The Definitions of Managed Objects for
the Link Control Protocol of
the Point-to-Point Protocol

Status of this Memo

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

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in TCP/IP-based internets. In particular, it describes managed objects used for managing the Link Control Protocol and Link Quality Monitoring on subnetwork interfaces that use the family of Point-to-Point Protocols [8, 9, 10, 11, & 12].

Table of Contents

1. The Network Management Framework ...................... 2
2. Objects ............................................ 2
2.1 Format of Definitions ................................ 2
3. Overview ........................................... 2
3.1 Object Selection Criteria ............................ 2
3.2 Structure of the PPP ............................... 3
3.3 MIB Groups ....................................... 4
3.4 Relationship to Interface and Interface Extensions
   Groups ............................................... 5
4. Definitions ......................................... 6
4.1 PPP Link Group .................................... 7
4.2 PPP LQR Group .................................... 16
4.3 PPP LQR Extensions Group ........................... 21
4.4 PPP Tests ........................................ 22
4.4.1 PPP Echo Test ................................ 22
4.4.2 PPP Discard Test .............................. 23
5. Acknowledgements ................................... 23
6. Security Considerations ............................... 23
7. References ......................................... 24
8. Author’s Address ................................... 25
1. The Network Management Framework

The Internet-standard Network Management Framework consists of three components. They are:

STD 16/RFC 1155 which defines the SMI, the mechanisms used for describing and naming objects for the purpose of management. STD 16/RFC 1212 defines a more concise description mechanism, which is wholly consistent with the SMI.

STD 17/RFC 1213 which defines MIB-II, the core set of managed objects for the Internet suite of protocols.

STD 15/RFC 1157 which defines the SNMP, the protocol used for network access to managed objects.

The Framework permits new objects to be defined for the purpose of experimentation and evaluation.

2. Objects

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the subset of Abstract Syntax Notation One (ASN.1) defined in the SMI. In particular, each object type is named by an OBJECT IDENTIFIER, an administratively assigned name. The object type together with an object instance serves to uniquely identify a specific instantiation of the object. For human convenience, we often use a textual string, termed the descriptor, to refer to the object type.

2.1. Format of Definitions

Section 4 contains the specification of all object types contained in this MIB module. The object types are defined using the conventions defined in the SMI, as amended by the extensions specified in [5,6].

3. Overview

3.1. Object Selection Criteria

To be consistent with IAB directives and good engineering practice, an explicit attempt was made to keep this MIB as simple as possible. This was accomplished by applying the following criteria to objects proposed for inclusion:

(1) Require objects be essential for either fault or configuration management. In particular, objects for
which the sole purpose was to debug implementations were explicitly excluded from the MIB.

(2) Consider evidence of current use and/or utility.

(3) Limit the total number of objects.

(4) Exclude objects which are simply derivable from others in this or other MIBs.

3.2. Structure of the PPP

This section describes the basic model of PPP used in developing the PPP MIB. This information should be useful to the implementor in understanding some of the basic design decisions of the MIB.

The PPP is not one single protocol but a large family of protocols. Each of these is, in itself, a fairly complex protocol. The PPP protocols may be divided into three rough categories:

Control Protocols
The Control Protocols are used to control the operation of the PPP. The Control Protocols include the Link Control Protocol (LCP), the Password Authentication Protocol (PAP), the Link Quality Report (LQR), and the Challenge Handshake Authentication Protocol (CHAP).

Network Protocols
The Network Protocols are used to move the network traffic over the PPP interface. A Network Protocol encapsulates the datagrams of a specific higher-layer protocol that is using the PPP as a data link. Note that within the context of PPP, the term "Network Protocol" does not imply an OSI Layer-3 protocol; for instance, there is a Bridging network protocol.

Network Control Protocols (NCPs)
The NCPs are used to control the operation of the Network Protocols. Generally, each Network Protocol has its own Network Control Protocol; thus, the IP Network Protocol has its IP Control Protocol, the Bridging Network Protocol has its Bridging Network Control Protocol and so on.

