are a subset of tripRouteAdvertisementPath."

This object is probably best represented as sequence of TripItads. For SMI compatibility, though, it is represented as OCTET STRING. This object is a sequence of ITADs where each set of 4 octets corresponds to a TRIP ITAD in network byte order."

REFERENCE

"RFC 3219, section 4.3.4.5"

```
::= { tripRouteEntry 15 }
```

tripRouteAtomicAggregate OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Indicates that a route MAY traverse domains not listed in tripRouteRoutedPath. If an LS selects the less specific route from a set of overlapping routes, then this value returns TRUE."

REFERENCE

"RFC 3219, section 4.3.4.6"

```
::= { tripRouteEntry 16 }
```

tripRouteUnknown OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(0..255))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object contains one or more attributes that were not understood, and because they were transitive, were dropped during aggregation. They take the format of a triple <attribute type, attribute length, attribute value>, of variable length. If no attributes were dropped, this returns an OCTET STRING of size 0."

REFERENCE

"RFC 3219, sections 4.3.1, 4.3.2.3"

```
::= { tripRouteEntry 17 }
```

tripRouteWithdrawn OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Indicates if this route is to be removed from service by the receiving LS."

```
::= { tripRouteEntry 18 }
```

tripRouteConverted OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"Indicates if this route has been converted to a different application protocol than it had originally."

```
::= { tripRouteEntry 19 }
```

```

tripRouteReceivedTime OBJECT-TYPE
    SYNTAX          TimeStamp
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The value of sysUpTime when this route was received."
    ::= { tripRouteEntry 20 }

--
-- TRIP Received Route Community Table.
--

tripRouteCommunityTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF TripRouteCommunityEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "A table containing a list of TRIP communities associated
        with a route. Each instance of tripRouteTypeEntry that has
        the tripRouteTypePeer object set to remote(2) has an
        instance in the tripRouteTable as a parent. The objects
        in this table are volatile and are refreshed after a
        reboot."
    REFERENCE
        "RFC 3219, section 5.9."
    ::= { tripMIBObjects 7 }

tripRouteCommunityEntry OBJECT-TYPE
    SYNTAX          TripRouteCommunityEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "Information about communities associated with a route.
        An entry with a tripRouteAddress of 00 and a
        tripRoutePeer of 0 refers to the local LS."
    INDEX { applIndex,
            tripRouteAppProtocol,
            tripRouteAddressFamily,
            tripRouteAddress,
            tripRoutePeer,
            tripRouteCommunityId
          }
    ::= { tripRouteCommunityTable 1 }

TripRouteCommunityEntry ::= SEQUENCE {
    tripRouteCommunityId    TripCommunityId,
    tripRouteCommunityItad  TripItad
}

```

```

tripRouteCommunityId OBJECT-TYPE
    SYNTAX      TripCommunityId
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The community identifier."
    ::= { tripRouteCommunityEntry 1 }

tripRouteCommunityItad OBJECT-TYPE
    SYNTAX      TripItad
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The ITAD associated with this community."
    ::= { tripRouteCommunityEntry 2 }

--
-- tripItadTopologyTable
--

tripItadTopologyTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF TripItadTopologyEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The sequence of link connections between peers within an
        ITAD. The objects in this table are volatile and are
        refreshed after a reboot."
    ::= { tripMIBObjects 8 }

tripItadTopologyEntry OBJECT-TYPE
    SYNTAX      TripItadTopologyEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Information about a peer of the LS identified by
        tripItadTopologyOrigId."
    INDEX { applIndex, tripItadTopologyOrigId }
    ::= { tripItadTopologyTable 1 }

TripItadTopologyEntry ::= SEQUENCE {
    tripItadTopologyOrigId  TripId,
    tripItadTopologySeqNum  Unsigned32
}

tripItadTopologyOrigId OBJECT-TYPE
    SYNTAX      TripId
    MAX-ACCESS  not-accessible

