PTOPOMIB Working Group
Internet Draft

Andy Bierman
Cisco Systems, Inc.
Keith McCloghrie
Cisco Systems, Inc.
16 November 1998

Physical Topology Discovery Protocol and MIB

<draft-ietf-ptopomib-pdp-03.txt>

Status of this Memo

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as ``work in progress.''

To learn the current status of any Internet-Draft, please check the 1id-abstracts.txt listing contained in the Internet- Drafts Shadow Directories on ftp.ietf.org, nic.nordu.net, venera.isi.edu, or munnari.oz.au.

Distribution of this document is unlimited. Please send comments to the PTOPOMIB Working Group, <ptopo@3com.com>.

1. Copyright Notice

Copyright (C) The Internet Society (1998). All Rights Reserved.

2. Abstract

This memo defines an experimental protocol, and an experimental portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes a

physical topology discovery protocol and managed objects used for managing the protocol.

3. Table of Contents

4. The SNMP Network Management Framework

The SNMP Management Framework presently consists of five major components:

- An overall architecture, described in RFC 2271 [RFC2271].
- 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 RFC 1155 [RFC1155], RFC 1212 [RFC1212] and RFC 1215 [RFC1215]. The second version, called SMIv2, is described in RFC 1902 [RFC1902], RFC 1903 [RFC1903] and RFC 1904 [RFC1904].
- Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in 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 2272 [RFC2272] and RFC 2274 [RFC2274].
- Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in RFC 1157 [RFC1157]. A second set of protocol operations and associated PDU formats is described in RFC 1905 [RFC1905].
- A set of fundamental applications described in RFC 2273 [RFC2273] and the view-based access control mechanism described in RFC 2275 [RFC2275].

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 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.

5. Overview

There is a need for a standardized way of representing the physical network connections pertaining to a given management domain. A standardized discovery mechanism is also required to increase the likelihood of multi-vendor interoperability of such physical topology management information.

This document specifies a discovery protocol, suitable for use with the Physical Topology MIB [PTOPO].

5.1. Terms

Some terms are used throughout this document:

SNMP Agent
: This term refers to an SNMP agent co-located with a particular PDP Agent. Specifically, it refers to the SNMP Agent providing PDP MIB, Entity MIB, Interfaces MIB, and possibly PTOPO MIB support for a particular chassis.

PDP Agent
: This term refers to a software entity which implements the PTOPO Discovery Protocol for a particular chassis.

5.2. Persistent Identifiers

The PTOPO MIB utilizes non-volatile identifiers to distinguish individual chassis and port components. These identifiers are associated with external objects in order to relate topology information to the existing managed objects.

In particular, an object from the Entity MIB or Interfaces MIB can be used as the 'reference-point' for a connection component identifier.

5.3. Relationship to the Physical Topology MIB

The Physical Topology MIB [PTOPO] allows a PDP Agent to expose learned physical topology information, using a standard MIB. PDP is intended to

fully support the PTOPO MIB.

5.4. Relationship to Entity MIB

The Entity MIB [RFC2037] allows the physical component inventory and hierarchy to be identified. The chassis identifier strings passed in PDP messages identify entPhysicalTable entries, and implementation of the entPhysicalTable as specified in the Version 1 of the Entity MIB [RFC2037], and implementation of the entPhysicalAlias object from Version 2 of the Entity MIB [ENTITY-MIB], are required for SNMP agents which also implement the PDP MIB.

5.5. Relationship to Interfaces MIB

The Interfaces MIB provides a standard mechanism for managing network interfaces. The port identifier strings passed in PDP messages identify ifTable (or entPhysicalTable) entries, and implementation of the ifTable and ifXTable [RFC2233] are required for SNMP agents which also implement the PDP MIB, for the ports which are represented in the Interfaces MIB.

6. PTOPO Discovery Protocol

This section defines a discovery protocol, suitable for supporting the data requirements of the PTOPO MIB.

The PTOPO Discovery Protocol (PDP) is a media independent protocol intended to be run on routers, bridges, access servers, switches, repeaters, etc., allowing a PDP agent to learn SNMP reachability and connection endpoint information from adjacent devices.

PDP runs on various media that support Subnetwork Access Protocol (SNAP), and runs over the data-link layer only, allowing two systems running different network layer protocols can learn about each other.

Each device configured with an active PDP Agent sends periodic messages to a multicast MAC address on all physical interfaces enabled for PDP transmission, and listens for PDP messages on the same set on interfaces. Each PDP message contains information identifying the source port as a PTOPO connection endpoint identifier. It also contains at least one network address which can be used by an NMS to reach an SNMP agent on the device (via the indicated source port). Each PDP message contains a configurable time-to-live value, which tells the recipient

PDP agent when to discard each element of learned topology information.