This document specifies the objects used in managing one of these protocols, namely the Link Control Protocol and Link Quality Monitoring Protocol.
3.3. MIB Groups

Objects in this MIB are arranged into several MIB groups. Each group is organized as a set of related objects.

These groups are the basic unit of conformance: if the semantics of a group are applicable to an implementation then all objects in the group must be implemented.

The PPP MIB is organized into several MIB Groups, including, but not limited to, the following groups:

- The PPP Link Group
- The PPP LQR Group
- The PPP LQR Extensions Group
- The PPP IP Group
- The PPP Bridge Group
- The PPP Security Group

This document specifies the following groups:

The PPP Link Group
   This group represents the lowest "level" of the PPP protocol.

   This group contains two tables, one containing status information and the other configuration information. The configuration table is split off of the status so that it may be placed in a separate MIB View for security purposes.

   Implementation of this group is mandatory for all PPP implementations.

The PPP LQR Group
   This group provides the basic MIB variables that apply to the PPP LQR Protocol. This group provides MIB access to the information required for LQR processing. This group contains two tables, one containing status information and the other configuration information. The configuration table is split off of the status so that it may be placed in a separate MIB View for security purposes.

   Implementation of the PPP LQR Group is mandatory for all PPP implementations that implement LQR.

The PPP LQR Extensions Group
   The PPP LQR Extensions group contains the most recently received LQR packet, as well as the "save" fields that are "logically appended" [12] to received LQR packets. This is done in order to
facilitate external implementations of the Link Quality Monitoring policies.

It is not practical to examine the relevant MIB objects which are used to generate LQR packets since LQR policies may require synchronization of the values of all data used to determine Link Quality; i.e., the values of the relevant counters must all be taken at the same instant in time. Thus, by recording the last received LQR packet, a synchronized record of the relevant data is available.

As this information may not be efficiently maintained on all PPP implementations, implementation of this group is optional.

3.4. Relationship to Interface and Interface Extensions Groups

The PPP Mib is a medium-specific extension to the standard MIB-2 interface group [2] and to the Interface Extensions MIB [7]. This section discusses certain components of these groups when the interface is a PPP interface.

The PPP interface represents a single interface in the sense used in [2] and thus has a single entry in the ifTable.

Furthermore, the PPP interface may be operating over a lower layer hardware interface (such as an RS-232 port). It is important to capture the relationship between the PPP interface and the lower-layer interface over which it operates. This MIB presumes that the lower-layer interface has an ifEntry associated with it. The lower-layer ifEntry is identified via the pppLinkStatusPhysicalIndex object, which contains the value of ifIndex for the lower-layer ifEntry.

For example, suppose that you run PPP over a RS-232 port. This would use two entries in the ifTable. Let’s suppose that entry number 123 is for the PPP "interface" and entry number 987 is for the RS-232 port. So, ifSpecific.123 would contain the ppp OBJECT IDENTIFIER, pppLinkStatusPhysicalIndex.123 would contain 987, and ifSpecific.987 would contain the rs_232 OBJECT IDENTIFIER (or whatever it is).

All PPP packets are defined in [8] as being broadcast packets. Thus, the packets are counted as non-unicast packets in the ifTable (ifInNUcastPkts and ifOutNUCastPkts) and as broadcasts in the ifExtnsTable (ifExtnsBroadcastsReceivedOks and ifExtnsBroadcastsTransmittedOks).
ifSpecific
Contains the OBJECT IDENTIFIER ppp.

ifAdminStatus
Setting this object to up will inject an administrative open event into the LCP’s finite state machine. Setting this object to down will inject an administrative close event into the LCP’s finite state machine.

The use of the testing value is beyond the scope of this document.

ifOperStatus
Represents the state of the LCP Finite State Machine. If the Finite State Machine is in the Opened state then the value of ifOperStatus is up, otherwise the value of ifOperStatus is down.

The meaning of the testing value is beyond the scope of this document.

