```
Network Working Group                                        D. Thaler
Request for Comments: 2667                                   Microsoft
Category: Standards Track                                  August 1999
```

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

1. Abstract

   This memo defines a Management Information Base (MIB) 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 networks. Extension MIBs may be designed for
   managing protocol-specific objects. Likewise, extension MIBs may be
   designed for managing security-specific objects. This MIB does not
   support tunnels over non-IPv4 networks (including IPv6 networks).
   Management of such tunnels may be supported by other MIBs.

Table of Contents

2. Introduction

   Over the past several years, there have been a number of "tunneling"
   protocols specified by the IETF (see [28] for an early discussion of
   the model and examples). This document describes a Management
   Information Base (MIB) used for managing tunnels of any type over
   IPv4 networks, including GRE [16,17], IP-in-IP [18], Minimal
   Encapsulation [19], L2TP [20], PPTP [21], L2F [25], UDP (e.g., [26]),
   ATMP [22], and IPv6-in-IPv4 [27] tunnels.

   Extension MIBs may be designed for managing protocol-specific
   objects. Likewise, extension MIBs may be designed for managing
   security-specific objects (e.g., IPSEC [24]), and traffic conditioner
   [29] objects. Finally, this MIB does not support tunnels over non-
   IPv4 networks (including IPv6 networks). Management of such tunnels
   may be supported by other MIBs.

3. The SNMP Network Management Framework

   The SNMP Management Framework presently consists of five major
   components:

   o   An overall architecture, described in RFC 2571 [1].

   o   Mechanisms for describing and naming objects and events for the
          purpose of management. The first version of this Structure of
          Management Information (SMI) is called SMIv1 and described in
          STD 16, RFC 1155 [2], STD 16, RFC 1212 [3] and RFC 1215 [4]. The
          second version, called SMIv2, is described in STD 58, RFC 2578
          [5], STD 58, RFC 2579 [6] and STD 58, RFC 2580 [7].

   o   Message protocols for transferring management information. The
          first version of the SNMP message protocol is called SNMPv1 and
          described in STD 15, RFC 1157 [8]. A second version of the SNMP
          message protocol, which is not an Internet standards track
          protocol, is called SNMPv2c and described in RFC 1901 [9] and
          RFC 1906 [10]. The third version of the message protocol is
          called SNMPv3 and described in RFC 1906 [10], RFC 2572 [11] and
          RFC 2574 [12].

   o   Protocol operations for accessing management information. The
          first set of protocol operations and associated PDU formats is
          described in STD 15, RFC 1157 [8]. A second set of protocol
          operations and associated PDU formats is described in RFC 1905
          [13].

   o   A set of fundamental applications described in RFC 2573 [14] and
        the view-based access control mechanism described in RFC 2575
        [15].

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

   This memo specifies a MIB module that is compliant to the SMIv2. A
   MIB conforming to the SMIv1 can be produced through the appropriate
   translations. The resulting translated MIB must be semantically
   equivalent, except where objects or events are omitted because no
   translation is possible (use of Counter64). Some machine readable
   information in SMIv2 will be converted into textual descriptions in
   SMIv1 during the translation process. However, this loss of machine
   readable information is not considered to change the semantics of the
   MIB.

4. Overview
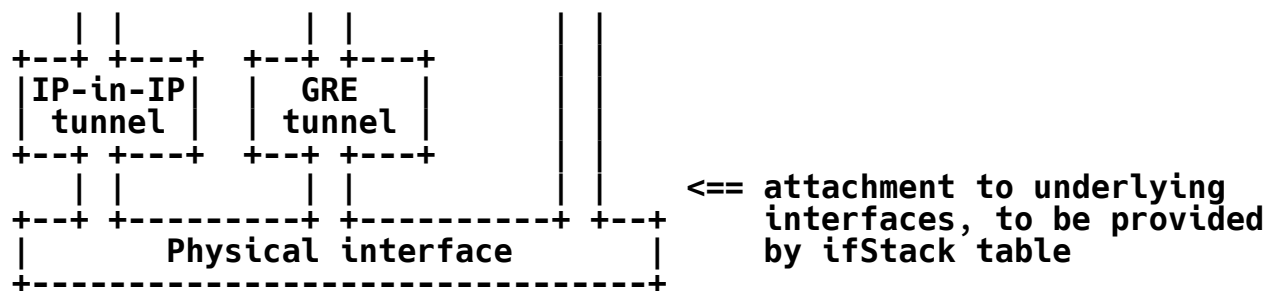
   This MIB module contains two tables:

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

   o   the Tunnel 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.

4.1. Relationship to the Interfaces MIB

   This section clarifies the relationship of this MIB to the Interfaces
   MIB [23]. 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.