```

```

        STATUS      current
        DESCRIPTION
            "Indicates the internal LS that originated the ITAD
            topology information into the ITAD."
        ::= { tripItadTopologyEntry 1 }

tripItadTopologySeqNum OBJECT-TYPE
    SYNTAX      Unsigned32 (1..2147483647)
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "Indicates the version of the ITAD topology originated
        by the LS identified by tripItadTopologyOrigId."
    ::= { tripItadTopologyEntry 2 }

--
-- tripItadTopologyIdTable
--

tripItadTopologyIdTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF TripItadTopologyIdEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "The list of other LS's within the ITAD domain that the
        LS identified by tripItadTopologyOrigId is currently
        peering. Each instance of tripItadTopologyIdEntry has an
        instance in the tripItadTopologyTable as a parent. The
        objects in this table are volatile and are refreshed
        after a reboot."
    ::= { tripMIBObjects 9 }

tripItadTopologyIdEntry OBJECT-TYPE
    SYNTAX      TripItadTopologyIdEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "Information about a peer to the LS identified by
        tripItadTopologyOrigId."
    INDEX { applIndex,
            tripItadTopologyOrigId,
            tripItadTopologyId }
    ::= { tripItadTopologyIdTable 1 }

TripItadTopologyIdEntry ::= SEQUENCE {
    tripItadTopologyId      TripId
}

```

```

tripItadTopologyId OBJECT-TYPE
    SYNTAX      TripId
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "The index into this entry. Indicates the other location
        servers within the ITAD domain that this LS identified
        by tripItadTopologyOrigId is currently peering."
    ::= { tripItadTopologyIdEntry 1 }

--
-- Notification objects
--

tripNotifApplIndex OBJECT-TYPE
    SYNTAX      Integer32 (1..2147483647)
    MAX-ACCESS   accessible-for-notify
    STATUS      current
    DESCRIPTION
        "This object contains the application Index. It is used
        to bind this notification with a specific instance of
        TRIP entity."
    REFERENCE
        "RFC 2788, section 3."
    ::= { tripMIBNotifObjects 1 }

tripNotifPeerAddrInetType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS   accessible-for-notify
    STATUS      current
    DESCRIPTION
        "The type of Inet Address of the tripNotifPeerAddr."
    REFERENCE
        "RFC 3291, section 3."
    ::= { tripMIBNotifObjects 2 }

tripNotifPeerAddr OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS   accessible-for-notify
    STATUS      current
    DESCRIPTION
        "The IP address of this entry's TRIP peer LS. This object
        contains the value of tripPeerRemoteAddr. The type of this
        address is determined by the value of the
        tripNotifPeerAddrInetType object."
    REFERENCE
        "RFC 3291, section 3."
    ::= { tripMIBNotifObjects 3 }

```

tripNotifPeerErrCode OBJECT-TYPE

```

SYNTAX      INTEGER {
                messageHeader(1),
                openMessage(2),
                updateMessage(3),
                holdTimerExpired(4),
                finiteStateMachine(5),
                cease(6),
                tripNotification(7)
            }

```

```

MAX-ACCESS  accessible-for-notify

```

```

STATUS      current

```

DESCRIPTION

"Notification message of TRIP error. The meaning of this value is applicable to the following functions:

messageHeader(1)

- All errors detected while processing the TRIP message header.

openMessage(2)

- All errors detected while processing the OPEN message.

updateMessage(3)

- All errors detected while processing the UPDATE message.

holdTimerExpired(4)

- A notification generated when the hold timer expires.

finiteStateMachine(5)

- All errors detected by the TRIP Finite State Machine.

cease(6)

- Any fatal error condition that the rest of the values do not cover.

tripNotification(7)

- Any error encountered while sending a notification message."

```

 ::= { tripMIBNotifObjects 4 }

```

tripNotifPeerErrSubcode OBJECT-TYPE

```

SYNTAX      Unsigned32 (1..2147483647)

```

```

MAX-ACCESS  accessible-for-notify

```

```

STATUS      current

```

DESCRIPTION

"The sub error code associated with error code. The

meaning of this value is dependent on the value of tripNotifPeerErrCode.