6.1. Frame Encapsulation

The following open issues are under consideration by the working group:

An EtherType value must be selected to identify PDP messages transmitted over DIX Ethernet, and IEEE 802.3,802.5 media types (using LLC/SNAP encapsulation).

A multicast MAC address must be selected for the destination address (DA) field in PDP messages transmitted over DIX Ethernet, IEEE 802.3, and IEEE 802.5 media types.

6.2. PDP Forwarding

If at all possible, PDP agents are not supposed to forward PDP messages received on any port. However, some devices, such as repeaters, cannot examine each frame received on an interface or port. Such a device will allow PDP messages to be retransmitted on one or more local ports, and will transmit its own PDP messages on those ports as well. These agents are termed 'forwarding' PDP agents.

PDP agents located on devices which examine each frame before retransmitting it (e.g., routers and bridges), are expected to process received PDP messages and not retransmit them on any local port. These agents are termed 'non-forwarding' PDP agents.

An NMS may find physical topology information about the same physical port, represented by several PTOPO agents. This may occur for one of several reasons, including a mixture of forwarding and non-forwarding PDP agents within a network.

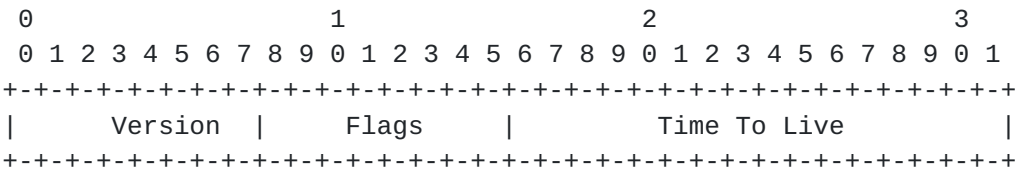
6.3. PDP Message Format

The basic PDP packet consists of a header, followed by a variable number of variable bindings in an ASN.1/BER encoded 'VarBindList', as indicated in Figure 1.

[Figure 1 -- Basic PDP Message Format]

6.3.1. PDP Header Format

The PDP header is a 4 byte header, in network byte order, containing 3 fields, as shown in figure 2:



[figure 2 -- PDP Message Format]

The PDP header contains the following fields:

- Version
  The PDP protocol version number, set to 0x01 for this version of the protocol.

- Flags
  The PDP flags field provide for future header extensions and keep the header word-aligned for easier processing. No flag definition bits are defined at this time. This field must be set to zero in all version 1 PDP messages.

- Time to live
  The number of seconds the information in this PDP message should be regarded as valid by the recipient. Agents of the PTOPO MIB must not return MIB information based on expired PDP messages. The valid range is 0 to 65535 for this field.

6.3.2. PDP PDU Encoding

Following the PDP header is an SNMP varbind list encoded in ASN.1/BER, (as defined in the SMIv2 document [RFC1902]), also referred to as the PDP protocol data unit (PDP-PDU). The individual MIB instances contained in a PDP PDU are referred to as PDP data elements.

The standard PDP data elements, defined in the PDP Data MIB, are encoded as a VarBindList in each PDP message. This data enables a PDP agent to implement the PTOPO MIB for connections terminating on the local chassis.

This section defines the ASN.1 syntax specific to the PDP message. Refer to the Protocol Operations specification [RFC1905] for a complete definition of the 'VarBindList' construct.

Note that PDP places no constraints on which MIB instances may be included in a particular VarBindList.

```
PDP-PDU DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY
        FROM SNMPv2-SMI
    VarBindList
        FROM SNMPv2-PDU;

PDPv1-PDU MODULE-IDENTITY
    LAST-UPDATED "9709150000Z"
    ORGANIZATION "IETF PTOPOMIB Working Group"
    CONTACT-INFO
       "PTOPOMIB WG Discussion:
        ptopo@3com.com
        Subscription:
        majordomo@3com.com
          msg body: [un]subscribe ptopomib

        Andy Bierman
        Cisco Systems Inc.
        170 West Tasman Drive
        San Jose, CA 95134
        408-527-3711
        abierman@cisco.com

        Keith McCloghrie
```

```
        Cisco Systems Inc.
        170 West Tasman Drive
        San Jose, CA 95134
        408-526-5260
        kzm@cisco.com"
    DESCRIPTION
            "The definition module for version 1 of the PTOPO Discovery
            Protocol PDU syntax."
    ::= { experimental xx }

PDP-PDU ::=
    SEQUENCE {
        pdp-variable-bindings
            VarBindList
    }

END
```

6.4. PDP Data MIB

This section defines the standard data elements which may be contained in PDP messages. These elements are defined as MIB objects, but are only intended to be encoded into PDP PDUs, and not intended to be instrumented by an SNMP agent.