Per the SNMP Protocol Specification [13], the linkUp and linkDown traps apply to the PPP Protocol entity. When the LCP’s Finite State Machine attains the Opened state, a linkUp trap should be sent. When the Finite State Machine leaves the Opened state, a linkDown trap should be sent.

Some tests for the link are defined in this document. Execution of these tests does not place the link’s ifOperStatus in the testing state as these tests do not prevent normal data transmission from occuring over the link.

4. Definitions

PPP-LCP-MIB DEFINITIONS ::= BEGIN

IMPORTS
    Counter
FROM RFC1155-SMI
    ifIndex, transmission
FROM RFC1213-MIB
OBJECT-TYPE
    FROM RFC-1212;

-- PPP MIB

ppp  OBJECT IDENTIFIER ::= { transmission 23 }

pppLcp OBJECT IDENTIFIER ::= { ppp 1 }
-- The individual groups within the PPP-LCP-MIB

pppLink OBJECT IDENTIFIER ::= { pppLcp 1 }
pppLqr OBJECT IDENTIFIER ::= { pppLcp 2 }
pppTests OBJECT IDENTIFIER ::= { pppLcp 3 }

-- 4.1. PPP Link Group

-- The PPP Link Group. Implementation of this
group is mandatory for all PPP entities.

-- The following object reflect the values of the option
-- parameters used in the PPP Link Control Protocol
--  pppLinkStatusLocalMRU
--  pppLinkStatusRemoteMRU
--  pppLinkStatusLocalToPeerACCMap
--  pppLinkStatusPeerToLocalACCMap
--  pppLinkStatusLocalToRemoteProtocolCompression
--  pppLinkStatusRemoteToLocalProtocolCompression
--  pppLinkStatusLocalToRemoteACCompression
--  pppLinkStatusRemoteToLocalACCompression
--  pppLinkStatusTransmitFcsSize
--  pppLinkStatusReceiveFcsSize

-- These values are not available until after the PPP Option
-- negotiation has completed, which is indicated by the link
-- reaching the open state (i.e., ifOperStatus is set to
-- up).

-- Therefore, when ifOperStatus is not up
-- the contents of these objects is undefined. The value
-- returned when accessing the objects is an implementation
-- dependent issue.

pppLinkStatusTable OBJECT-TYPE
SYNTAX     SEQUENCE OF PppLinkStatusEntry
ACCESS     not-accessible
STATUS     mandatory
DESCRIPTION
  "A table containing PPP-link specific variables
   for this PPP implementation."
 ::= { pppLink 1 }
pppLinkStatusEntry OBJECT-TYPE
SYNTAX    PppLinkStatusEntry
ACCESS    not-accessible
STATUS    mandatory
DESCRIPTION
   "Management information about a particular PPP
   Link."
INDEX     { ifIndex }
::= { pppLinkStatusTable 1 }

PppLinkStatusEntry ::= SEQUENCE {
   pppLinkStatusPhysicalIndex        INTEGER,
   pppLinkStatusBadAddresses         Counter,
   pppLinkStatusBadControls          Counter,
   pppLinkStatusPacketTooLongs       Counter,
   pppLinkStatusBadFCSs              Counter,
   pppLinkStatusLocalMRU             INTEGER,
   pppLinkStatusRemoteMRU            INTEGER,
   pppLinkStatusLocalToPeerACCMap    OCTET STRING,
   pppLinkStatusPeerToLocalACCMap    OCTET STRING,
   pppLinkStatusLocalToRemoteProtocolCompression INTEGER,
   pppLinkStatusRemoteToLocalProtocolCompression INTEGER,
   pppLinkStatusLocalToRemoteACCompression INTEGER,
   pppLinkStatusRemoteToLocalACCompression INTEGER,
   pppLinkStatusTransmitFcsSize      INTEGER,
   pppLinkStatusReceiveFcsSize       INTEGER
}