Message Header (1) Error Subcodes:

- 1 - Bad Message Length.
- 2 - Bad Message Type.

OPEN Message (2) Error Subcodes:

- 1 - Unsupported Version Number.
- 2 - Bad Peer ITAD.
- 3 - Bad TRIP Identifier.
- 4 - Unsupported Optional Parameter.
- 5 - Unacceptable Hold Time.
- 6 - Unsupported Capability.
- 7 - Capability Mismatch.

UPDATE Message (3) Error Subcodes:

- 1 - Malformed Attribute List.
- 2 - Unrecognized Well-known Attribute.
- 3 - Missing Well-known Mandatory Attribute.
- 4 - Attribute Flags Error.
- 5 - Attribute Length Error.
- 6 - Invalid Attribute."

::= { tripMIBNotifObjects 5 }

--

-- Notifications

--

tripConnectionEstablished NOTIFICATION-TYPE

OBJECTS { tripNotifApplIndex,
tripNotifPeerAddrInetType,
tripNotifPeerAddr
}

STATUS current

DESCRIPTION

"The TRIP Connection Established event is generated when the TRIP finite state machine enters the ESTABLISHED state."

::= { tripMIBNotifications 1 }

tripConnectionDropped NOTIFICATION-TYPE

OBJECTS { tripNotifApplIndex,
tripNotifPeerAddrInetType,
tripNotifPeerAddr
}

STATUS current

DESCRIPTION

"The TRIP Connection Dropped event is generated when the

TRIP finite state machine leaves the ESTABLISHED state."
 ::= { tripMIBNotifications 2 }

tripFSM NOTIFICATION-TYPE

OBJECTS { tripNotifApplIndex,
tripNotifPeerAddrInetType,
tripNotifPeerAddr,
tripNotifPeerErrCode,
tripNotifPeerErrSubcode,
tripPeerState
}

STATUS current

DESCRIPTION

"The trip FSM Event is generated when any error is
detected by the TRIP Finite State Machine."

::= { tripMIBNotifications 3 }

tripOpenMessageError NOTIFICATION-TYPE

OBJECTS { tripNotifApplIndex,
tripNotifPeerAddrInetType,
tripNotifPeerAddr,
tripNotifPeerErrCode,
tripNotifPeerErrSubcode,
tripPeerState
}

STATUS current

DESCRIPTION

"Errors detected while processing the OPEN message."

::= { tripMIBNotifications 4 }

tripUpdateMessageError NOTIFICATION-TYPE

OBJECTS { tripNotifApplIndex,
tripNotifPeerAddrInetType,
tripNotifPeerAddr,
tripNotifPeerErrCode,
tripNotifPeerErrSubcode,
tripPeerState
}

STATUS current

DESCRIPTION

"Errors detected while processing the UPDATE message."