The MIB defines six standard data elements:

- Chassis ID
- Chassis ID Type
- Port ID
- Port ID Type
- Management Address Type
- Management Address

6.4.1. Definitions

```
PDP-DATA-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY, OBJECT-TYPE, Integer32
        FROM SNMPv2-SMI
    MODULE-COMPLIANCE, OBJECT-GROUP
        FROM SNMPv2-CONF
```

```
    IANAAddrFamily, PtopoGenAddr, PtopoChassisIdType,
    PtopoChassisId, PtopoPortIdType, PtopoPortId
        FROM PTOPO-MIB;


pdpDataMIB MODULE-IDENTITY
    LAST-UPDATED "9709150000Z"
    ORGANIZATION "IETF PTOPOMIB Working Group"
    CONTACT-INFO
       "PTOPOMIB WG Discussion:
        ptopo@3com.com
        Subscription:
        majordomo@3com.com
          msg body: [un]subscribe ptopomib

        Andy Bierman
        Cisco Systems Inc.
        170 West Tasman Drive
        San Jose, CA 95134
        408-527-3711
        abierman@cisco.com

        Keith McCloghrie
        Cisco Systems Inc.
        170 West Tasman Drive
        San Jose, CA 95134
        408-526-5260
        kzm@cisco.com"
    DESCRIPTION
            "The MIB module for describing PDP data elements."
    ::= { experimental xx }

pdpDataMIBObjects   OBJECT IDENTIFIER ::= { pdpDataMIB 1 }

-- MIB groups
pdpDataElements     OBJECT IDENTIFIER ::= { pdpDataMIBObjects 1 }

--
--  ********************************************************
--
--              P D P    D A T A    E L E M E N T S
--
--  ********************************************************
--
```

```
pdpChassisIdType OBJECT-TYPE
    SYNTAX      PtopoChassisIdType
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object identifies the type of chassis component
            identifier contained in the pdpChassisId object, within a
            given PDP message."
    ::= { pdpDataElements 1 }

pdpChassisId OBJECT-TYPE
    SYNTAX      PtopoChassisId
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object identifies the chassis component of the
            particular connection endpoint identifier containing the PDP
            agent transmitting PDP messages.

            If the chassis ID is unknown for the entry, then this object
            will contain an empty string."
    ::= { pdpDataElements 2 }

pdpPortIdType OBJECT-TYPE
    SYNTAX      PtopoPortIdType
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object identifies the type of port component
            identifier contained in the pdpPortId object, within a given
            PDP message."
    ::= { pdpDataElements 3 }

pdpPortId OBJECT-TYPE
    SYNTAX      PtopoPortId
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object identifies the port component of a particular
            connection endpoint identifier, associated with the port
            chosen for transmission of a given PDP message.

            For PDP agents contained within repeaters or concentrators,
            this object may identify the backplane component chosen for
            transmission of a given PDP message, instead of a specific
```

```
            port component (attached to the identified backplane).

            If the port ID is unknown for the entry, then this object
            will contain an empty string."
    ::= { pdpDataElements 4 }

pdpMgmtAddrType OBJECT-TYPE
    SYNTAX      IANAAddrFamily
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object identifies the type of network address
            contained in the pdpMgmtAddr object, within a given PDP
            message."
    ::= { pdpDataElements 5 }

pdpMgmtAddr OBJECT-TYPE
    SYNTAX      PtopoGenAddr
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object identifies a particular network address,
            associated with an SNMP agent which contains additional
            information pertaining to the connection endpoint identified
            in a given PDP message.

            If a management address is unknown for the endpoint
            described in a given PDP message, then this object will
            contain an empty string."
    ::= { pdpDataElements 6 }



-- conformance information
pdpDataConformance OBJECT IDENTIFIER ::= { pdpDataMIB 2 }

pdpDataCompliances OBJECT IDENTIFIER ::= { pdpDataConformance 1 }
pdpDataGroups      OBJECT IDENTIFIER ::= { pdpDataConformance 2 }

-- compliance statements

pdpDataCompliance MODULE-COMPLIANCE
   STATUS  current
    DESCRIPTION
            "The compliance statement for entities which implement the
```

```
            PTOPO Discovery Protocol."
    MODULE  -- this module
        MANDATORY-GROUPS { pdpPtopoDataGroup }

    ::= { pdpDataCompliances 1 }

-- MIB groupings

pdpPtopoDataGroup    OBJECT-GROUP
    OBJECTS {
        pdpChassisIdType,
        pdpChassisId,
        pdpPortIdType,
        pdpPortId,
        pdpMgmtAddrType,
        pdpMgmtAddr
    }
    STATUS  current
    DESCRIPTION
            "The collection of objects which identify connection
            endpoint data elements, as used in the PTOPO Discovery
            Protocol, and represented in the PTOPO MIB.

            This group is mandatory for agents which implement the PTOPO
            Discovery Protocol."
    ::= { pdpDataGroups 1 }

END
```

