Dan Romascanu
Avaya Inc.
22 February 2001

Power Ethernet (DTE Power via MDI) MIB

<draft-romascanu-hubmib-power-ethernet-mib-00.txt>

Status of this Memo

Copyright Notice

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. The document proposes an extension to the Ethernet-like Interfaces MIB [RFC2665] with a set of objects for managing a power Ethernet Powered Device (PD) and/or Power Source Equipment (PSE).

Distribution of this memo is unlimited.

Table of Contents

1. 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 defines a set of MIB objects to manage a Power Ethernet (DTE Power via MDI)Powered Device (PD) and/or power Source Equipment (PSE).

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 [RFC2863].

2. The SNMP Management Framework

The SNMP Management Framework presently consists of five major components:

- An overall architecture, described in RFC 2571 [RFC2571].
- Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIv1 and described in STD 16, RFC 1155 [RFC1155], STD 16, RFC 1212 [RFC1212] and RFC 1215 [RFC1215]. The second version, called SMIv2, is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].
- Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in STD 15, RFC 1157 [RFC1157]. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in RFC 1901 [RFC1901] and RFC 1906 [RFC1906]. The third version of the message protocol is called SNMPv3 and described in RFC 1906 [RFC1906], RFC 2572 [RFC2572] and RFC 2574 [RFC2574].
- Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in STD 15, RFC 1157 [RFC1157]. A second set of protocol operations and associated PDU formats is described in

RFC 1905 [RFC1905].

- o A set of fundamental applications described in RFC 2573 [RFC2573] and the view-based access control mechanism described in RFC 2575 [RFC2575].

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

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

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

3. Overview

The emergence of IP telephony as an application that allows for voice applications to be run over the same infrastructure as data applications led to the emergence of Ethernet IP phones, with similar functions and characteristics as the traditional phones. Powering a phone is one of these functions that are being taken as granted. The IEEE 802.3 Working Group initiated a standard work on this subject, currently known as the IEEE 802.3af work [IEEE-802.3af].

The IEEE 802.3af WG will not define a full management interface, but only the hardware registers that will allow for a management interfaces to be built for a powered Ethernet device. The MIB module defined in this document extends the Ethernet-like Interfaces MIB [RFC2665] with the management objects required for the management of the powered Ethernet devices and ports.

The following abrviations are defined in [IEEE-802.3af] and will be used with the same significance in this document: PSE - Power Sourcing Equipment; PD - Powered Device

4. MIB Structure

This MIB module is composed of two tables and one MIB group.

The pethPsePortTable deines the objects used for the configuration and describing the status of ports on a PSE device. Examples of PSE devices are Ethernet switches that support power Ethernet and mid-span boxes.

The pethPdPortTable defines the objects used for the configuration and describing the status of ports on a PD device. Examples of PD devices are Ethernet phones.

The pethMainPseObjects MIB group defines the management objects for a managed main power source in a PSE device. Ethernet switches are one example of boxes that would support these objects.

5. Evolution of the Document, Limitations and Future Work

The IEEE 802.3af is at this stage work in progress. The scope of this document is to initiate standards work in the IETF in order to allow for the publication of a standard track document conmtaining an SNMP MIB simultaneously or close to the date of the publication of the IEEE revised standard. It is expected that changes may be brought to the IEEE proposal. There are aspects that were not yet included in the first version of the MIB like use of notifications.

6. Definitions

```
PETH-MIB DEFINITIONS ::= BEGIN

   IMPORTS
        MODULE-IDENTITY, OBJECT-TYPE, Integer32
                FROM SNMPv2-SMI
        dot3
                FROM EtherLike-MIB
        TruthValue
                FROM SNMPv2-TC
        InterfaceIndex
               FROM IF-MIB
        MODULE-COMPLIANCE, OBJECT-GROUP
                FROM SNMPv2-CONF;

   powerEthernetMIB MODULE-IDENTITY
        LAST-UPDATED "200102220000Z"
        ORGANIZATION "Avaya Inc."
        CONTACT-INFO
        "
```

