                            IP Tunnel MIB

Status of This Memo

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

Abstract

   This memo defines a Management Information Base (MIB) module for use
   with network management protocols in the Internet community. In
   particular, it describes managed objects used for managing tunnels of
   any type over IPv4 and IPv6 networks. Extension MIB modules may be
   designed for managing protocol-specific objects. Likewise, extension
   MIB modules may be designed for managing security-specific objects.
   This MIB module does not support tunnels over non-IP networks.
   Management of such tunnels may be supported by other MIB modules.
   This memo obsoletes RFC 2667.

1. Introduction

   Over the past several years, there has been a number of "tunneling"
   protocols specified by the IETF (see [RFC1241] for an early
   discussion of the model and examples). This document describes a
   Management Information Base (MIB) module used for managing tunnels of
   any type over IPv4 and IPv6 networks, including Generic Routing
   Encapsulation (GRE) [RFC1701,RFC1702], IP-in-IP [RFC2003], Minimal
   Encapsulation [RFC2004], Layer 2 Tunneling Protocol (L2TP) [RFC2661],
   Point-to-Point Tunneling Protocol (PPTP) [RFC2637], Layer 2
   Forwarding (L2F) [RFC2341], UDP (e.g., [RFC1234]), Ascend Tunnel
   Management Protocol (ATMP) [RFC2107], and IPv6-in-IPv4 [RFC2893]
   tunnels, among others.

   Extension MIB modules may be designed for managing protocol-specific
   objects. Likewise, extension MIB modules may be designed for
   managing security-specific objects (e.g., IPsec [RFC2401]), and
   traffic conditioner [RFC2474] objects.

2. The Internet-Standard Management Framework

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

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

3. Overview

   This MIB module contains two current tables and one deprecated table.
   The current tables are:

   o   the Tunnel Interface Table, containing information on the tunnels
       known to a router; and

   o   the Tunnel Inet Config Table, which can be used for dynamic
       creation of tunnels, and also provides a mapping from endpoint
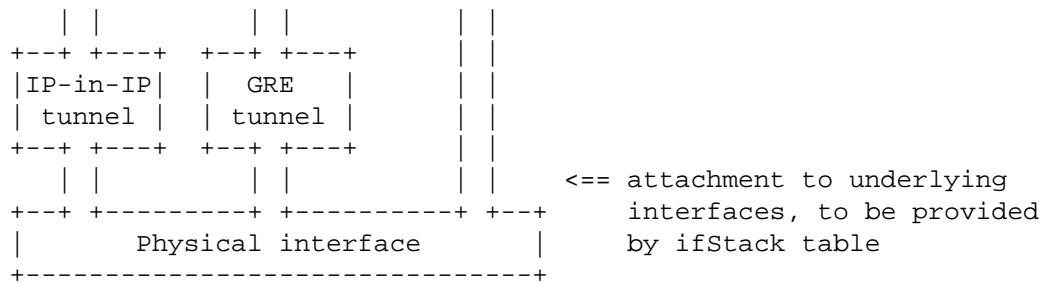   addresses to the current interface index value.

   The version of this MIB module that appeared in RFC 2667 contained
   the Tunnel Config Table, which mapped IPv4 endpoint addresses to
   interface indexes. It is now deprecated in favor of the Tunnel Inet
   Config Table.

3.1. Relationship to the Interfaces MIB

   This section clarifies the relationship of this MIB module to the
   Interfaces MIB [RFC2863]. Several areas of correlation are addressed
   in the following subsections. The implementor is referred to the
   Interfaces MIB document in order to understand the general intent of
   these areas.

3.1.1. Layering Model

   Each logical interface (physical or virtual) has an ifEntry in the
   Interfaces MIB [RFC2863]. Tunnels are handled by creating a logical
   interface (ifEntry) for each tunnel. These are then correlated,
   using the ifStack table of the Interfaces MIB, to those interfaces on
   which the local IPv4 or IPv6 addresses of the tunnels are configured.
   The basic model, therefore, looks something like this (for example):

```
      | |         | |                 | |
   +--+ +---+   +--+ +---+            | |
   |IP-in-IP|   |   GRE   |           | |
   | tunnel |   | tunnel |            | |
   +--+ +---+   +--+ +---+            | |
      | |         | |                 | |     <== attachment to underlying
   +--+ +---------+ +----------+ +--+     interfaces, to be provided
   |         Physical interface     |      by ifStack table
   +-------------------------------+
```

3.1.2. ifRcvAddressTable

   The ifRcvAddressTable usage can be defined in the MIB modules
   defining the encapsulation below the network layer, and holds the
   local IP addresses on which decapsulation will occur. For example,
   if IP-in-IP encapsulation is being used, the ifRcvAddressTable can be
   defined by IP-in-IP. If it is not specified, the default is that one
   entry will exist for the tunnel interface, where ifRcvAddressAddress
   contains the local IP address used for encapsulation/decapsulation
   (i.e., tunnelIfLocalInetAddress in the Tunnel Interface Table).

3.1.3. ifEntry

   IfEntries are defined in the MIB modules defining the encapsulation
   below the network layer. For example, if IP-in-IP encapsulation [20]
   is being used, the ifEntry is defined by IP-in-IP.