6.5. Protocol Operation

An active PDP Agent must perform the following tasks:

- transmit PDP messages
- process received PDP messages
- maintain an instance of the PDP MIB
- maintain an instance of the PTOPO MIB
- maintain appropriate ifEntry and/or entPhysicalEntry instances
- implement ifAlias and/or entPhysicalAlias MIB objects

6.5.1. Protocol Initialization

Upon system reinitialization, the following tasks are performed by the PDP agent:

Non-volatile configuration for the PDP MIB is retrieved if applicable, otherwise appropriate default values are assigned to all PDP configuration variables.

If pdpAdminStatus is equal to 'disabled(2)', then PDP initialization is terminated (until such time that the pdpAdminStatus object is set to 'enabled(1)'), otherwise continue.

Internal (implementation-specific) data structures are initialized. such that appropriate local physical topology information and PDP transmission parameters are set.

6.5.2. Message Encoding

This section does not assume a particular buffering strategy, and such details are omitted.

6.5.2.1. Header Fields

The version field is set to one (0x01).

The flags field is set to zero (0x00).

The time-to-live field is set to the value obtained by the following formula:

TTL = min(65535, (pdpMessageTxInterval * pdpMessageTxHoldMultiplier))

6.5.2.2. VarBindList

Each message must contain one instance of each of the six mandatory PDP-PDU data elements, defined in the PDP Data MIB. Additional data elements may be added as maximum frame size permits.

ASN.1/BER encoding is defined in Version 2 of the Structure of Management Information (SMIv2) [RFC1902], and is outside the scope of this document.

6.5.3. Message Transmission

An active PDP agent must transmit a PDP message out each appropriate port, once each message interval, as determined by the pdpMessageTxInterval MIB object. Messages transmitted on repeater devices may be sent for each repeater backplane, once per message interval. Actual transmission intervals should be jittered to prevent synchronization effects.

Note that the agent must suppress the transmission of multiple PDP messages during a single message interval, in the event message transmission cannot be restricted to a single port, but rather a group of ports (e.g., a repeater device).

In this case, a single port in the port group should be selected (in an implementation-specific manner) to represent the port group. Note that an agent is encouraged to represent port groups as 'backplanes', in the entPhysicalTable of the Entity MIB, rather than individual ports in either the Entity MIB or Interfaces MIB.

Regarding the transmission of a single PDP message, for the indicated physical interface contained in the local system:

The PDP agent checks for the existence of a pdpSuppressEntry for the port. If an entry exists then this port is skipped, otherwise continue.

The PDP message is encapsulated as appropriate for the port.

The MAC header is filled in with appropriate SA and DA and EtherType fields. (Ignoring LLC/SNAP details).

The frame is transmitted or passed to a lower layer for transmission.

The pdpStatsOutPkts counter is incremented for the indicated local port.

6.5.4. Received Message Processing

An active PDP agent must process PDP messages received on each appropriate port, as such messages arrive.

The following sections refer to the reception of a single PDP message,

for the indicated physical interface contained in the local system:

6.5.4.1. Header Fields

The PDP message and the chassis/port indices associated with the receiving port are retrieved.

The PDP version and flags field are checked. The version should equal one (0x01) and the flags should equal zero (0x00). If not, the pdpStatsInErrors counter for the receiving port is incremented and processing is terminated; otherwise continue.

6.5.4.2. VarBindList

The ASN.1/BER portion of the message is decoded. (Such parsing techniques are beyond the scope of this document.) If this portion of the PDP message is not properly encoded, as defined in the PDP Data MIB, then the pdpStatsInErrors counter for the receiving port is incremented, and processing is terminated; otherwise continue.