Dan Romascanu
Avaya Inc.
Tel: +972-3-645-8414
Email: dromasca@avaya.com"

```
        DESCRIPTION
                "The MIB module for for managing Powered Devices (PD) or
                Power Source Equipment (PSE) working according to the IEEE
                802.af Powere Ethernet (DTE Power via MDI) standard."
         ::= { dot3 20 }

  pethObjects       OBJECT IDENTIFIER ::= { powerEthernetMIB 1 }
  pethNotifications OBJECT IDENTIFIER ::= { powerEthernetMIB 2 }
  pethConformance   OBJECT IDENTIFIER ::= { powerEthernetMIB 3 }

  -- pethAgentControl MIB group defines the control objects for the power
  -- Ethernet Agent

    pethPsePortTable OBJECT-TYPE
        SYNTAX      SEQUENCE OF PethPsePortEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "A table of objects that display and control the power
             characteristics power Ethernet ports on a Power Source
             Entity (PSE) device. This group will be implemented in
             managed power Ethernet switches and mid-span devices."
        ::= { pethObjects 1 }

     pethPsePortEntry OBJECT-TYPE
        SYNTAX      PethPsePortEntry
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
                "A set of objects that display and control the power
             characteristics of a power Ethernet PSE port."
        INDEX    { pethPsePortIndex }
        ::= { pethPsePortTable 1 }

     PethPsePortEntry ::= SEQUENCE {
        pethPsePortIndex
            InterfaceIndex,
        pethPsePortPowerEnable
            INTEGER,
        pethPsePortPowerIdPairsControl
            TruthValue,
        pethPsePortPowerIdPairs
            INTEGER,
```

```
        pethPsePortPowerDetectionStatus
            INTEGER,
        pethPsePortDetectionOperStatus
            INTEGER,
        pethPsePortPowerPriority
            INTEGER,
        pethPsePortDenyError
            INTEGER,
        pethPsePortFaultError
            INTEGER,
        pethPsePortFaultErrorClear
            INTEGER,
        pethPsePortType
            INTEGER
    }

  pethPsePortIndex OBJECT-TYPE
        SYNTAX      InterfaceIndex
        MAX-ACCESS  not-accessible
        STATUS      current
        DESCRIPTION
            "An index value that uniquely identifies an
             interface to a PSE device.  The
             interface identified by a particular value of
             this index is the same interface as identified
             by the same value of ifIndex. The mapping
             between the ifIndex values and the numbering of
             the port on the device is an implementation
             issue."
        ::= { pethPsePortEntry 1 }

  pethPsePortPowerEnable OBJECT-TYPE
    SYNTAX INTEGER   {
               auto(1),
               off(2),
               test(3)
  }
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "Enables power supply on this port.
         Setting this object at a value auto(1) enables power
         and detection mechanism for this port.
         Setting this object at a value off(2) disables power
         and detection mechanism for this port.
         Setting this object at a value test(3) sets the port
         in a testing mode - deection pulses are permanently
         sent, power is turned off."
```

```
        ::= { pethPsePortEntry 2 }

    pethPsePortPowerIdPairsControl OBJECT-TYPE
        SYNTAX TruthValue
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "Describes the capability of controlling the power
             pairs functionality to switch pins for sourcing power."
        ::= { pethPsePortEntry 3 }

    pethPsePortPowerIdPairs OBJECT-TYPE
        SYNTAX INTEGER   {
                   signal(1),
                   spare(2),
                   both(3)
    }
        MAX-ACCESS read-write
        STATUS current
        DESCRIPTION
            "Describes or controls the pairs in use. If the value of
             pethPsePortPowerIdpairsControl is true, thisobject is
             writable.
             A value of signal(1) menas that the signal pairs
             only are in use.
             A value of spare(2) means that the spare pairs
             only are in use.
             A value of both(3) means that both the signal
             and the spare pairs are inuse."
        ::= { pethPsePortEntry 4 }

    pethPsePortPowerDetectionStatus OBJECT-TYPE
        SYNTAX INTEGER   {
                   auto(1),
                   off(2),
                   test(3)
    }
        MAX-ACCESS read-write
        STATUS current
        DESCRIPTION
            "Controls the power detection mechanism of the port.
             Setting the value auto(1) enables the power detection
             mechanism of the port.
             Setting the value off(2) disables the power detection
             mechanism of the port.
             Setting the value test(3) "
        ::= { pethPsePortEntry 5 }
```

```
   pethPsePortDetectionOperStatus OBJECT-TYPE
       SYNTAX INTEGER   {
                  deliveringPower(1),
                  off(2),
                  searching(3),
                  fault(4)
   }
       MAX-ACCESS read-only
       STATUS current
       DESCRIPTION
           "Describes the operational status of the port detection.
            A value of deliveringPower(1) indicates that the port
            executed the detection algorithm, found a PD connection
            and is currently delivering power.
            A value of off(2) indicates that the port did not find