   The ifType of a tunnel should be set to "tunnel" (131). An entry in
   the IP Tunnel MIB module will exist for every ifEntry with this
   ifType. An implementation of the IP Tunnel MIB module may allow
   ifEntries to be created via the tunnelConfigTable. Creating a tunnel
   will also add an entry in the ifTable and in the tunnelIfTable, and
   deleting a tunnel will likewise delete the entry in the ifTable and
   the tunnelIfTable.

   The use of two different tables in this MIB module was an important
   design decision. Traditionally, ifIndex values are chosen by agents,
   and are permitted to change across restarts. Allowing row creation
   directly in the Tunnel Interface Table, indexed by ifIndex, would

complicate row creation and/or cause interoperability problems (if
each agent had special restrictions on ifIndex). Instead, a separate
table is used that is indexed only by objects over which the manager
has control. Namely, these are the addresses of the tunnel endpoints
and the encapsulation protocol. Finally, an additional manager-
chosen ID is used in the index to support protocols such as L2F which
allow multiple tunnels between the same endpoints.

4. Definitions

    TUNNEL-MIB DEFINITIONS ::= BEGIN

    IMPORTS
        MODULE-IDENTITY, OBJECT-TYPE, transmission,
        Integer32, IpAddress   FROM SNMPv2-SMI   -- [RFC2578]

        RowStatus, StorageType FROM SNMPv2-TC   -- [RFC2579]

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

        InetAddressType,
        InetAddress   FROM INET-ADDRESS-MIB   -- [RFC4001]

        IPv6FlowLabelOrAny   FROM IPV6-FLOW-LABEL-MIB -- [RFC3595]

        ifIndex,
        InterfaceIndexOrZero   FROM IF-MIB   -- [RFC2863]

        IANAtunnelType   FROM IANAifType-MIB;   -- [IFTYPE]

    tunnelMIB MODULE-IDENTITY
        LAST-UPDATED "200505160000Z" -- May 16, 2005
        ORGANIZATION "IETF IP Version 6 (IPv6) Working Group"
        CONTACT-INFO
                " Dave Thaler
                  Microsoft Corporation
                  One Microsoft Way
                  Redmond, WA 98052-6399
                  EMail: dthaler@microsoft.com"
        DESCRIPTION
                "The MIB module for management of IP Tunnels,
                independent of the specific encapsulation scheme in
                use.

                Copyright (C) The Internet Society (2005). This
                version of this MIB module is part of RFC 4087; see
                the RFC itself for full legal notices."

        REVISION   "200505160000Z" -- May 16, 2005
        DESCRIPTION
                "IPv4-specific objects were deprecated, including
                tunnelIfLocalAddress, tunnelIfRemoteAddress, the
                tunnelConfigTable, and the tunnelMIBBasicGroup.

                Added IP version-agnostic objects that should be used
                instead, including tunnelIfAddressType,
                tunnelIfLocalInetAddress, tunnelIfRemoteInetAddress,
                the tunnelInetConfigTable, and the
                tunnelIMIBInetGroup.

                The new tunnelIfLocalInetAddress and
                tunnelIfRemoteInetAddress objects are read-write,
                rather than read-only.

                Updated DESCRIPTION clauses of existing version-
                agnostic objects (e.g., tunnelIfTOS) that contained
                IPv4-specific text to cover IPv6 as well.

                Added tunnelIfFlowLabel for tunnels over IPv6.

                The encapsulation method was previously an INTEGER
                type, and is now an IANA-maintained textual
                convention.

                Published as RFC 4087."
        REVISION   "199908241200Z" -- August 24, 1999
        DESCRIPTION
                "Initial version, published as RFC 2667."
        ::= { transmission 131 }

    tunnelMIBObjects OBJECT IDENTIFIER ::= { tunnelMIB 1 }

    tunnel   OBJECT IDENTIFIER ::= { tunnelMIBObjects 1 }

    -- the IP Tunnel MIB-Group
    --
    -- a collection of objects providing information about
    -- IP Tunnels

    tunnelIfTable OBJECT-TYPE
        SYNTAX   SEQUENCE OF TunnelIfEntry
        MAX-ACCESS not-accessible
        STATUS   current
        DESCRIPTION
                "The (conceptual) table containing information on
                configured tunnels."

```
      ::= { tunnel 1 }

   tunnelIfEntry OBJECT-TYPE
       SYNTAX     TunnelIfEntry
       MAX-ACCESS not-accessible
       STATUS     current
       DESCRIPTION
               "An entry (conceptual row) containing the information
               on a particular configured tunnel."
       INDEX     { ifIndex }
       ::= { tunnelIfTable 1 }

   TunnelIfEntry ::= SEQUENCE {
       tunnelIfLocalAddress            IpAddress,   -- deprecated
       tunnelIfRemoteAddress           IpAddress,   -- deprecated
       tunnelIfEncapsMethod            IANAtunnelType,
       tunnelIfHopLimit                Integer32,
       tunnelIfSecurity                INTEGER,
       tunnelIfTOS                     Integer32,
       tunnelIfFlowLabel               IPv6FlowLabelOrAny,
       tunnelIfAddressType             InetAddressType,
       tunnelIfLocalInetAddress        InetAddress,
       tunnelIfRemoteInetAddress       InetAddress,
       tunnelIfEncapsLimit             Integer32
   }

   tunnelIfLocalAddress OBJECT-TYPE
       SYNTAX     IpAddress
       MAX-ACCESS read-only
       STATUS     deprecated
       DESCRIPTION
               "The address of the local endpoint of the tunnel
               (i.e., the source address used in the outer IP
               header), or 0.0.0.0 if unknown or if the tunnel is
               over IPv6.

               Since this object does not support IPv6, it is
               deprecated in favor of tunnelIfLocalInetAddress."
       ::= { tunnelIfEntry 1 }

   tunnelIfRemoteAddress OBJECT-TYPE
       SYNTAX     IpAddress
       MAX-ACCESS read-only
       STATUS     deprecated
       DESCRIPTION
               "The address of the remote endpoint of the tunnel
               (i.e., the destination address used in the outer IP
               header), or 0.0.0.0 if unknown, or an IPv6 address, or
```