::= { tripMIBNotifications 5 }

tripHoldTimerExpired NOTIFICATION-TYPE

OBJECTS { tripNotifApplIndex,
tripNotifPeerAddrInetType,
tripNotifPeerAddr,
tripNotifPeerErrCode,
}

```

        tripNotifPeerErrSubcode,
        tripPeerState
    }
    STATUS current
    DESCRIPTION
        "The system does not receive successive messages within
        the period specified by the negotiated Hold Time."
    ::= { tripMIBNotifications 6 }

tripConnectionCollision NOTIFICATION-TYPE
    OBJECTS { tripNotifApplIndex }
    STATUS current
    DESCRIPTION
        "A pair of LSs tried to simultaneously to establish a
        transport connection to each other."
    ::= { tripMIBNotifications 7 }

tripCease NOTIFICATION-TYPE
    OBJECTS { tripNotifApplIndex,
              tripNotifPeerAddrInetType,
              tripNotifPeerAddr,
              tripNotifPeerErrCode,
              tripNotifPeerErrSubcode,
              tripPeerState
            }
    STATUS current
    DESCRIPTION
        "A TRIP peer MAY choose at any given time to close its TRIP
        connection by sending this notification message. However,
        the Cease notification message MUST NOT be used when a
        fatal error occurs."

    ::= { tripMIBNotifications 8 }

tripNotificationErr NOTIFICATION-TYPE
    OBJECTS { tripNotifApplIndex }
    STATUS current
    DESCRIPTION
        "Generated if there is an error detected in a TRIP
        notification message sent with another cause. Note that
        the TRIP notification referred to in this object is not
        an SNMP notification, it is a specific message described
        in the TRIP specification."
    REFERENCE
        "RFC 3219, section 6.4."
    ::= { tripMIBNotifications 9 }

```

--

```
-- Compliance Statements
--
```

```
tripMIBFullCompliance MODULE-COMPLIANCE
```

```
STATUS current
```

```
DESCRIPTION
```

```
"The compliance statement for TRIP entities that
implement this MIB module in read-write mode, such
that it can be used for both monitoring and configuring
the TRIP entity.
```

```
There is one INDEX object that cannot be represented in
the form of OBJECT clauses in SMIV2, but for which there
is a compliance requirement, expressed in OBJECT clause
form in this description:
```

```
-- OBJECT      tripRouteTypeAddrInetType
-- SYNTAX      InetAddressType (ipv4(1), ipv6(2),
--                               ipv4z(3), ipv6z(4))
-- DESCRIPTION
--      This MIB requires support for global and
--      non-global ipv4 and ipv6 addresses.
--
-- OBJECT      tripRouteTypeAddr
-- SYNTAX      InetAddress (SIZE (4 | 8 | 16 | 20))
-- DESCRIPTION
--      This MIB requires support for global and
--      non-global IPv4 and IPv6 addresses.
--
"
```

```
MODULE -- this module
```

```
MANDATORY-GROUPS { tripConfigGroup,
                    tripPeerTableConfigGroup,
                    tripRouteGroup,
                    tripItadTopologyGroup,
                    tripPeerTableStatsGroup }
```

```
GROUP tripNotificationGroup
```

```
DESCRIPTION
```

```
"This group is OPTIONAL. A TRIP entity can choose not to
send any notifications. If this group is implemented,
the tripNotifObjectGroup MUST also be implemented."
```

```
GROUP tripNotifObjectGroup
```

```
DESCRIPTION
```

```
"This group is OPTIONAL. A TRIP entity can choose not to
send any notifications. If this group is implemented,
```

the tripNotificationGroup MUST also be implemented."

```
OBJECT      tripSupportedCommunityRowStatus
SYNTAX      RowStatus { active(1) }
WRITE-SYNTAX RowStatus { createAndGo(4), destroy(6) }
DESCRIPTION
    "Support for createAndWait and notInService is not
    required."
```

```
OBJECT      tripPeerRowStatus
SYNTAX      RowStatus { active(1) }
WRITE-SYNTAX RowStatus { createAndGo(4), destroy(6) }
DESCRIPTION
    "Support for createAndWait and notInService is not
    required."
```

```
MODULE NETWORK-SERVICES-MIB
    MANDATORY-GROUPS { applRFC2788Group }
```

```
::= { tripMIBCompliances 1 }
tripMIBReadOnlyCompliance MODULE-COMPLIANCE
STATUS      current
DESCRIPTION
    "The compliance statement for TRIP entities that
    implement this MIB module in read only mode. Such TRIP
    entities can then only be monitored, but not be
    configured via this MIB module."
```

In read-only mode, the manager will not be able to add, remove or modify rows to any table, however the TRIP application may modify, remove or add rows to a table.