The list of data elements is examined. The agent must skip and ignore PDU data elements unknown to the agent. If any of the mandatory data elements are missing, then the pdpStatsInErrors counter for the receiving port is incremented, and processing is terminated; otherwise continue.

The pdpStatsInGoodPkts counter is incremented for the receiving port.

6.5.4.3. PTOPO MIB Update

If the time-to-live field in the PDP message header is zero then execute this interface shutdown procedure, described below. Processing of the PDP message is now complete.

If the time-to-live field is non-zero, then the appropriate ptopoConnEntry is found or created, based on the data elements included in the PDP message. If the indicated entry is dynamic (i.e., ptopoConnIsStatic is true), then the current sysUpTime value is stored in the ptopoConnLastVerifyTime field for the entry.

If a ptopoConnEntry was added then the ptopoConnTabInserts counter is incremented.

If a ptopoConnEntry of one type was replaced with an entry of a different type, then the ptopoConnTabReplaces counter is incremented.

If any ptopoConnEntry was added or deleted, or if information other than the ptopoConnLastVerifyTime changed for any entry due to the processing of this PDP message, then the ptopoLastChangeTime object is set with the current sysUpTime, and a ptopoConfigChange trap event is generated. (See the PTOPO MIB for information on ptopoConfigChange trap generation.)

6.5.5. Interface Shutdown Procedure

A special procedure exists for the case in which a PDP agent knows a particular port is about to become non-operational.

Note that the pdpSuppressTable has precedence over these procedures, and they are only executed if the indicated interface is not specified in the pdpSuppressTable.

If any entries are deleted as a result of these procedures, the ptopoConnTabDeletes counter is incremented for each deleted entry.

6.5.5.1. PDP Shutdown Transmission

In the event an interface, currently configured with PDP message transmission enabled, either becomes disabled for PDP activity, or the interface is administratively disabled, a final PDP message is transmitted with a time to live value of zero (before the interface is disabled).

In the event the pdpOperStatus is transitioning to the disabled state, then this shutdown procedure should be executed for all local interfaces.

6.5.5.2. PDP Shutdown Reception

After reception of a valid PDP message with a time-to-live value equal to zero, the PDP Agent must remove all information in the PTOPO MIB learned from the particular PDP agent, which is associated with the indicated remote connection endpoint.

7. PTOPO Discovery Protocol MIB

This section defines the MIB used to configure PDP agent behavior.