pppLinkStatusPhysicalIndex OBJECT-TYPE
SYNTAX    INTEGER(0..2147483647)
ACCESS    read-only
STATUS    mandatory
DESCRIPTION
"The value of ifIndex that identifies the lower-level interface over which this PPP Link is operating. This interface would usually be an HDLC or RS-232 type of interface. If there is no lower-layer interface element, or there is no ifEntry for the element, or the element cannot be identified, then the value of this object is 0. For example, suppose that PPP is operating over a serial port. This would use two entries in the ifTable. The PPP could be running over 'interface' number 123 and the serial port could be running over 'interface' number 987. Therefore, ifSpecific.123 would contain the OBJECT IDENTIFIER ppp pppLinkStatusPhysicalIndex.123 would contain 987, and ifSpecific.987 would contain the OBJECT IDENTIFIER for the serial-port’s media-specific MIB."

::= { pppLinkStatusEntry 1 }

pppLinkStatusBadAddresses OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of packets received with an incorrect Address Field. This counter is a component of the ifInErrors variable that is associated with the interface that represents this PPP Link."
REFERENCE
"Section 3.1, Address Field, of RFC1331."
::= { pppLinkStatusEntry 2 }

pppLinkStatusBadControls OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of packets received on this link with an incorrect Control Field. This counter is a component of the ifInErrors variable that is associated with the interface that represents this PPP Link."
REFERENCE
"Section 3.1, Control Field, of RFC1331."
::= { pppLinkStatusEntry 3 }

pppLinkStatusPacketTooLongs OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of received packets that have been
discarded because their length exceeded the
MRU. This counter is a component of the
ifInErrors variable that is associated with the
interface that represents this PPP Link. NOTE,
packets which are longer than the MRU but which
are successfully received and processed are NOT
included in this count."
::= { pppLinkStatusEntry 4 }

pppLinkStatusBadFCSs OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of received packets that have been
discarded due to having an incorrect FCS. This
counter is a component of the ifInErrors
variable that is associated with the interface
that represents this PPP Link."
::= { pppLinkStatusEntry 5 }

pppLinkStatusLocalMRU OBJECT-TYPE
SYNTAX INTEGER(1..2147483648)
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The current value of the MRU for the local PPP
Entity. This value is the MRU that the remote
entity is using when sending packets to the
local PPP entity. The value of this object is
meaningful only when the link has reached the
open state (ifOperStatus is up)."
::= { pppLinkStatusEntry 6 }

pppLinkStatusRemoteMRU OBJECT-TYPE
SYNTAX INTEGER(1..2147483648)
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The current value of the MRU for the remote PPP Entity. This value is the MRU that the local entity is using when sending packets to the remote PPP entity. The value of this object is meaningful only when the link has reached the open state (ifOperStatus is up)."
::= { pppLinkStatusEntry 7 }

pppLinkStatusLocalToPeerACCMap OBJECT-TYPE
SYNTAX OCTET STRING (SIZE (4))
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The current value of the ACC Map used for sending packets from the local PPP entity to the remote PPP entity. The value of this object is meaningful only when the link has reached the open state (ifOperStatus is up)."
::= { pppLinkStatusEntry 8 }

pppLinkStatusPeerToLocalACCMap OBJECT-TYPE
SYNTAX OCTET STRING (SIZE (4))
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The ACC Map used by the remote PPP entity when transmitting packets to the local PPP entity. The value of this object is meaningful only when the link has reached the open state (ifOperStatus is up)."
::= { pppLinkStatusEntry 9 }

pppLinkStatusLocalToRemoteProtocolCompression OBJECT-TYPE
SYNTAX INTEGER {
  enabled(1),
  disabled(2)
}
ACCESS read-only
STATUS mandatory
DESCRIPTION
"Indicates whether the local PPP entity will
use Protocol Compression when transmitting packets to the remote PPP entity. The value of this object is meaningful only when the link has reached the open state (ifOperStatus is up).