There is one INDEX object that cannot be represented in the form of OBJECT clauses in SMIV2, but for which there is a compliance requirement, expressed in OBJECT clause form in this description:

```
-- OBJECT      tripRouteTypeAddrInetType
-- SYNTAX      InetAddressType (ipv4(1), ipv6(2),
--                               ipv4z(3), ipv6z(4))
-- DESCRIPTION
--     This MIB requires support for global and
--     non-global ipv4 and ipv6 addresses.
--
-- OBJECT      tripRouteTypeAddr
-- SYNTAX      InetAddress (SIZE (4 | 8 | 16 | 20))
-- DESCRIPTION
--     This MIB requires support for global and
```

```
--      non-global IPv4 and IPv6 addresses.
--
"

MODULE -- this module
    MANDATORY-GROUPS { tripConfigGroup,
                        tripPeerTableConfigGroup,
                        tripRouteGroup,
                        tripItadTopologyGroup,
                        tripPeerTableStatsGroup }

GROUP tripNotificationGroup
DESCRIPTION
    "This group is OPTIONAL. A TRIP entity can choose not to
    send any notifications. If this group is implemented,
    the tripNotifObjectGroup MUST also be implemented."

GROUP tripNotifObjectGroup
DESCRIPTION
    "This group is OPTIONAL. A TRIP entity can choose not to
    send any notifications. If this group is implemented,
    the tripNotificationGroup MUST also be implemented."
OBJECT      tripCfgItad
MIN-ACCESS  read-only
DESCRIPTION
    "Write access is not required."

OBJECT      tripCfgAdminStatus
MIN-ACCESS  not-accessible
DESCRIPTION
    "Object is not needed when implemented in read-only mode."

OBJECT      tripCfgPort
MIN-ACCESS  read-only
DESCRIPTION
    "Write access is not required."

OBJECT      tripCfgMinItadOriginationInterval
MIN-ACCESS  read-only
DESCRIPTION
    "Write access is not required."

OBJECT      tripCfgMinRouteAdvertisementInterval
MIN-ACCESS  read-only
DESCRIPTION
    "Write access is not required."

OBJECT      tripCfgMaxPurgeTime
```

MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT tripCfgDisableTime
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT tripCfgStorage
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT tripSupportedCommunityItad
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT tripSupportedCommunityStorage
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT tripSupportedCommunityRowStatus
SYNTAX RowStatus { active(1) }
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required, and active is the only status that needs to be supported."

OBJECT tripPeerAdminStatus
MIN-ACCESS not-accessible
DESCRIPTION
"Object is not needed when implemented in read-only mode."

OBJECT tripPeerConnectRetryInterval
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT tripPeerMaxRetryInterval
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT tripPeerHoldTimeConfigured
MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT tripPeerKeepAliveConfigured

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT tripPeerMaxPurgeTime

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT tripPeerDisableTime

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT tripPeerStorage

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT tripPeerRowStatus

SYNTAX RowStatus { active(1) }

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required, and active is the only status that needs to be supported."

MODULE NETWORK-SERVICES-MIB

MANDATORY-GROUPS { app1RFC2788Group }

::= { tripMIBCompliances 2 }

--

-- Object and event conformance groups

--

tripConfigGroup OBJECT-GROUP

OBJECTS {

tripCfgProtocolVersion,

tripCfgItad,

tripCfgIdentifier,

tripCfgOperStatus,

tripCfgAdminStatus,

tripCfgAddrIAddrType,

tripCfgAddr,

tripCfgPort,

```

        tripCfgMinItadOriginationInterval,
        tripCfgMinRouteAdvertisementInterval,
        tripCfgMaxPurgeTime,
        tripCfgDisableTime,
        tripCfgSendReceiveMode,
        tripCfgStorage,
        tripSupportedCommunityItad,
        tripSupportedCommunityStorage,
        tripRouteTypePeer,
        tripSupportedCommunityRowStatus
    }
    STATUS current
    DESCRIPTION
        "The global objects for configuring trip."
    ::= { tripMIBGroups 1 }