4.1.1. Layering Model

   Each logical interface (physical or virtual) has an ifEntry in the
   Interfaces MIB [23]. 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 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
     +---------------------------------+
```

4.1.2. ifRcvAddressTable

   The ifRcvAddressTable usage is defined in the MIBs defining the
   encapsulation below the network layer. For example, if IP-in-IP
   encapsulation is being used, the ifRcvAddressTable is defined by IP-
   in-IP.

4.1.3. ifEntry

   IfEntries are defined in the MIBs 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 will exist for every ifEntry with this ifType. An
   implementation of the IP Tunnel MIB 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 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 which 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.

5. Definitions

TUNNEL-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY, OBJECT-TYPE, transmission,
    Integer32, IpAddress   FROM SNMPv2-SMI
    RowStatus   FROM SNMPv2-TC
    MODULE-COMPLIANCE, OBJECT-GROUP FROM SNMPv2-CONF
    ifIndex, InterfaceIndexOrZero   FROM IF-MIB;

tunnelMIB MODULE-IDENTITY
    LAST-UPDATED "9908241200Z" -- August 24, 1999
    ORGANIZATION "IETF Interfaces MIB Working Group"
    CONTACT-INFO
            " Dave Thaler
              Microsoft Corporation
              One Microsoft Way
              Redmond, WA   98052-6399
              EMail: dthaler@dthaler.microsoft.com"
    DESCRIPTION
            "The MIB module for management of IP Tunnels, independent of
            the specific encapsulation scheme in use."
    REVISION   "9908241200Z" -- 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,
    tunnelIfRemoteAddress           IpAddress,
    tunnelIfEncapsMethod            INTEGER,
    tunnelIfHopLimit                Integer32,
    tunnelIfSecurity                INTEGER,
    tunnelIfTOS                     Integer32
}

tunnelIfLocalAddress OBJECT-TYPE
    SYNTAX      IpAddress
    MAX-ACCESS read-only
    STATUS      current
    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."
    ::= { tunnelIfEntry 1 }

tunnelIfRemoteAddress OBJECT-TYPE
    SYNTAX      IpAddress
    MAX-ACCESS read-only
    STATUS      current
    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."
    ::= { tunnelIfEntry 2 }

tunnelIfEncapsMethod OBJECT-TYPE
    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
```

```
                }
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "The encapsulation method used by the tunnel. The value
                direct indicates that the packet is encapsulated directly
                within a normal IPv4 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 remaining protocol-specific values indicate that
                a header of the protocol of that name is inserted between
                the outer header and the payload header."
        ::= { tunnelIfEntry 3 }

tunnelIfHopLimit OBJECT-TYPE
        SYNTAX      Integer32 (0..255)
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
                "The TTL 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 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 of the TOS 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.  A value
                between 0 and 63 inclusive indicates that the bit field is
                set to the indicated value."
        ::= { tunnelIfEntry 6 }

tunnelConfigTable OBJECT-TYPE
    SYNTAX       SEQUENCE OF TunnelConfigEntry
    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 tunnelConfigTable, regardless of
                whether it was created via SNMP."
        ::= { tunnel 2 }

tunnelConfigEntry OBJECT-TYPE
    SYNTAX       TunnelConfigEntry
    MAX-ACCESS not-accessible
    STATUS       current
    DESCRIPTION
                "An entry (conceptual row) containing the information on a
                particular configured tunnel."
    INDEX      { tunnelConfigLocalAddress,
                  tunnelConfigRemoteAddress,
                  tunnelConfigEncapsMethod,
                  tunnelConfigID }
        ::= { tunnelConfigTable 1 }

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

tunnelConfigLocalAddress OBJECT-TYPE
```

```
    SYNTAX      IpAddress
    MAX-ACCESS  not-accessible
    STATUS      current
    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."
    ::= { tunnelConfigEntry 1 }

tunnelConfigRemoteAddress OBJECT-TYPE
    SYNTAX      IpAddress
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The address of the remote endpoint of the tunnel."
    ::= { tunnelConfigEntry 2 }

tunnelConfigEncapsMethod OBJECT-TYPE
    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)
                }
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The encapsulation method used by the tunnel."
    ::= { tunnelConfigEntry 3 }

tunnelConfigID 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."
```

    ::= { tunnelConfigEntry 4 }

tunnelConfigIfIndex OBJECT-TYPE
    SYNTAX   InterfaceIndexOrZero
    MAX-ACCESS read-only
    STATUS   current
    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."
    ::= { tunnelConfigEntry 5 }

tunnelConfigStatus 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
            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."
        ::= { tunnelConfigEntry 6 }

-- conformance information

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

-- compliance statements

tunnelMIBCompliance MODULE-COMPLIANCE
     STATUS   current
     DESCRIPTION
             "The compliance statement for the IP Tunnel MIB."
     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 }

-- units of conformance

tunnelMIBBasicGroup OBJECT-GROUP
     OBJECTS { tunnelIfLocalAddress, tunnelIfRemoteAddress,
       tunnelIfEncapsMethod, tunnelIfHopLimit, tunnelIfTOS,
       tunnelIfSecurity, tunnelConfigIfIndex, tunnelConfigStatus }
     STATUS   current
     DESCRIPTION
             "A collection of objects to support basic management of IP
             Tunnels."
     ::= { tunnelMIBGroups 1 }

END

6. Security Considerations

   This MIB contains readable objects whose values provide information
   related to IP tunnel interfaces. There are also a number of objects
   that have a MAX-ACCESS clause of read-write and/or read-create, such
   as those which allow an administrator to dynamically configure
   tunnels.