::= { pppLinkStatusEntry 10 }

pppLinkStatusRemoteToLocalProtocolCompression

OBJECT-TYPE
SYNTAX INTEGER {
   enabled(1),
   disabled(2)
}
ACCESS read-only
STATUS mandatory
DESCRIPTION
"Indicates whether the remote PPP entity will use Protocol Compression when transmitting packets to the local PPP entity. The value of this object is meaningful only when the link has reached the open state (ifOperStatus is up)."
::= { pppLinkStatusEntry 11 }

pppLinkStatusLocalToRemoteACCompression

OBJECT-TYPE
SYNTAX INTEGER {
   enabled(1),
   disabled(2)
}
ACCESS read-only
STATUS mandatory
DESCRIPTION
"Indicates whether the local PPP entity will use Address and Control Compression when transmitting packets to the remote PPP entity. The value of this object is meaningful only when the link has reached the open state (ifOperStatus is up)."
::= { pppLinkStatusEntry 12 }

pppLinkStatusRemoteToLocalACCompression

OBJECT-TYPE
SYNTAX INTEGER {
   enabled(1),
   disabled(2)
}
PPP/LCP MIB                      June 1993

ACCESS    read-only
STATUS    mandatory
DESCRIPTION

"Indicates whether the remote PPP entity will
use Address and Control Compression when
transmitting packets to the local PPP entity.
The value of this object is meaningful only
when the link has reached the open state
(ifOperStatus is up)."
::= { pppLinkStatusEntry 13 }

pppLinkStatusTransmitFcsSize   OBJECT-TYPE
SYNTAX    INTEGER (0..128)
ACCESS    read-only
STATUS    mandatory
DESCRIPTION

"The size of the Frame Check Sequence (FCS) in
bits that the local node will generate when
sending packets to the remote node. The value
of this object is meaningful only when the link
has reached the open state (ifOperStatus is
up)."
::= { pppLinkStatusEntry 14 }

pppLinkStatusReceiveFcsSize   OBJECT-TYPE
SYNTAX    INTEGER (0..128)
ACCESS    read-only
STATUS    mandatory
DESCRIPTION

"The size of the Frame Check Sequence (FCS) in
bits that the remote node will generate when
sending packets to the local node. The value
of this object is meaningful only when the link
has reached the open state (ifOperStatus is
up)."
::= { pppLinkStatusEntry 15 }

pppLinkConfigTable   OBJECT-TYPE
SYNTAX    SEQUENCE OF PppLinkConfigEntry
ACCESS    not-accessible
STATUS    mandatory
DESCRIPTION

"A table containing the LCP configuration
parameters for this PPP Link. These variables
represent the initial configuration of the PPP
Link. The actual values of the parameters may be changed when the link is brought up via the LCP options negotiation mechanism."
::= { pppLink 2 }

pppLinkConfigEntry OBJECT-TYPE
SYNTAX PppLinkConfigEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION "Configuration information about a particular PPP Link."
INDEX { ifIndex }
::= { pppLinkConfigTable 1 }

PppLinkConfigEntry ::= SEQUENCE {
  pppLinkConfigInitialMRU INTEGER,
  pppLinkConfigReceiveACCMap OCTET STRING,
  pppLinkConfigTransmitACCMap OCTET STRING,
  pppLinkConfigMagicNumber INTEGER,
  pppLinkConfigFcsSize INTEGER
}

pppLinkConfigInitialMRU OBJECT-TYPE
SYNTAX INTEGER(0..2147483647)
ACCESS read-write
STATUS mandatory
DESCRIPTION "The initial Maximum Receive Unit (MRU) that the local PPP entity will advertise to the remote entity. If the value of this variable is 0 then the local PPP entity will not advertise any MRU to the remote entity and the default MRU will be assumed. Changing this object will have effect when the link is next restarted."
REFERENCE "Section 7.2, Maximum Receive Unit of RFC1331."
DEFVAL { 1500 }
::= { pppLinkConfigEntry 1 }
pppLinkConfigReceiveACCMap OBJECT-TYPE
SYNTAX OCTET STRING (SIZE (4))
ACCESS read-write
STATUS mandatory
DESCRIPTION "The Asynchronous-Control-Character-Map (ACC) that the local PPP entity requires for use on its receive side. In effect, this is the ACC Map that is required in order to ensure that the local modem will successfully receive all characters. The actual ACC map used on the receive side of the link will be a combination of the local node’s pppLinkConfigReceiveACCMap and the remote node’s pppLinkConfigTransmitACCMap. Changing this object will have effect when the link is next restarted."