7.1. Definitions

```
PDP-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY, OBJECT-TYPE, Integer32, Counter32
        FROM SNMPv2-SMI
    RowStatus
        FROM SNMPv2-TC
    MODULE-COMPLIANCE, OBJECT-GROUP
        FROM SNMPv2-CONF
    PhysicalIndex
        FROM ENTITY-MIB;

pdpMIB MODULE-IDENTITY
    LAST-UPDATED "9707300000Z"
    ORGANIZATION "IETF PTOPOMIB Working Group"
    CONTACT-INFO
       "PTOPOMIB WG Discussion:
        ptopo@3com.com
        Subscription:
        majordomo@3com.com
          msg body: [un]subscribe ptopomib

        Andy Bierman
        Cisco Systems Inc.
        170 West Tasman Drive
        San Jose, CA 95134
        408-527-3711
        abierman@cisco.com

        Keith McCloghrie
        Cisco Systems Inc.
        170 West Tasman Drive
        San Jose, CA 95134
        408-526-5260
        kzm@cisco.com"
    DESCRIPTION
            "The MIB module for managing the Physical Topology Discovery
            Protocol."
```

```
    ::= { experimental xx }

pdpMIBObjects   OBJECT IDENTIFIER ::= { pdpMIB 1 }

-- MIB groups
pdpConfig    OBJECT IDENTIFIER ::= { pdpMIBObjects 1 }
pdpStats     OBJECT IDENTIFIER ::= { pdpMIBObjects 2 }


PdpPortIdType ::=  TEXTUAL-CONVENTION
    STATUS        current
    DESCRIPTION
            "The type of index value used to represent a port component.

            If an object of this type has a value of 'ifIndexType(1)',
            then the associated 'port ID' value represents an ifEntry,
            with the same ifIndex value.

            If an object of this type has a value of
            'entPhysicalIndexType(2)', then the associated 'port ID'
            value represents an entPhysicalEntry, with the same
            entPhysicalIndex value."
    SYNTAX        INTEGER  {
                     ifIndexType(1),
                     entPhysicalIndexType(2)
                  }

--
--  ********************************************************
--
--                    P D P    C O N F I G
--
--  ********************************************************
--
--   The Physical Topology Discovery Protocol Configuration Group

pdpAdminStatus OBJECT-TYPE
    SYNTAX      INTEGER  {
                           enabled(1),
                           disabled(2)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The administratively desired status of the the local PDP
```

```
            agent.

            If the agent is capable of storing non-volatile
            configuration, then the value of this object must be
            restored after a re-initialization of the management
            system."
    ::= { pdpConfig 1 }

pdpOperStatus OBJECT-TYPE
    SYNTAX      INTEGER  {
                        enabled(1),
                        disabled(2)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The current operational status of the local PDP agent."
    ::= { pdpConfig 2 }

pdpMessageTxInterval OBJECT-TYPE
    SYNTAX      Integer32 (5..32768)
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The interval at which PDP messages are transmitted on
            behalf of this PDP agent.

            If the agent is capable of storing non-volatile
            configuration, then the value of this object must be
            restored after a re-initialization of the management
            system."
    DEFVAL       { 60 }
    ::= { pdpConfig 3 }

pdpMessageTxHoldMultiplier OBJECT-TYPE
    SYNTAX      Integer32 (2..10)
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The time-to-live value expressed as a multiple of the
            pdpMessageTxInterval object.  The actual time-to-live value
            used in PDP messages, transmitted on behalf of this PDP
            agent, can be expressed by the following formula:
```

```
            TTL = min(65535, (pdpMessageTxInterval *
pdpMessageTxHoldMultiplier))

            For example, if the value of pdpMessageTxInterval is '60',
            and the value of pdpMessageTxHoldMultiplier is '3', then the
            value '180' is encoded in the TTL field in the PDP header.

            If the agent is capable of storing non-volatile
            configuration, then the value of this object must be
            restored after a re-initialization of the management
            system."
    DEFVAL        { 3 }
    ::= { pdpConfig 4 }

--
-- PdpSuppressTable:
--   Disable PDP activity on individual local ports

pdpSuppressTable   OBJECT-TYPE
    SYNTAX      SEQUENCE OF PdpSuppressEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "A table controlling PDP message transmission on individual
            interfaces, ports, or backplanes."
    ::= { pdpConfig 6 }

pdpSuppressEntry   OBJECT-TYPE
    SYNTAX      PdpSuppressEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "PDP message configuration information for a particular
            port.  The port must be contained in the same chassis as the
            PDP agent. PDP messages will not be transmitted or received
            on the indicated port, even if the port is enabled.

            If the agent is capable of storing non-volatile
            configuration, then each active pdpSuppressEntry must be
            re-created after a re-initialization of the management
            system.  An agent should store enough information about the
            associated entPhysicalEntry (e.g., entPhysicalAlias) or
            ifEntry (e.g. ifAlias), to properly re-create the entry,
            even if the pdpSuppressChassisId and/or pdpSuppressPortId
            values change across a system re-initialization."
    INDEX  {
```

```
        pdpSuppressChassisId,
        pdpSuppressPortIdType,
        pdpSuppressPortId
    }
    ::= { pdpSuppressTable 1 }

PdpSuppressEntry ::= SEQUENCE {
      pdpSuppressChassisId       PhysicalIndex,
      pdpSuppressPortIdType      PdpPortIdType,
      pdpSuppressPortId          Integer32,
      pdpSuppressRowStatus       RowStatus
}

pdpSuppressChassisId  OBJECT-TYPE
    SYNTAX      PhysicalIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The entPhysicalIndex value used to identify the chassis
            component associated with this entry. The associated
            entPhysicalEntry must be active, and the associated
            entPhysicalClass object must be equal to 'chassis(3)'."
    ::= { pdpSuppressEntry 1 }

pdpSuppressPortIdType OBJECT-TYPE
    SYNTAX        PdpPortIdType
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
            "The type of index value contained in the associated
            pdpSuppressPortId object."
    ::= { pdpSuppressEntry 2 }

pdpSuppressPortId    OBJECT-TYPE
    SYNTAX      Integer32  (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The index value used to identify the port component of this
            entry.  The type of index value depends on the