                    the tunnel is not a point-to-point link (e.g., if it
                    is a 6to4 tunnel).

                    Since this object does not support IPv6, it is
                    deprecated in favor of tunnelIfRemoteInetAddress."
            ::= { tunnelIfEntry 2 }

        tunnelIfEncapsMethod OBJECT-TYPE
            SYNTAX   IANAtunnelType
            MAX-ACCESS read-only
            STATUS   current
            DESCRIPTION
                    "The encapsulation method used by the tunnel."
            ::= { tunnelIfEntry 3 }

        tunnelIfHopLimit OBJECT-TYPE
            SYNTAX   Integer32 (0 | 1..255)
            MAX-ACCESS read-write
            STATUS   current
            DESCRIPTION
                    "The IPv4 TTL or IPv6 Hop Limit to use in the outer IP
                    header. A value of 0 indicates that the value is
                    copied from the payload's header."
            ::= { tunnelIfEntry 4 }

        tunnelIfSecurity OBJECT-TYPE
            SYNTAX   INTEGER {
                        none(1),   -- no security
                        ipsec(2), -- IPsec security
                        other(3)
                    }
            MAX-ACCESS read-only
            STATUS   current
            DESCRIPTION
                    "The method used by the tunnel to secure the outer IP
                    header. The value ipsec indicates that IPsec is used
                    between the tunnel endpoints for authentication or
                    encryption or both. More specific security-related
                    information may be available in a MIB module for the
                    security protocol in use."
            ::= { tunnelIfEntry 5 }