REFERENCE "Section 7.3, page 4, Async-Control-Character-Map of RFC1331."
DEFVAL { 'ffffffff'h }
::= { pppLinkConfigEntry 2 }

pppLinkConfigTransmitACCMap OBJECT-TYPE
SYNTAX OCTET STRING (SIZE (4))
ACCESS read-write
STATUS mandatory
DESCRIPTION "The Asynchronous-Control-Character-Map (ACC) that the local PPP entity requires for use on its transmit side. In effect, this is the ACC Map that is required in order to ensure that all characters can be successfully transmitted through the local modem. The actual ACC map used on the transmit side of the link will be a combination of the local node’s pppLinkConfigTransmitACCMap and the remote node’s pppLinkConfigReceiveACCMap. Changing this object will have effect when the link is next restarted."

REFERENCE "Section 7.3, page 4, Async-Control-Character-Map of RFC1331."
DEFVAL { 'ffffffff'h }
::= { pppLinkConfigEntry 3 }
pppLinkConfigMagicNumber OBJECT-TYPE
SYNTAX INTEGER {false (1), true (2)}
ACCESS read-write
STATUS mandatory
DESCRIPTION "If true(2) then the local node will attempt to perform Magic Number negotiation with the remote node. If false(1) then this negotiation is not performed. In any event, the local node will comply with any magic number negotiations attempted by the remote node, per the PPP specification. Changing this object will have effect when the link is next restarted."
REFERENCE "Section 7.6, Magic Number, of RFC1331."
DEFVAL { false }
::= { pppLinkConfigEntry 4 }

pppLinkConfigFcsSize OBJECT-TYPE
SYNTAX INTEGER (0..128)
ACCESS read-write
STATUS mandatory
DESCRIPTION "The size of the FCS, in bits, the local node will attempt to negotiate for use with the remote node. Regardless of the value of this object, the local node will comply with any FCS size negotiations initiated by the remote node, per the PPP specification. Changing this object will have effect when the link is next restarted."
DEFVAL { 16 }
::= { pppLinkConfigEntry 5 }

-- 4.2. PPP LQR Group

--
-- The PPP LQR Group.
-- Implementation of this group is mandatory for all
-- PPP implementations that implement LQR.
--

pppLqrTable OBJECT-TYPE
SYNTAX SEQUENCE OF PppLqrEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION "Table containing the LQR parameters and statistics for the local PPP entity."
::= { pppLqr 1 }

pppLqrEntry OBJECT-TYPE
SYNTAX PppLqrEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION "LQR information for a particular PPP link. A PPP link will have an entry in this table if and only if LQR Quality Monitoring has been successfully negotiated for said link."
INDEX { ifIndex }
::= { pppLqrTable 1 }

PppLqrEntry ::= SEQUENCE {
    pppLqrQuality INTEGER,
    pppLqrInGoodOctets Counter,
    pppLqrLocalPeriod INTEGER,
    pppLqrRemotePeriod INTEGER,
    pppLqrOutLQRs Counter,
    pppLqrInLQRs Counter
}

pppLqrQuality OBJECT-TYPE
SYNTAX INTEGER {
    good(1),
    bad(2),
    not-determined(3)
}
ACCESS read-only
STATUS mandatory
DESCRIPTION "The current quality of the link as declared by the local PPP entity’s Link-Quality Management modules. No effort is made to define good or bad, nor the policy used to determine it. The
not-determined value indicates that the entity does not actually evaluate the link’s quality. This value is used to disambiguate the "determined to be good" case from the "no determination made and presumed to be good" case.