            pdpSuppressPortIdType value for this entry.

            If the associated pdpSuppressPortIdType is equal to
            'ifIndexType(1)', then  this pdpSuppressPortId represents an
            ifEntry with the same ifIndex value.  The associated ifEntry
```

```
            must be active, and represent a physical interface on the
            local chassis.

            If the associated pdpSuppressPortIdType is equal to
            'entPhysicalIndexType(2)', then this pdpSuppressPortId
            represents an entPhysicalEntry with the same
            entPhysicalIndex value.  The associated entPhysicalEntry
            must be active, and the associated entPhysicalClass object
            must be equal to 'port(10)' or 'backplane(4)'.

            Note that some devices, such as repeaters, cannot restrict
            frame transmission to a single port, but rather to a group
            of ports.  In such an event, an agent will disable PDP
            activity on all ports in the port group, if any of the
            individual ports in the group are specified in this table."
    ::= { pdpSuppressEntry 3 }

pdpSuppressRowStatus  OBJECT-TYPE
    SYNTAX        RowStatus
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
            "The status of this entry."
    ::= { pdpSuppressEntry 4 }

--
--  ********************************************************
--
--                   P D P    S T A T S
--
--  ********************************************************
--
-- PDP Stats Group

pdpStatsTable   OBJECT-TYPE
    SYNTAX      SEQUENCE OF PdpStatsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "A table containing PDP statistics for individual ports.

            Entries are not required to exist in this table while the
            pdpAdminStatus or pdpOperStatus objects are equal to
            'disabled(2)'.
```

```
            Entries are not required to exist in this table if a
            corresponding entry (with identical index values) exists in
            the pdpSuppressTable."
    ::= { pdpStats 1 }

pdpStatsEntry   OBJECT-TYPE
    SYNTAX      PdpStatsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "PDP message statistics for a particular port.  The port
            must be contained in the same chassis as the PDP agent."
    INDEX  {
             pdpStatsChassisId,
             pdpStatsPortIdType,
             pdpStatsPortId
    }
    ::= { pdpStatsTable 1 }

PdpStatsEntry ::= SEQUENCE {
      pdpStatsChassisId       PhysicalIndex,
      pdpStatsPortIdType      PdpPortIdType,
      pdpStatsPortId          Integer32,
      pdpStatsInGoodPkts      Counter32,
      pdpStatsInErrors        Counter32,
      pdpStatsOutPkts         Counter32
}

pdpStatsChassisId  OBJECT-TYPE
    SYNTAX      PhysicalIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The entPhysicalIndex value used to identify the chassis
            component associated with this entry. The associated
            entPhysicalEntry must be active, and the associated
            entPhysicalClass object must be equal to 'chassis(3)'."
    ::= { pdpStatsEntry 1 }

pdpStatsPortIdType OBJECT-TYPE
    SYNTAX        PdpPortIdType
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
            "The type of index value contained in the associated
```

```
            pdpStatsPortId object."
    ::= { pdpStatsEntry 2 }

pdpStatsPortId    OBJECT-TYPE
    SYNTAX      Integer32  (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The index value used to identify the port component of this
            entry.  The type of index value depends on the
            pdpStatsPortType value for this entry.

            If the associated pdpStatsPortIdType is equal to
            'ifIndexType(1)', then  this pdpStatsPortId represents an
            ifEntry with the same ifIndex value.  The associated ifEntry
            must be active, and represent a physical interface on the
            local chassis.

            If the associated pdpStatsPortIdType is equal to
            'entPhysicalIndexType(2)', then this pdpStatsPortId
            represents an entPhysicalEntry with the same
            entPhysicalIndex value.  The associated entPhysicalEntry
            must be active, and the associated entPhysicalClass object
            must be equal to 'port(10)' or 'backplane(4)'."
    ::= { pdpStatsEntry 3 }

pdpStatsInGoodPkts  OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
            "The number of valid PDP messages received by this PDP agent
            on the indicated port, while this PDP agent is enabled."
    ::= { pdpStatsEntry 4 }

pdpStatsInErrors  OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
            "The number of invalid PDP messages received by this PDP
            agent on the indicated port, while this PDP agent is
            enabled. A PDP message may be invalid for several reasons,
            including:
               - invalid MAC header; length or DA fields
```

```
                - invalid PDP header; version or flags fields
                - invalid PDP VarBindList ASN.1/BER encoding
                - invalid or missing PDP VarBindList data elements"
    ::= { pdpStatsEntry 5 }

pdpStatsOutPkts  OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
            "The number of PDP messages transmitted by this PDP agent on
            the indicated port."
    ::= { pdpStatsEntry 6 }


-- conformance information
pdpConformance OBJECT IDENTIFIER ::= { pdpMIB 2 }

pdpCompliances OBJECT IDENTIFIER ::= { pdpConformance 1 }
pdpGroups      OBJECT IDENTIFIER ::= { pdpConformance 2 }

-- compliance statements

pdpCompliance MODULE-COMPLIANCE
   STATUS  current
    DESCRIPTION
            "The compliance statement for SNMP entities which implement
            the PDP MIB."
    MODULE  -- this module
        MANDATORY-GROUPS { pdpConfigGroup, pdpStatsGroup }

    ::= { pdpCompliances 1 }

-- MIB groupings

pdpConfigGroup    OBJECT-GROUP
    OBJECTS {
        pdpAdminStatus,
        pdpOperStatus,
        pdpMessageTxInterval,
        pdpMessageTxHoldMultiplier,
        pdpSuppressRowStatus
    }
    STATUS  current
    DESCRIPTION
```

```
            "The collection of objects which are used to configure the
            PTOPO Discovery Protocol implementation behavior.

            This group is mandatory for agents which implement the PTOPO
            Discovery Protocol."
    ::= { pdpGroups 1 }

pdpStatsGroup    OBJECT-GROUP
    OBJECTS {
        pdpStatsInGoodPkts,
        pdpStatsInErrors,
        pdpStatsOutPkts
    }
    STATUS  current
    DESCRIPTION
            "The collection of objects which are used to represent PTOPO
            Discovery Protocol statistics.

            This group is mandatory for agents which implement the PTOPO
            Discovery Protocol."
    ::= { pdpGroups 2 }

END
```