        tunnelIfTOS OBJECT-TYPE
            SYNTAX   Integer32 (-2..63)
            MAX-ACCESS read-write
            STATUS   current
            DESCRIPTION
                    "The method used to set the high 6 bits (the

```
                   differentiated services codepoint) of the IPv4 TOS or
                   IPv6 Traffic Class in the outer IP header.  A value of
                   -1 indicates that the bits are copied from the
                   payload's header.  A value of -2 indicates that a
                   traffic conditioner is invoked and more information
                   may be available in a traffic conditioner MIB module.
                   A value between 0 and 63 inclusive indicates that the
                   bit field is set to the indicated value.

                   Note: instead of the name tunnelIfTOS, a better name
                   would have been tunnelIfDSCPMethod, but the existing
                   name appeared in RFC 2667 and existing objects cannot
                   be renamed."
         ::= { tunnelIfEntry 6 }

   tunnelIfFlowLabel OBJECT-TYPE
       SYNTAX      IPv6FlowLabelOrAny
       MAX-ACCESS read-write
       STATUS      current
       DESCRIPTION
                   "The method used to set the IPv6 Flow Label value.
                   This object need not be present in rows where
                   tunnelIfAddressType indicates the tunnel is not over
                   IPv6.  A value of -1 indicates that a traffic
                   conditioner is invoked and more information may be
                   available in a traffic conditioner MIB.  Any other
                   value indicates that the Flow Label field is set to
                   the indicated value."
         ::= { tunnelIfEntry 7 }

   tunnelIfAddressType OBJECT-TYPE
       SYNTAX      InetAddressType
       MAX-ACCESS read-write
       STATUS      current
       DESCRIPTION
                   "The type of address in the corresponding
                   tunnelIfLocalInetAddress and tunnelIfRemoteInetAddress
                   objects."
         ::= { tunnelIfEntry 8 }

   tunnelIfLocalInetAddress OBJECT-TYPE
       SYNTAX      InetAddress
       MAX-ACCESS read-write
       STATUS      current
       DESCRIPTION
                   "The address of the local endpoint of the tunnel
                   (i.e., the source address used in the outer IP
                   header).  If the address is unknown, the value is
```

```
                  0.0.0.0 for IPv4 or :: for IPv6.  The type of this
                  object is given by tunnelIfAddressType."
           ::= { tunnelIfEntry 9 }

       tunnelIfRemoteInetAddress OBJECT-TYPE
           SYNTAX      InetAddress
           MAX-ACCESS read-write
           STATUS      current
           DESCRIPTION
                  "The address of the remote endpoint of the tunnel
                  (i.e., the destination address used in the outer IP
                  header).  If the address is unknown or the tunnel is
                  not a point-to-point link (e.g., if it is a 6to4
                  tunnel), the value is 0.0.0.0 for tunnels over IPv4 or
                  :: for tunnels over IPv6.  The type of this object is
                  given by tunnelIfAddressType."
           ::= { tunnelIfEntry 10 }

       tunnelIfEncapsLimit OBJECT-TYPE
           SYNTAX      Integer32 (-1 | 0..255)
           MAX-ACCESS read-write
           STATUS      current
           DESCRIPTION
                  "The maximum number of additional encapsulations
                  permitted for packets undergoing encapsulation at this
                  node.  A value of -1 indicates that no limit is
                  present (except as a result of the packet size)."
           REFERENCE  "RFC 2473, section 4.1.1"
           ::= { tunnelIfEntry 11 }

       tunnelConfigTable OBJECT-TYPE
           SYNTAX      SEQUENCE OF TunnelConfigEntry
           MAX-ACCESS not-accessible
           STATUS      deprecated
           DESCRIPTION
                  "The (conceptual) table containing information on
                  configured tunnels.  This table can be used to map a
                  set of tunnel endpoints to the associated ifIndex
                  value.  It can also be used for row creation.  Note
                  that every row in the tunnelIfTable with a fixed IPv4
                  destination address should have a corresponding row in
                  the tunnelConfigTable, regardless of whether it was
                  created via SNMP.

                  Since this table does not support IPv6, it is
                  deprecated in favor of tunnelInetConfigTable."
           ::= { tunnel 2 }
```

```
   tunnelConfigEntry OBJECT-TYPE
       SYNTAX      TunnelConfigEntry
       MAX-ACCESS not-accessible
       STATUS      deprecated
       DESCRIPTION
               "An entry (conceptual row) containing the information
               on a particular configured tunnel.

               Since this entry does not support IPv6, it is
               deprecated in favor of tunnelInetConfigEntry."
       INDEX      { tunnelConfigLocalAddress,
                     tunnelConfigRemoteAddress,
                     tunnelConfigEncapsMethod,
                     tunnelConfigID }
       ::= { tunnelConfigTable 1 }

   TunnelConfigEntry ::= SEQUENCE {
       tunnelConfigLocalAddress          IpAddress,
       tunnelConfigRemoteAddress         IpAddress,
       tunnelConfigEncapsMethod          IANAtunnelType,
       tunnelConfigID                    Integer32,
       tunnelConfigIfIndex               InterfaceIndexOrZero,
       tunnelConfigStatus                RowStatus
   }

   tunnelConfigLocalAddress OBJECT-TYPE
       SYNTAX      IpAddress
       MAX-ACCESS not-accessible
       STATUS      deprecated
       DESCRIPTION
               "The address of the local endpoint of the tunnel, or
               0.0.0.0 if the device is free to choose any of its
               addresses at tunnel establishment time.

               Since this object does not support IPv6, it is
               deprecated in favor of tunnelInetConfigLocalAddress."
       ::= { tunnelConfigEntry 1 }

   tunnelConfigRemoteAddress OBJECT-TYPE
       SYNTAX      IpAddress
       MAX-ACCESS not-accessible
       STATUS      deprecated
       DESCRIPTION
               "The address of the remote endpoint of the tunnel.

               Since this object does not support IPv6, it is
               deprecated in favor of tunnelInetConfigRemoteAddress."
       ::= { tunnelConfigEntry 2 }
```

    tunnelConfigEncapsMethod OBJECT-TYPE
        SYNTAX   IANAtunnelType
        MAX-ACCESS not-accessible
        STATUS   deprecated
        DESCRIPTION
                "The encapsulation method used by the tunnel.

                Since this object does not support IPv6, it is
                deprecated in favor of tunnelInetConfigEncapsMethod."
        ::= { tunnelConfigEntry 3 }

    tunnelConfigID OBJECT-TYPE
        SYNTAX   Integer32 (1..2147483647)
        MAX-ACCESS not-accessible
        STATUS   deprecated
        DESCRIPTION
                "An identifier used to distinguish between multiple
                tunnels of the same encapsulation method, with the
                same endpoints. If the encapsulation protocol only
                allows one tunnel per set of endpoint addresses (such
                as for GRE or IP-in-IP), the value of this object is
                1. For encapsulation methods (such as L2F) which
                allow multiple parallel tunnels, the manager is
                responsible for choosing any ID which does not
                conflict with an existing row, such as choosing a
                random number.

                Since this object does not support IPv6, it is
                deprecated in favor of tunnelInetConfigID."
        ::= { tunnelConfigEntry 4 }

    tunnelConfigIfIndex OBJECT-TYPE
        SYNTAX   InterfaceIndexOrZero
        MAX-ACCESS read-only
        STATUS   deprecated
        DESCRIPTION
                "If the value of tunnelConfigStatus for this row is
                active, then this object contains the value of ifIndex
                corresponding to the tunnel interface. A value of 0
                is not legal in the active state, and means that the
                interface index has not yet been assigned.

                Since this object does not support IPv6, it is
                deprecated in favor of tunnelInetConfigIfIndex."
        ::= { tunnelConfigEntry 5 }

    tunnelConfigStatus OBJECT-TYPE
        SYNTAX   RowStatus

          MAX-ACCESS read-create
          STATUS   deprecated
          DESCRIPTION
                  "The status of this row, by which new entries may be
                  created, or old entries deleted from this table. The
                  agent need not support setting this object to
                  createAndWait or notInService since there are no other
                  writable objects in this table, and writable objects
                  in rows of corresponding tables such as the
                  tunnelIfTable may be modified while this row is
                  active.