            a PD connection and is not delivering power.
            A value of searching(3) indicates that the detection
            algorithm is in work, and did not completwe its action. No
            power is currently provided.
            A value of fault(4) indicates that a fault was detected
            on the port. "
       ::= { pethPsePortEntry 6 }

   pethPsePortPowerPriority OBJECT-TYPE
       SYNTAX INTEGER   {
                  critical(1),
                  high(2),
                  low(3)
   }
       MAX-ACCESS read-write
       STATUS current
       DESCRIPTION
           "This object controls the priority of the port from the point
            of view of a power management algorithm. The priority that
            is set by this variable could be used by a control mechanism
            that prevents over current situations by disconnecting first
            ports with lower power priority. Ports that connect devices
            critical to the operation of the network - like the E911
            telephones ports - should be set to higher priority."
       ::= { pethPsePortEntry 7 }

   pethPsePortDenyError OBJECT-TYPE
       SYNTAX INTEGER   {
                  other(1),
                  lowPriority(2)
   }
       MAX-ACCESS read-only
       STATUS current
```

```
        DESCRIPTION
            "This object describes an error resulted from an action of the
              power management mechanism. The value lowPriority(2) indicates
              that the port was disabled by the power management system, in
              order to keep active higher priority ports."
        ::= { pethPsePortEntry 8 }

    pethPsePortFaultError OBJECT-TYPE
        SYNTAX INTEGER   {
                    none(1),
                    underCurrent(2),
                    overCurrent(3)
    }
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "Describes a current port error related to the power generation
             The value underCurrent(2) indicates that the port current
             is below the minimal value.
             The value overCurrent(3) indicates that the port current
             exceeds the maximal value."
        ::= { pethPsePortEntry 9 }

    pethPsePortFaultErrorClear OBJECT-TYPE
        SYNTAX INTEGER   {
                    clear(1),
                    off(2)
    }
        MAX-ACCESS read-write
        STATUS current
        DESCRIPTION
            "Setting the value of this object to clear(1) clears the value
             of the pethPsePortFaulError to none(1)."
        ::= { pethPsePortEntry 10 }

    pethPsePortType OBJECT-TYPE
        SYNTAX INTEGER   {
                    other(1),
                    telephone(2),
                    webcam(3),
                    wireless(4)
    }
        MAX-ACCESS read-write
        STATUS current
        DESCRIPTION
            "A manager will set the value of this variable to a value
             that indicates the type of the device that is connected
             to theport. This value can be the result of the mapping
```

```
            the address of the station connected to the port and of
            the value of the pethPdPortType of the respective PD port."
        ::= { pethPsePortEntry 11 }

   -- PD Port table

   pethPdPortTable OBJECT-TYPE
          SYNTAX      SEQUENCE OF PethPdPortEntry
          MAX-ACCESS  not-accessible
          STATUS      current
          DESCRIPTION
              "A table of objects that display and control the power
               characteristics power Ethernet ports on a Powered
               Device(PD) device. This group will be implemented in
               managed powered and mid-span devices."
          ::= { pethObjects 2 }

      pethPdPortEntry OBJECT-TYPE
          SYNTAX      PethPdPortEntry
          MAX-ACCESS  not-accessible
          STATUS      current
          DESCRIPTION
                  "A set of objects that display and control the power
               characteristics of a Powered Device port."
          INDEX    { pethPdPortIndex }
          ::= { pethPdPortTable 1 }

      PethPdPortEntry ::= SEQUENCE {
          pethPdPortIndex
              InterfaceIndex,
          pethPdPortPowerPairs
              INTEGER,
          pethPdPortDetectionOperStatus
              INTEGER,
          pethPdPortType
              INTEGER
      }

   pethPdPortIndex OBJECT-TYPE
          SYNTAX      InterfaceIndex
          MAX-ACCESS  not-accessible
          STATUS      current
          DESCRIPTION
                  "An index value that uniquely identifies an
               interface to a PD device.  The
               interface identified by a particular value of
               this index is the same interface as identified
               by the same value of ifIndex. The mapping
```

```
                between the ifIndex values and the numbering of
                the port on the device is an implementation
                issue."
            ::= { pethPdPortEntry 1 }

    pethPdPortPowerPairs OBJECT-TYPE
        SYNTAX INTEGER   {
                    signal(1),
                    spare(2),
                    both(3)
    }
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "Describes the pairs in use.
             A value of signal(1) menas that the signal pairs
             only are in use.
             A value of spare(2) means that the spare pairs
             only are in use.
             A value of both(3) means that both the signal