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.

The IETF has been notified of intellectual property rights claimed in regard to some or all of the specification contained in this document. For more information consult the online list of claimed rights.

9. Acknowledgements

The PTOPO Discovery Protocol is a product of the IETF PTOPOMIB Working Group.

10. References

[ENTITY-MIB]
McCloghrie, K., and A. Bierman, "Entity MIB using SMIv2 (Version 2)", draft-ietf-entmib-v2-01.txt, Cisco Systems, November 1998.

[PTOPO]
Bierman, A., and K. Jones, "Physical Topology MIB", draft-ietf-ptopomib-mib-01.txt, Cisco Systems, Bay Networks, November 1998.

[RFC1155]
Rose, M., and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets", RFC 1155, Performance Systems International, Hughes LAN Systems, May 1990.

[RFC1157]
Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", RFC 1157, SNMP Research, Performance Systems International, Performance Systems International, MIT Laboratory for Computer Science, May 1990.

[RFC1212]
Rose, M., and K. McCloghrie, "Concise MIB Definitions", RFC 1212, Performance Systems International, Hughes LAN Systems, March 1991.

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

[RFC1901]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC1902]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1902, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC1903]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Textual Conventions for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1903, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC1904]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Conformance Statements for version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1904, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC1905]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC1906]
SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)"", RFC 1906, SNMP Research, Inc., Cisco Systems, Inc., Dover Beach Consulting, Inc., International Network Services, January 1996.

[RFC2037]
McCloghrie, K., and A. Bierman, "Entity MIB using SMIv2", RFC 2037, Cisco Systems, October 1996.

[RFC2233]
McCloghrie, K., and F. Kastenholtz, "The Interfaces Group MIB using SMIv2", RFC 2233, Cisco Systems, FTP Software, November 1997.

[RFC2271]
Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for

Describing SNMP Management Frameworks", RFC 2271, Cabletron Systems, Inc., BMC Software, Inc., IBM T. J. Watson Research, January 1998.

[RFC2272]
Case, J., Harrington D., Presuhn R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2272, SNMP Research, Inc., Cabletron Systems, Inc., BMC Software, Inc., IBM T. J. Watson Research, January 1998.

[RFC2273]
Levi, D., Meyer, P., and B. Stewart, "SNMPv3 Applications", RFC 2273, SNMP Research, Inc., Secure Computing Corporation, Cisco Systems, January 1998.

[RFC2274]
Blumenthal, U., and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2274, IBM T. J. Watson Research, January 1998.

[RFC2275]
Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", RFC 2275, IBM T. J. Watson Research, BMC Software, Inc., Cisco Systems, Inc., January 1998.

11. Security Considerations

This protocol and associated MIB can expose the existence of physical components, MAC layer addresses, and network layer addresses, pertaining to devices within a given network. A network administrator may wish to restrict access to this management information, using SNMP access control mechanisms, and restrict PDP message processing to a particular set of ports, by configuring entries in the pdpSuppressTable.

12. Authors' Addresses

Andy Bierman
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA USA 95134
Phone: +1 408-527-3711
Email: abierman@cisco.com

Keith McCloghrie

Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA USA 95134
Phone: +1 408-526-5260
Email: kzm@cisco.com

13. Full Copyright Statement

Copyright (C) The Internet Society (1998). All Rights Reserved.

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

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

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