   While unauthorized access to the readable objects is relatively
   innocuous, unauthorized access to the write-able objects could cause
   a denial of service, or could cause unauthorized creation and/or
   manipulation of tunnels. Hence, the support for SET operations in a
   non-secure environment without proper protection can have a negative
   effect on network operations.

   SNMPv1 by itself is such an insecure environment. Even if the
   network itself is secure (for example by using IPSec [24]), even
   then, there is no control as to who on the secure network is allowed
   to access and SET (change/create/delete) the objects in this MIB.

   It is recommended that the implementers consider the security
   features as provided by the SNMPv3 framework. Specifically, the use
   of the User-based Security Model RFC 2574 [12] and the View-based
   Access Control Model RFC 2575 [15] is recommended.

   It is then a customer/user responsibility to ensure that the SNMP
   entity giving access to this MIB, is properly configured to give
   access to those objects only to those principals (users) that have
   legitimate rights to access them.

7. Acknowledgements

   This MIB module was updated based on feedback from the IETF's
   Interfaces MIB (IF-MIB) and Point-to-Point Protocol Extensions
   (PPPEXT) Working Groups.

8. Author's Address

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

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

9. References

[1] Wijnen, B., Harrington, D. and R. Presuhn, "An Architecture for
     Describing SNMP Management Frameworks", RFC 2571, April 1999.

[2] Rose, M. and K. McCloghrie, "Structure and Identification of
     Management Information for TCP/IP-based Internets", STD 16, RFC
     1155, May 1990.

[3] Rose, M. and K. McCloghrie, "Concise MIB Definitions", STD 16,
     RFC 1212, March 1991.

[4] Rose, M., "A Convention for Defining Traps for use with the
     SNMP", RFC 1215, March 1991.

[5] McCloghrie, K., Perkins, D. and J. Schoenwaelder, "Structure of
     Management Information Version 2 (SMIv2)", STD 58, RFC 2578,
     April 1999.

[6] McCloghrie, K., Perkins, D. and J. Schoenwaelder, "Textual
     Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[7] McCloghrie, K., Perkins, D. and J. Schoenwaelder, "Conformance
     Statements for SMIv2", STD 58, RFC 2580, April 1999.

[8] Case, J., Fedor, M., Schoffstall, M. and J. Davin, "Simple
     Network Management Protocol", STD 15, RFC 1157, May 1990.

[9] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser,
     "Introduction to Community-based SNMPv2", RFC 1901, January
     1996.

[10] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Transport
     Mappings for Version 2 of the Simple Network Management Protocol
     (SNMPv2)", RFC 1906, January 1996.

[11] Case, J., Harrington D., Presuhn R. and B. Wijnen, "Message
     Processing and Dispatching for the Simple Network Management
     Protocol (SNMP)", RFC 2572, April 1999.

[12] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM)
     for version 3 of the Simple Network Management Protocol
     (SNMPv3)", RFC 2574, April 1999.

[13] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Protocol
     Operations for Version 2 of the Simple Network Management
     Protocol (SNMPv2)", RFC 1905, January 1996.

[14] Levi, D., Meyer, P. and B. Stewart, "SNMPv3 Applications", RFC
     2573, April 1999.

[15] Wijnen, B., Presuhn, R. and K. McCloghrie, "View-based Access
     Control Model (VACM) for the Simple Network Management Protocol
     (SNMP)", RFC 2575, April 1999.

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

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

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

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

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

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

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

[23] McCloghrie, K. and F. Kastenholz. "The Interfaces Group MIB
     using SMIv2", RFC 2233, November 1997.

[24] R. Atkinson, "Security architecture for the internet protocol",
     RFC 2401, November 1998.

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

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

[27] Gilligan, R. and E. Nordmark. "Transition Mechanisms for IPv6
     Hosts and Routers", RFC 1933, April 1996.

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

[29] 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.

10. Intellectual Property Notice

The IETF takes no position regarding the validity or scope of any
intellectual property or other rights that might be claimed to
pertain to the implementation or use of the technology described in
this document or the extent to which any license under such rights
might or might not be available; neither does it represent that it
has made any effort to identify any such rights. Information on the
IETF's procedures with respect to rights in standards-track and
standards-related documentation can be found in BCP-11. Copies of
claims of rights made available for publication and any assurances of
licenses to be made available, or the result of an attempt made to
obtain a general license or permission for the use of such
proprietary rights by implementers or users of this specification can
be obtained from the IETF Secretariat."

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

11. Full Copyright Statement

Acknowledgement