```

tripPeerTableConfigGroup OBJECT-GROUP

```

    OBJECTS {
        tripPeerIdentifier,
        tripPeerState,
        tripPeerAdminStatus,
        tripPeerNegotiatedVersion,
        tripPeerSendReceiveMode,
        tripPeerRemoteItad,
        tripPeerConnectRetryInterval,
        tripPeerMaxRetryInterval,
        tripPeerHoldTime,
        tripPeerKeepAlive,
        tripPeerHoldTimeConfigured,
        tripPeerKeepAliveConfigured,
        tripPeerMaxPurgeTime,
        tripPeerDisableTime,
        tripPeerLearned,
        tripPeerStorage,
        tripPeerRowStatus
    }

```

```

    STATUS current
    DESCRIPTION
        "The global objects for configuring the TRIP peer
        table."
    ::= { tripMIBGroups 2 }

```

tripPeerTableStatsGroup OBJECT-GROUP

```

    OBJECTS {
        tripPeerInUpdates,
        tripPeerOutUpdates,
        tripPeerInTotalMessages,

```

```

        tripPeerOutTotalMessages,
        tripPeerFsmEstablishedTransitions,
        tripPeerFsmEstablishedTime,
        tripPeerInUpdateElapsedTime,
        tripPeerStateChangeTime
    }
    STATUS current
    DESCRIPTION
        "The global statistics the TRIP peer table."
    ::= { tripMIBGroups 3 }

tripRouteGroup OBJECT-GROUP
    OBJECTS {
        tripRouteTRIBMask,
        tripRouteAddressSequenceNumber,
        tripRouteAddressOriginatorId,
        tripRouteNextHopServerIAddrType,
        tripRouteNextHopServer,
        tripRouteNextHopServerPort,
        tripRouteNextHopServerItad,
        tripRouteMultiExitDisc,
        tripRouteLocalPref,
        tripRouteAdvertisementPath,
        tripRouteRoutedPath,
        tripRouteAtomicAggregate,
        tripRouteUnknown,
        tripRouteWithdrawn,
        tripRouteConverted,
        tripRouteReceivedTime,
        tripRouteCommunityItad
    }

    STATUS current
    DESCRIPTION
        "The global objects for configuring route attribute."
    ::= { tripMIBGroups 4 }

tripItadTopologyGroup OBJECT-GROUP
    OBJECTS {
        tripItadTopologySeqNum,
        tripItadTopologyId
    }

    STATUS current
    DESCRIPTION
        "The objects that define the TRIP ITAD topology."
    ::= { tripMIBGroups 5 }

tripNotificationGroup NOTIFICATION-GROUP
    NOTIFICATIONS {

```

```

        tripConnectionEstablished,
        tripConnectionDropped,
        tripFSM,
        tripOpenMessageError,
        tripUpdateMessageError,
        tripHoldTimerExpired,
        tripConnectionCollision,
        tripCease,
        tripNotificationErr
    }
    STATUS current
    DESCRIPTION
        "A collection of notifications defined for TRIP."
    ::= { tripMIBGroups 6 }

tripNotifObjectGroup OBJECT-GROUP
    OBJECTS {
        tripNotifApplIndex,
        tripNotifPeerAddrInetType,
        tripNotifPeerAddr,
        tripNotifPeerErrCode,
        tripNotifPeerErrSubcode
    }
    STATUS current
    DESCRIPTION
        "The collection of objects that specify information for
        TRIP notifications."
    ::= { tripMIBGroups 7 }

END

```

7. Security Considerations

The managed objects in this MIB module contain sensitive information since, collectively, they allow tracing and influencing of connections in TRIP devices and provide information of their connection characteristics. As such, improper manipulation of the objects represented by this MIB module MAY result in denial of service to a large number of available routes.