             and the spare pairs are inuse."
        ::= { pethPdPortEntry 2 }

    pethPdPortDetectionOperStatus OBJECT-TYPE
        SYNTAX INTEGER   {
                    off(1),
                    receivingPower(2)
    }
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "Describes the operational status of the port detection.
             The value off(1) means that the port does not receive
             power and the detection algorithm might still be operating.
             The value receivingPower(2) means that the port is
             receiving power. "
        ::= { pethPdPortEntry 3 }

    pethPdPortType OBJECT-TYPE
        SYNTAX INTEGER   {
                    other(1),
                    telephone(2),
                    webcam(3),
                    wireless(4)
    }
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
```

```
          "The type of the device. A management application may read
            the value of this variable and use it for setting the
            corresponding value of pethPsePortType of the port that
            connects the device."
        ::= { pethPdPortEntry 4 }

   -- Main PSE Objects

   pethMainPseObjects       OBJECT IDENTIFIER ::= { pethObjects 3 }

   pethMainPsePower OBJECT-TYPE
          SYNTAX      Integer32 (0..65535)
          MAX-ACCESS  read-write
          STATUS      current
          DESCRIPTION
                  "The nominal power of the PSE expressed in Watts."
          ::= { pethMainPseObjects 1 }

   pethMainPseMaxVoltage OBJECT-TYPE
          SYNTAX      Integer32 (0..65535)
          MAX-ACCESS  read-write
          STATUS      current
          DESCRIPTION
                  "The maximum admitted voltage expressed in mV."
          ::= { pethMainPseObjects 2 }

   pethMainPseMinVoltage OBJECT-TYPE
          SYNTAX      Integer32 (0..65535)
          MAX-ACCESS  read-write
          STATUS      current
          DESCRIPTION
                  "The minimal admitted voltage expressed in mV."
          ::= { pethMainPseObjects 3 }

   pethMainPseOperStatus OBJECT-TYPE
          SYNTAX INTEGER   {
                  on(1),
                  off(2),
                  faulty(3)
   }
          MAX-ACCESS  read-only
          STATUS      current
          DESCRIPTION
                  "The operational status of the main PSE."
          ::= { pethMainPseObjects 4 }

   pethMainPseUsagePower OBJECT-TYPE
          SYNTAX      Integer32 (0..65535)
```

```
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Measured usage power expressed in mW."
        ::= { pethMainPseObjects 5 }

   pethMainPseUsageCurrent OBJECT-TYPE
        SYNTAX      Integer32 (0..65535)
        MAX-ACCESS  read-only
        STATUS      current
        DESCRIPTION
                "Measured usage current expressed in mA."
        ::= { pethMainPseObjects 6 }

   pethMainPseUsageThreshold OBJECT-TYPE
        SYNTAX      Integer32 (1..99)
        MAX-ACCESS  read-write
        STATUS      current
        DESCRIPTION
                "The usage threshold expressed in percens for
                 comparing the measured power and initiating
                 an alarm if the threshold is exceeded."
        ::= { pethMainPseObjects 7 }

   --
   -- Notifications Section
   -- (none defined)
   --

   --
   -- Conformance Section
   --
   pethCompliances OBJECT IDENTIFIER ::= { pethConformance 1 }
   pethGroups      OBJECT IDENTIFIER ::= { pethConformance 2 }

   pethCompliance MODULE-COMPLIANCE
       STATUS  current
       DESCRIPTION
               "Describes the requirements for conformance to the
               Power Ethernet MIB."
       MODULE  -- this module
           GROUP   pethPsePortGroup
           DESCRIPTION
               "The pethPsePortGroup is mandatory for systems which
               implement PSE ports."
           GROUP   pethPdPortGroup
           DESCRIPTION
               "The pethPdPortGroup is mandatory for systems which
```

```
                implement PD Ports."
            GROUP   pethMainPseGroup
            DESCRIPTION
                "The pethMainPseGroup is mandatory for systems which
                implement main power supply within a PSE Device."
        ::= { pethCompliances 1 }

    pethPsePortGroup OBJECT-GROUP
        OBJECTS {
           pethPsePortPowerEnable,
           pethPsePortPowerIdPairsControl,
           pethPsePortPowerIdPairs,
           pethPsePortPowerDetectionStatus,
           pethPsePortDetectionOperStatus,
           pethPsePortPowerPriority,
           pethPsePortDenyError,
           pethPsePortFaultError,
           pethPsePortFaultErrorClear,
           pethPsePortType
        }
        STATUS  current
        DESCRIPTION
                "PSE Port objects."
        ::= { pethGroups 1 }

    pethPdPortGroup OBJECT-GROUP
        OBJECTS {
           pethPdPortPowerPairs,
           pethPdPortDetectionOperStatus,
           pethPdPortType
        }
        STATUS  current
        DESCRIPTION
                "PD Port Objects."
        ::= { pethGroups 2 }

    pethMainPseGroup OBJECT-GROUP
        OBJECTS {
           pethMainPsePower,
           pethMainPseMaxVoltage,
           pethMainPseMinVoltage,
           pethMainPseOperStatus,
           pethMainPseUsageCurrent,
           pethMainPseUsagePower,
           pethMainPseUsageThreshold
        }
        STATUS  current
        DESCRIPTION
```

```
            "Main PSE Objects. "
        ::= { pethGroups 3 }
    END
```