                  To create a row in this table for an encapsulation
                  method which does not support multiple parallel
                  tunnels with the same endpoints, the management
                  station should simply use a tunnelConfigID of 1, and
                  set tunnelConfigStatus to createAndGo. For
                  encapsulation methods such as L2F which allow multiple
                  parallel tunnels, the management station may select a
                  pseudo-random number to use as the tunnelConfigID and
                  set tunnelConfigStatus to createAndGo. In the event
                  that this ID is already in use and an
                  inconsistentValue is returned in response to the set
                  operation, the management station should simply select
                  a new pseudo-random number and retry the operation.

                  Creating a row in this table will cause an interface
                  index to be assigned by the agent in an
                  implementation-dependent manner, and corresponding
                  rows will be instantiated in the ifTable and the
                  tunnelIfTable. The status of this row will become
                  active as soon as the agent assigns the interface
                  index, regardless of whether the interface is
                  operationally up.

                  Deleting a row in this table will likewise delete the
                  corresponding row in the ifTable and in the
                  tunnelIfTable.

                  Since this object does not support IPv6, it is
                  deprecated in favor of tunnelInetConfigStatus."
          ::= { tunnelConfigEntry 6 }

      tunnelInetConfigTable OBJECT-TYPE
          SYNTAX   SEQUENCE OF TunnelInetConfigEntry
          MAX-ACCESS not-accessible
          STATUS   current
          DESCRIPTION

```
            "The (conceptual) table containing information on
            configured tunnels.  This table can be used to map a
            set of tunnel endpoints to the associated ifIndex
            value.  It can also be used for row creation.  Note
            that every row in the tunnelIfTable with a fixed
            destination address should have a corresponding row in
            the tunnelInetConfigTable, regardless of whether it
            was created via SNMP."
    ::= { tunnel 3 }

tunnelInetConfigEntry OBJECT-TYPE
    SYNTAX     TunnelInetConfigEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
            "An entry (conceptual row) containing the information
            on a particular configured tunnel.  Note that there is
            a 128 subid maximum for object OIDs.  Implementers
            need to be aware that if the total number of octets in
            tunnelInetConfigLocalAddress and
            tunnelInetConfigRemoteAddress exceeds 110 then OIDs of
            column instances in this table will have more than 128
            sub-identifiers and cannot be accessed using SNMPv1,
            SNMPv2c, or SNMPv3.  In practice this is not expected
            to be a problem since IPv4 and IPv6 addresses will not
            cause the limit to be reached, but if other types are
            supported by an agent, care must be taken to ensure
            that the sum of the lengths do not cause the limit to
            be exceeded."
    INDEX      { tunnelInetConfigAddressType,
                 tunnelInetConfigLocalAddress,
                 tunnelInetConfigRemoteAddress,
                 tunnelInetConfigEncapsMethod,
                 tunnelInetConfigID }
    ::= { tunnelInetConfigTable 1 }

TunnelInetConfigEntry ::= SEQUENCE {
    tunnelInetConfigAddressType       InetAddressType,
    tunnelInetConfigLocalAddress      InetAddress,
    tunnelInetConfigRemoteAddress     InetAddress,
    tunnelInetConfigEncapsMethod      IANAtunnelType,
    tunnelInetConfigID                Integer32,
    tunnelInetConfigIfIndex           InterfaceIndexOrZero,
    tunnelInetConfigStatus            RowStatus,
    tunnelInetConfigStorageType       StorageType
}

tunnelInetConfigAddressType OBJECT-TYPE
```

```
       SYNTAX      InetAddressType
       MAX-ACCESS not-accessible
       STATUS      current
       DESCRIPTION
               "The address type over which the tunnel encapsulates
               packets."
       ::= { tunnelInetConfigEntry 1 }

   tunnelInetConfigLocalAddress OBJECT-TYPE
       SYNTAX      InetAddress
       MAX-ACCESS not-accessible
       STATUS      current
       DESCRIPTION
               "The address of the local endpoint of the tunnel, or
               0.0.0.0 (for IPv4) or :: (for IPv6) if the device is
               free to choose any of its addresses at tunnel
               establishment time."
       ::= { tunnelInetConfigEntry 2 }

   tunnelInetConfigRemoteAddress OBJECT-TYPE
       SYNTAX      InetAddress
       MAX-ACCESS not-accessible
       STATUS      current
       DESCRIPTION
               "The address of the remote endpoint of the tunnel."
       ::= { tunnelInetConfigEntry 3 }

   tunnelInetConfigEncapsMethod OBJECT-TYPE
       SYNTAX      IANAtunnelType
       MAX-ACCESS not-accessible
       STATUS      current
       DESCRIPTION
               "The encapsulation method used by the tunnel."
       ::= { tunnelInetConfigEntry 4 }

   tunnelInetConfigID OBJECT-TYPE
       SYNTAX      Integer32 (1..2147483647)
       MAX-ACCESS not-accessible
       STATUS      current
       DESCRIPTION
               "An identifier used to distinguish between multiple
               tunnels of the same encapsulation method, with the
               same endpoints.  If the encapsulation protocol only
               allows one tunnel per set of endpoint addresses (such
               as for GRE or IP-in-IP), the value of this object is
               1.  For encapsulation methods (such as L2F) which
               allow multiple parallel tunnels, the manager is
               responsible for choosing any ID which does not
```