::= { pppLqrEntry 1 }

pppLqrInGoodOctets OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION "The LQR InGoodOctets counter for this link."
REFERENCE "Section 2.2, Counters, of RFC1333."
::= { pppLqrEntry 2 }

pppLqrLocalPeriod OBJECT-TYPE
SYNTAX INTEGER(1..2147483648)
ACCESS read-only
STATUS mandatory
DESCRIPTION "The LQR reporting period, in hundredths of a second that is in effect for the local PPP entity."
REFERENCE "Section 2.5, Configuration Option Format, of RFC1333."
::= { pppLqrEntry 3 }

pppLqrRemotePeriod OBJECT-TYPE
SYNTAX INTEGER(1..2147483648)
ACCESS read-only
STATUS mandatory
DESCRIPTION "The LQR reporting period, in hundredths of a second, that is in effect for the remote PPP entity."
REFERENCE "Section 2.5, Configuration Option Format, of RFC1333."
::= { pppLqrEntry 4 }
pppLqrOutLQRs OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION "The value of the OutLQRs counter on the local node for the link identified by ifIndex."
REFERENCE "Section 2.2, Counters, of RFC1333."
::= { pppLqrEntry 5 }

pppLqrInLQRs OBJECT-TYPE
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION "The value of the InLQRs counter on the local node for the link identified by ifIndex."
REFERENCE "Section 2.2, Counters, of RFC1333."
::= { pppLqrEntry 6 }

--
-- The PPP LQR Configuration table.
--

pppLqrConfigTable OBJECT-TYPE
SYNTAX SEQUENCE OF PppLqrConfigEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION "Table containing the LQR Configuration parameters for the local PPP entity."
::= { pppLqr 2 }

pppLqrConfigEntry OBJECT-TYPE
SYNTAX PppLqrConfigEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION "LQR configuration information for a particular PPP link."
INDEX { ifIndex }
::= { pppLqrConfigTable 1 }
PppLqrConfigEntry ::= SEQUENCE {
    pppLqrConfigPeriod
        INTEGER,
    pppLqrConfigStatus
        INTEGER
}

pppLqrConfigPeriod   OBJECT-TYPE
SYNTAX    INTEGER(0..2147483647)
ACCESS    read-write
STATUS    mandatory
DESCRIPTION
    "The LQR Reporting Period that the local PPP
    entity will attempt to negotiate with the
    remote entity, in units of hundredths of a
    second. Changing this object will have effect
    when the link is next restarted."
REFERENCE
    "Section 2.5, Configuration Option Format, of
    RFC1333."
DEFVAL    { 0 }
 ::= { pppLqrConfigEntry 1 }

pppLqrConfigStatus   OBJECT-TYPE
SYNTAX    INTEGER {disabled (1), enabled (2)}
ACCESS    read-write
STATUS    mandatory
DESCRIPTION
    "If enabled(2) then the local node will attempt
    to perform LQR negotiation with the remote
    node. If disabled(1) then this negotiation is
    not performed. In any event, the local node
    will comply with any magic number negotiations
    attempted by the remote node, per the PPP
    specification. Changing this object will have
    effect when the link is next restarted.
    Setting this object to the value disabled(1)
    has the effect of invalidating the
    corresponding entry in the pppLqrConfigTable
    object. It is an implementation-specific matter
    as to whether the agent removes an invalidated
    entry from the table. Accordingly, management
    stations must be prepared to receive tabular
    information from agents that corresponds to
    entries not currently in use."
REFERENCE
    "Section 7.6, Magic Number, of RFC1331."
DEFVAL { enabled } ::= { pppLqrConfigEntry 2 }

-- 4.3. PPP LQR Extensions Group

-- The PPP LQR Extensions Group.
-- Implementation of this group is optional.
-- The intent of this group is to allow external
-- implementation of the policy mechanisms that
-- are used to declare a link to be "bad" or not.
-- It is not practical to examine the MIB objects
-- which are used to generate LQR packets since
-- LQR policies tend to require synchronization of
-- the values of all data used to determine Link
-- Quality; i.e. the values of the relevant counters
-- must all be taken at the same instant in time.