There are a number of management objects defined in this MIB module 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. These objects include:

tripCfgItad:

Improper setting of tripCfgItad value can make all peer connections drop and not be re-established.

tripCfgAdminStatus:

Improper setting of tripCfgAdminStatus from up to down will cause the TRIP Location Server stop processing TRIP messages.

tripCfgPort:

Improper setting of tripCfgPort can cause the failure of a peer establishing a connection.

**tripCfgMinItadOriginationInterval,
tripCfgMinRouteAdvertisementInterval:**

Improper configuration of these values MAY adversely affect local and global convergence of the routes advertised by this TRIP Location Server.

tripPeerAdminStatus:

Improper setting of tripPeerAdminStatus from up to down can cause significant disruption of the connectivity to the destination via the applicable remote TRIP Location Server peer.

tripPeerConnectRetryInterval,tripPeerMaxRetryInterval:

Improper configuration of these values can cause connections to be disrupted for extremely long time periods when otherwise they would be restored in a relatively short period of time.

tripPeerHoldTimeConfigured, tripPeerKeepAliveConfigured:

Improper configuration of these value can make TRIP peer sessions more fragile and less resilient to denial of service attacks.

There are a number of managed objects in this MIB module that contain sensitive information regarding the operation of a network. For example, a TRIP Location Server peer's local and remote addresses might be sensitive for ISPs who want to keep interface addresses on TRIP Location Server confidential so as to prevent TRIP Location Server addresses used for a denial of service attack or address spoofing.

Therefore, it is thus important to control even GET access to these objects and possibly to even encrypt the values of these object when sending them over the network via SNMP. Not all versions of SNMP provide features for such a secure environment.

SNMPv1 by itself is not a secure environment. Even if the network itself is secure (for example by using IPSec), even then, 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 module.

It is RECOMMENDED that the implementers consider the security features as provided by the SNMPv3 framework (see [RFC3410], section 8), including full support for the SNMPv3 cryptographic mechanisms (for authentication and privacy).

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module 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.

8. References

8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2579] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Textual Conventions for SMIV2", STD 58, RFC 2579, April 1999.
- [RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIV2", STD 58, RFC 2580, April 1999.
- [RFC2788] Freed, N. and S. Kille, "Network Services Monitoring MIB", RFC 2788, March 2000.
- [RFC3219] Rosenberg, J., Salama, H., and M. Squire, "Telephony Routing over IP (TRIP)", RFC 3219, January 2002.
- [RFC3291] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 3291, May 2002.
- [RFC2578] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Structure of Management Information Version 2 (SMIV2)", STD 58, RFC 2578, April 1999.

8.2. Informative References

- [RFC1657] Willis, S., Burruss, J., and J. Chu, Ed., "Definitions of Managed Objects for the Fourth Version of the Border Gateway Protocol (BGP-4) using SMIV2", RFC 1657, July 1994.
- [RFC1771] Rekhter, Y. and T. Li, "Border Gateway Protocol 4 (BGP-4)", RFC 1771, March 1995.
- [RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

9. Acknowledgments

The authors wish to thank Bert Wijnen, Dan Romascanu, and Jonathan Rosenberg for their insightful comments and suggestions.

Thanks to Kevin Lingle for his invaluable comments, help with MIB things and great ideas.

10. Authors' Addresses

David Zinman
Editor
265 Ridley Blvd
Toronto ON M5M 4N8
Canada

Phone: +1 416 433 4298
Email: dzinman@rogers.com

David Walker
Sedna Wireless Inc.
495 March Road, Suite 500
Ottawa, ON K2K 3G1
Canada

Phone: +1 613 878 8142
Email: david.walker@sedna-wireless.com

Jianping Jiang
Syndesis Limited
30 Fulton Way
Richmond Hill, ON L4B 1J5
Canada

Phone: +1 905 886-7818 x2515
Email: jjjiang@syndesis.com

11. Full Copyright Statement

Copyright (C) The Internet Society (2004).

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/S HE 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 IETF's procedures with respect to rights in IETF 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.