                    conflict with an existing row, such as choosing a
                    random number."
           ::= { tunnelInetConfigEntry 5 }

       tunnelInetConfigIfIndex OBJECT-TYPE
           SYNTAX   InterfaceIndexOrZero
           MAX-ACCESS read-only
           STATUS   current
           DESCRIPTION
                    "If the value of tunnelInetConfigStatus for this row
                    is active, then this object contains the value of
                    ifIndex corresponding to the tunnel interface. A
                    value of 0 is not legal in the active state, and means
                    that the interface index has not yet been assigned."
           ::= { tunnelInetConfigEntry 6 }

       tunnelInetConfigStatus OBJECT-TYPE
           SYNTAX   RowStatus
           MAX-ACCESS read-create
           STATUS   current
           DESCRIPTION
                    "The status of this row, by which new entries may be
                    created, or old entries deleted from this table. The
                    agent need not support setting this object to
                    createAndWait or notInService since there are no other
                    writable objects in this table, and writable objects
                    in rows of corresponding tables such as the
                    tunnelIfTable may be modified while this row is
                    active.

                    To create a row in this table for an encapsulation
                    method which does not support multiple parallel
                    tunnels with the same endpoints, the management
                    station should simply use a tunnelInetConfigID of 1,
                    and set tunnelInetConfigStatus to createAndGo. For
                    encapsulation methods such as L2F which allow multiple
                    parallel tunnels, the management station may select a
                    pseudo-random number to use as the tunnelInetConfigID
                    and set tunnelInetConfigStatus to createAndGo. In the
                    event that this ID is already in use and an
                    inconsistentValue is returned in response to the set
                    operation, the management station should simply select
                    a new pseudo-random number and retry the operation.

                    Creating a row in this table will cause an interface
                    index to be assigned by the agent in an
                    implementation-dependent manner, and corresponding
                    rows will be instantiated in the ifTable and the

```
            tunnelIfTable.  The status of this row will become
            active as soon as the agent assigns the interface
            index, regardless of whether the interface is
            operationally up.

            Deleting a row in this table will likewise delete the
            corresponding row in the ifTable and in the
            tunnelIfTable."
    ::= { tunnelInetConfigEntry 7 }

tunnelInetConfigStorageType OBJECT-TYPE
    SYNTAX      StorageType
    MAX-ACCESS read-create
    STATUS      current
    DESCRIPTION
            "The storage type of this row.  If the row is
            permanent(4), no objects in the row need be writable."
    ::= { tunnelInetConfigEntry 8 }

-- conformance information

tunnelMIBConformance
                OBJECT IDENTIFIER ::= { tunnelMIB 2 }
tunnelMIBCompliances
                OBJECT IDENTIFIER ::= { tunnelMIBConformance 1 }
tunnelMIBGroups  OBJECT IDENTIFIER ::= { tunnelMIBConformance 2 }

-- compliance statements

tunnelMIBCompliance MODULE-COMPLIANCE
    STATUS  deprecated
    DESCRIPTION
            "The (deprecated) IPv4-only compliance statement for
            the IP Tunnel MIB.

            This is deprecated in favor of
            tunnelMIBInetFullCompliance and
            tunnelMIBInetReadOnlyCompliance."
    MODULE  -- this module
    MANDATORY-GROUPS { tunnelMIBBasicGroup }

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

        OBJECT       tunnelIfTOS
        MIN-ACCESS  read-only
```

```
        DESCRIPTION
            "Write access is not required."

        OBJECT       tunnelConfigStatus
        MIN-ACCESS   read-only
        DESCRIPTION
            "Write access is not required."
   ::= { tunnelMIBCompliances 1 }

tunnelMIBInetFullCompliance MODULE-COMPLIANCE
    STATUS   current
    DESCRIPTION
            "The full compliance statement for the IP Tunnel MIB."
    MODULE   -- this module
    MANDATORY-GROUPS { tunnelMIBInetGroup }

        OBJECT       tunnelIfAddressType
        SYNTAX       InetAddressType { ipv4(1), ipv6(2),
                                       ipv4z(3), ipv6z(4) }
        DESCRIPTION
            "An implementation is only required to support IPv4
            and/or IPv6 addresses.  An implementation only needs to
            support the addresses it actually supports on the
            device."
   ::= { tunnelMIBCompliances 2 }

tunnelMIBInetReadOnlyCompliance MODULE-COMPLIANCE
    STATUS   current
    DESCRIPTION
            "The read-only compliance statement for the IP Tunnel
            MIB."
    MODULE   -- this module
    MANDATORY-GROUPS { tunnelMIBInetGroup }

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

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

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

```
        OBJECT       tunnelIfAddressType
        SYNTAX       InetAddressType { ipv4(1), ipv6(2),
                                       ipv4z(3), ipv6z(4) }
        MIN-ACCESS   read-only
        DESCRIPTION
            "Write access is not required.

            An implementation is only required to support IPv4
            and/or IPv6 addresses.  An implementation only needs to
            support the addresses it actually supports on the
            device."