pppLqrExtnsTable OBJECT-TYPE
SYNTAX SEQUENCE OF PppLqrExtnsEntry
ACCESS not-accessible
STATUS mandatory
DESCRIPTION "Table containing additional LQR information for the local PPP entity."
 ::= { pppLqr 3 }

PppLqrExtnsEntry ::= SEQUENCE {

Kastenholz
pppLqrExtnsLastReceivedLqrPacket
   OCTET STRING(SIZE(68))
}

pppLqrExtnsLastReceivedLqrPacket  OBJECT-TYPE
SYNTAX    OCTET STRING(SIZE(68))
ACCESS    read-only
STATUS    mandatory
DESCRIPTION
   "This object contains the most recently
   received LQR packet. The format of the packet
   is as described in the LQM Protocol
   specification. All fields of the packet,
   including the 'save' fields, are stored in this
   object.

   The LQR packet is stored in network byte order.
   The LAP-B and PPP headers are not stored in
   this object; the first four octets of this
   variable contain the Magic-Number field, the
   second four octets contain the LastOutLQRs
   field and so on. The last four octets of this
   object contain the SaveInOctets field of the
   LQR packet."

REFERENCE
   "Section 2.6, Packet Format, of RFC1333"
::= { pppLqrExtnsEntry 1 }

-- 4.4. PPP Tests
-- The extensions to the interface table in RFC1229 define a
-- table through which the network manager can instruct the
-- managed object to perform various tests of the interface. This
-- is the ifExtnsTestTable.

-- The PPP MIB defines two such tests.

-- 4.4.1. PPP Echo Test
-- The PPP Echo Test is defined as

   pppEchoTest    OBJECT IDENTIFIER ::= { pppTests 1 }

-- Invoking this test causes a PPP Echo Packet to be sent on the
-- line. ifExtnsTestResult returns success(2) if the echo
-- response came back properly. It returns failed(7) if the
-- response did not properly return. The definition of "proper"
-- in this context is left to the discretion of the implementor.

4.4.2. PPP Discard Test

The PPP Discard Test is defined as

```
pppDiscardTest OBJECT IDENTIFIER ::= { pppTests 2 }
```

Invoking this test causes a PPP Discard Packet to be sent on the line. ifExtnsTestResult returns success(2) if the discard packet was successfully transmitted and failed(7) if an error was detected on transmission. The definition of "transmission error" in this context is left to the discretion of the -- implementor.

END

5. Acknowledgements

This document was produced by the PPP working group. In addition to the working group, the author wishes to thank the following individuals for their comments and contributions:

Bill Simpson -- Daydreamer
Glenn McGregor -- Merit
Jesse Walker -- DEC
Chris Gunner -- DEC

6. Security Considerations

The PPP MIB affords the network operator the ability to configure and control the PPP links of a particular system. This represents a security risk.

These risks are addressed in the following manners:

1. All variables which represent a significant security risk are placed in separate, optional, MIB Groups. As the MIB Group is the quantum of implementation within a MIB, the implementor of the MIB may elect not to implement these groups.

2. The implementor may choose to implement the variables which present a security risk so that they may not be written, i.e., the variables are READ-ONLY. This method still presents a security risk, and is not recommended, in that the variables, specifically the PPP Authentication Protocols’ variables, may be easily read.
(3) Using SNMPv2, the operator can place the variables into MIB views which are protected in that the parties which have access to those MIB views use authentication and privacy protocols, or the operator may elect to make these views not accessible to any party. In order to facilitate this placement, all security-related variables are placed in separate MIB Tables. This eases the identification of the necessary MIB View Subtree.

7. References


8. Author’s Address

Frank Kastenholz
FTP Software, Inc.
2 High Street
North Andover, Mass 01845 USA

Phone: (508) 685-4000
EMail: kasten@ftp.com