7. References

[RFC2571] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing SNMP Management Frameworks", RFC 2571, April 1999.

[RFC1155] Rose, M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", STD 16, RFC 1155, May 1990.

[RFC1212] Rose, M., and K. McCloghrie, "Concise MIB Definitions", STD 16, RFC 1212, March 1991.

[RFC1215] M. Rose, "A Convention for Defining Traps for use with the SNMP", RFC 1215, March 1991.

[RFC2578] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[RFC2579] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[RFC1157] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", STD 15, RFC 1157, May 1990.

[RFC1901] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, January 1996.

[RFC1906] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, January 1996.

[RFC2572] Case, J., Harrington D., Presuhn R., and B. Wijnen, "Message

Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, April 1999.

[RFC2574] Blumenthal, U., and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2574, April 1999.

[RFC1905] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, January 1996.

[RFC2573] Levi, D., Meyer, P., and B. Stewart, "SNMPv3 Applications", RFC 2573, April 1999.

[RFC2575] Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", RFC 2575, April 1999.

[RFC2570] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction to Version 3 of the Internet-standard Network Management Framework", RFC 2570, April 1999.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2665] Flick, J., and J. Johnson, "Definitions of Managed Objects for the Ethernet-like Interface Types", RFC 2665, August 1999.

[IEEE-802.3af] IEEE 802.3af Working Group, "Data Terminal Equipment (DTE) Power via Media Dependent Interface (MDI)", Draft D1.1, January 2001.

8. Intellectual Property

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

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

9. Security Considerations

There are a number of management objects defined in this MIB that have a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations.

There are a number of managed objects in this MIB that may contain sensitive information. These are:

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.

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 [RFC2274] and the View-based Access Control Model [RFC2275] is RECOMMENDED.

It is then a customer/user responsibility to ensure that the SNMP entity giving access to an instance of this MIB, is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

10. Author's Address

Dan Romascanu
Avaya Inc.
Atidim Technology Park, Bldg. #3
Tel Aviv, 61131

Israel
Tel: +972-3-645-8414
Email: dromasca@avaya.com

A. Full Copyright Statement