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

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

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

        OBJECT       tunnelInetConfigStatus
        MIN-ACCESS   read-only
        DESCRIPTION
            "Write access is not required, and active is the only
            status that needs to be supported."

        OBJECT       tunnelInetConfigStorageType
        MIN-ACCESS   read-only
        DESCRIPTION
            "Write access is not required."
    ::= { tunnelMIBCompliances 3 }

-- units of conformance

tunnelMIBBasicGroup OBJECT-GROUP
    OBJECTS { tunnelIfLocalAddress, tunnelIfRemoteAddress,
       tunnelIfEncapsMethod, tunnelIfHopLimit, tunnelIfTOS,
       tunnelIfSecurity, tunnelConfigIfIndex, tunnelConfigStatus }
    STATUS  deprecated
    DESCRIPTION
            "A collection of objects to support basic management
```

```
                    of IPv4 Tunnels.  Since this group cannot support
                    IPv6, it is deprecated in favor of
                    tunnelMIBInetGroup."
            ::= { tunnelMIBGroups 1 }

    tunnelMIBInetGroup OBJECT-GROUP
        OBJECTS { tunnelIfAddressType, tunnelIfLocalInetAddress,
           tunnelIfRemoteInetAddress, tunnelIfEncapsMethod,
           tunnelIfEncapsLimit,
           tunnelIfHopLimit, tunnelIfTOS, tunnelIfFlowLabel,
           tunnelIfSecurity, tunnelInetConfigIfIndex,
           tunnelInetConfigStatus, tunnelInetConfigStorageType }
        STATUS   current
        DESCRIPTION
                "A collection of objects to support basic management
                of IPv4 and IPv6 Tunnels."
        ::= { tunnelMIBGroups 2 }

    END
```

5. IANA Considerations

   This document introduces a new IANA-maintained textual convention
   (TC) which has been added to the IANAifType-MIB [IFTYPE]. The
   initial version of this IANAtunnelType TC can be found in Appendix A.
   The current version of the textual convention can be accessed at
   http://www.iana.org/assignments/ianaiftype-mib

   The assignment policy for IANAtunnelType values should always be
   identical to the policy for assigning IANAifType values.

   New types of tunnels over IPv4 or IPv6 should not be assigned
   IANAifType values. Instead, they should be assigned IANAtunnelType
   values and hence reuse the interface type tunnel(131). (Note this
   restriction does not apply to "tunnels" which are not over IPv4 or
   IPv6.)

   Previously, tunnel types that were not point-to-point tunnels were
   problematic in that they could not be properly expressed in the
   tunnel MIB, and hence were assigned IANAifType values. This document
   now corrects this problem, and as a result, IANA has deprecated the
   sixToFour(215) IANAifType value in favor of the sixToFour(11)
   IANAtunnelType value.

6. Security Considerations

   There are a number of management objects defined in this MIB module
   with 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.

   Unauthorized write access to any of the writable objects could cause
   unauthorized creation and/or manipulation of tunnels, resulting in a
   denial of service, or redirection of packets to an arbitrary
   destination.

   Some of the readable objects in this MIB module (i.e., objects with a
   MAX-ACCESS other than not-accessible) may be considered sensitive or
   vulnerable in some network environments. It is thus important to
   control even GET and/or NOTIFY access to these objects and possibly
   to even encrypt the values of these objects when sending them over
   the network via SNMP.

   Unauthorized read access to tunnelIfLocalInetAddress,
   tunnelIfRemoteInetAddress, tunnelIfLocalAddress,
   tunnelIfRemoteAddress, or any object in the tunnelConfigTable or
   tunnelInetConfigTable would reveal information about the tunnel
   topology.

   SNMP versions prior to SNMPv3 did not include adequate security.
   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 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.

7. Changes Since RFC 2667

   IPv4-specific objects were deprecated, including
   tunnelIfLocalAddress, tunnelIfRemoteAddress, the tunnelConfigTable,
   and the tunnelMIBBasicGroup.

   Added IP version-agnostic objects that should be used instead,
   including tunnelIfAddressType, tunnelIfLocalInetAddress,
   tunnelIfRemoteInetAddress, the tunnelInetConfigTable, and the
   tunnelIMIBInetGroup.

   The new tunnelIfLocalInetAddress and tunnelIfRemoteInetAddress
   objects are read-write, rather than read-only.

   Updated DESCRIPTION clauses of existing version-agnostic objects
   (e.g., tunnelIfTOS) that contained IPv4-specific text to cover IPv6
   as well.

   Added tunnelIfFlowLabel for tunnels over IPv6.

   The encapsulation method was previously an INTEGER type, and is now
   an IANA-maintained textual convention.

8. Acknowledgements

   This MIB module was updated based on feedback from the IETF's
   Interfaces MIB (IF-MIB), Point-to-Point Protocol Extensions (PPPEXT),
   and IPv6 Working Groups. Mike Heard and Ville Nuorvala also provided
   valuable MIB guidance on this version.

Appendix A: IANA Tunnel Type TC

   This appendix defines the initial content of the IANAtunnelType
   textual convention. The most up-to-date and current version is
   maintained in the IANAifType-MIB.

   IANAtunnelType ::= TEXTUAL-CONVENTION
       STATUS   current
       DESCRIPTION
               "The encapsulation method used by a tunnel. The value
               direct indicates that a packet is encapsulated
               directly within a normal IP header, with no
               intermediate header, and unicast to the remote tunnel
               endpoint (e.g., an RFC 2003 IP-in-IP tunnel, or an RFC
               1933 IPv6-in-IPv4 tunnel). The value minimal indicates
               that a Minimal Forwarding Header (RFC 2004) is
               inserted between the outer header and the payload
               packet. The value UDP indicates that the payload
               packet is encapsulated within a normal UDP packet
               (e.g., RFC 1234).

               The values sixToFour, sixOverFour, and isatap
               indicates that an IPv6 packet is encapsulated directly
               within an IPv4 header, with no intermediate header,
               and unicast to the destination determined by the 6to4,
               6over4, or ISATAP protocol.

               The remaining protocol-specific values indicate that a
               header of the protocol of that name is inserted
               between the outer header and the payload header.

               The assignment policy for IANAtunnelType values is
               identical to the policy for assigning IANAifType
               values."
       SYNTAX   INTEGER {
                       other(1),   -- none of the following
                       direct(2),   -- no intermediate header
                       gre(3),   -- GRE encapsulation
                       minimal(4),   -- Minimal encapsulation
                       l2tp(5),   -- L2TP encapsulation
                       pptp(6),   -- PPTP encapsulation
                       l2f(7),   -- L2F encapsulation
                       udp(8),   -- UDP encapsulation
                       atmp(9),   -- ATMP encapsulation
                       msdp(10),   -- MSDP encapsulation
                       sixToFour(11),   -- 6to4 encapsulation
                       sixOverFour(12), -- 6over4 encapsulation
                       isatap(13),   -- ISATAP encapsulation

```
                         teredo(14)       -- Teredo encapsulation
                 }
```

Normative References

    [IFTYPE]   Internet Assigned Numbers Authority, "IANAifType-MIB",
                 http://www.iana.org/assignments/ianaiftype-mib.

    [RFC2473]   Conta, A. and S. Deering, "Generic Packet Tunneling in
                 IPv6 Specification", RFC 2473, December 1998.

    [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.

    [RFC2863]   McCloghrie, K. and F. Kastenholz. "The Interfaces Group
                 MIB", RFC 2863, June 2000.

    [RFC3595]   Wijnen, B., "Textual Conventions for IPv6 Flow Label",
                 RFC 3595, September 2003.

    [RFC4001]   Daniele, M., Haberman, B., Routhier, S., and J.
                 Schoenwaelder, "Textual Conventions for Internet Network
                 Addresses", RFC 4001, February 2005.

Informative References

    [RFC1234]   Provan, D., "Tunneling IPX Traffic through IP Networks",
                 RFC 1234, June 1991.

    [RFC1241]   Woodburn, R. and D. Mills, "A Scheme for an Internet
                 Encapsulation Protocol: Version 1", RFC 1241, July 1991.

    [RFC1701]   Hanks, S., Li, T., Farinacci, D., and P. Traina, "Generic
                 Routing Encapsulation (GRE)", RFC 1701, October 1994.

    [RFC1702]   Hanks, S., Li, T., Farinacci, D., and P. Traina, "Generic
                 Routing Encapsulation over IPv4 networks", RFC 1702,
                 October 1994.

   [RFC2003]   Perkins, C., "IP Encapsulation within IP", RFC 2003,
                October 1996.

   [RFC2004]   Perkins, C., "Minimal Encapsulation within IP", RFC 2004,
                October 1996.

   [RFC2107]   Hamzeh, K., "Ascend Tunnel Management Protocol - ATMP",
                RFC 2107, February 1997.

   [RFC2341]   Valencia, A., Littlewood, M., and T. Kolar. "Cisco Layer
                Two Forwarding (Protocol) "L2F"", RFC 2341, May 1998.

   [RFC2401]   Kent, S. and R. Atkinson, "Security Architecture for the
                Internet Protocol", RFC 2401, November 1998.

   [RFC2474]   Nichols, K., Blake, S., Baker, F., and D. Black.
                "Definition of the Differentiated Services Field (DS
                Field) in the IPv4 and IPv6 Headers", RFC 2474, December
                1998.

   [RFC2637]   Hamzeh, K., Pall, G., Verthein, W. Taarud, J., Little,
                W., and G. Zorn, "Point-to-Point Tunneling Protocol",
                RFC 2637, July 1999.

   [RFC2661]   Townsley, W., Valencia, A., Rubens, A., Pall, G., Zorn,
                G., and B. Palter, "Layer Two Tunneling Protocol "L2TP"",
                RFC 2661, August 1999.

   [RFC2893]   Gilligan, R. and E. Nordmark. "Transition Mechanisms for
                IPv6 Hosts and Routers", RFC 2893, August 2000.

   [RFC3410]   Case, J., Mundy, R., Partain, D., and B. Stewart,
                "Introduction and Applicability Statements for Internet-
                Standard Management Framework", RFC 3410, December 2002.

Author's Address

   Dave Thaler
   Microsoft Corporation
   One Microsoft Way
   Redmond, WA 98052-6399

   Phone: +1 425 703 8835
   EMail: dthaler@microsoft.com

Full Copyright Statement

Intellectual Property

Acknowledgement