        The Point-to-Point Protocol (PPP) Initial Configuration Options

Status of this Memo

   This RFC specifies an IAB standards track protocol for the Internet
   community.

   Please refer to the current edition of the "IAB Official Protocol
   Standards" for the standardization state and status of this protocol.

   This proposal is the product of the Point-to-Point Protocol Working
   Group of the Internet Engineering Task Force (IETF). Comments on
   this memo should be submitted to the IETF Point-to-Point Protocol
   Working Group chair.

   Distribution of this memo is unlimited.

Abstract

   The Point-to-Point Protocol (PPP) provides a method for transmitting
   datagrams over serial point-to-point links. PPP is composed of

       1) a method for encapsulating datagrams over serial links,
       2) an extensible Link Control Protocol (LCP), and
       3) a family of Network Control Protocols (NCP) for establishing
       and configuring different network-layer protocols.

   The PPP encapsulating scheme, the basic LCP, and an NCP for
   controlling and establishing the Internet Protocol (IP) (called the
   IP Control Protocol, IPCP) are defined in The Point-to-Point Protocol
   (PPP) [1].

   This document defines the intial options used by the LCP and IPCP. It
   also defines a method of Link Quality Monitoring and a simple
   authentication scheme.

Table of Contents

1. Introduction

   The Point-to-Point Protocol (PPP) [1] proposes a standard method of
   encapsulating IP datagrams, and other Network Layer protocol
   information, over point-to-point links. PPP also proposes an
   extensible Option Negotiation Protocol. [1] specifies only the
   protocol itself; the initial set of Configuration Options are
   described in this document. These Configuration Options allow MTUs
   to be changed, IP addresses to be dynamically assigned, header
   compression to be enabled, and much more.

   This memo is divided into several sections. Section 2 describes
   Configuration Options for the Link Control Protocol. Section 3
   specifies the use of the Link Quality Monitoring option. Section 4
   defines a simple Password Authentication Protocol. Finally, Section 5
   specifies Configuration Options for the IP Control Protocol.

2. Link Control Protocol (LCP) Configuration Options

   As described in [1], LCP Configuration Options allow modifications to
   the standard characteristics of a point-to-point link to be
   negotiated. Negotiable modifications proposed in this document
   include such things as the maximum receive unit, async control
   character mapping, the link authentication method, etc.

   The initial proposed values for the LCP Configuration Option Type
   field (see [1]) are assigned as follows:

        1   Maximum-Receive-Unit
        2   Async-Control-Character-Map
        3   Authentication-Type
        4   NOT ASSIGNED
        5   Magic-Number
        6   Link-Quality-Monitoring
        7   Protocol-Field-Compression
        8   Address-and-Control-Field-Compression

2.1. Maximum-Receive-Unit

   Description

      This Configuration Option provides a way to negotiate the maximum
      packet size used across one direction of a link. By default, all
      implementations must be able to receive frames with 1500 octets of
      Information.

      This Configuration Option may be sent to inform the remote end
      that you can receive larger frames, or to request that the remote
      end send you smaller frames. If smaller frames are requested, an
      implementation MUST still be able to receive 1500 octet frames in
      case link synchronization is lost.

   A summary of the Maximum-Receive-Unit Configuration Option format is
   shown below. The fields are transmitted from left to right.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |        Maximum-Receive-Unit   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   Type

      1

   Length

      4

   Maximum-Receive-Unit

      The Maximum-Receive-Unit field is two octets and indicates the new
      maximum receive unit. The Maximum-Receive-Unit covers only the
      Data Link Layer Information field but not the header, trailer or
      any transparency bits or bytes.

   Default

      1500

2.2. Async-Control-Character-Map

   Description

      This Configuration Option provides a way to negotiate the use of
      control character mapping on asynchronous links. By default, PPP
      maps all control characters into an appropriate two character
      sequence. However, it is rarely necessary to map all control
      characters and often times it is unnecessary to map any
      characters. A PPP implementation may use this Configuration
      Option to inform the remote end which control characters must
      remain mapped and which control characters need not remain mapped
      when the remote end sends them. The remote end may still send
      these control characters in mapped format if it is necessary
      because of constraints at its (the remote) end. This option does
      not solve problems for communications links that can send only 7-
      bit characters or that can not send all non-control characters.

      There may be some use of synchronous-to-asynchronous converters
      (some built into modems) in Point-to-point links resulting in a
      synchronous PPP implementation on one end of a link and an
      asynchronous implemention on the other. It is the responsibility
      of the converter to do all mapping conversions during operation.
      To enable this functionality, synchronous PPP implementations MUST
      always accept a Async-Control-Character-Map Configuration Option
      (it MUST always respond to an LCP Configure-Request specifying
      this Configuration Option with an LCP Configure-Ack). However,
      acceptance of this Configuration Option does not imply that the
      synchronous implementation will do any character mapping, since
      synchronous PPP uses bit-stuffing rather than character-stuffing.
      Instead, all such character mapping will be performed by the
      asynchronous-to-synchronous converter.

   A summary of the Async-Control-Character-Map Configuration Option
   format is shown below. The fields are transmitted from left to
   right.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Type      |    Length     |    Async-Control-Character-Map
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
            (cont)                 |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
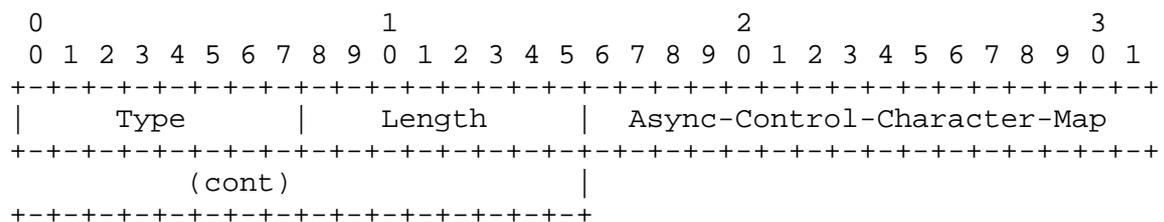
   Type

      2

Length

   6

Async-Control-Character-Map

   The Async-Control-Character-Map field is four octets and indicates
   the new async control character map. The map is encoded in big-
   endian fashion where each numbered bit corresponds to the ASCII
   control character of the same value. If the bit is cleared to
   zero, then that ASCII control character need not be mapped. If
   the bit is set to one, then that ASCII control character must
   remain mapped. E.g., if bit 19 is set to zero, then the ASCII
   control character 19 (DC3, Control-S) may be sent in the clear.

Default

   All ones (0xffffffff).

2.3. Authentication-Type

   Description

      On some links it may be desirable to require a peer to
      authenticate itself before allowing Network Layer protocol data to
      be exchanged. This Configuration Option provides a way to
      negotiate the use of a specific authentication protocol. By
      default, authentication is not necessary. If an implementation
      requires that the remote end authenticate with some specific
      authentication protocol, then it should negotiate the use of that
      authentication protocol with this Configuration Option.

      Successful negotiation of the Authentication-Type option adds an
      additional Authentication phase to the Link Control Protocol.
      This phase is after the Link Quality Determination phase, and
      before the Network Layer Protocol Configuration Negotiation phase.
      Advancement from the Authentication phase to the Network Layer
      Protocol Configuration Negotiation phase may not occur until the
      peer is successfully authenticated using the negotiated
      authentication protocol.

      An implementation may allow the remote end to pick from more than
      one authentication protocol. To achieve this, it may include
      multiple Authentication-Type Configuration Options in its
      Configure-Request packets. An implementation receiving a
      Configure-Request specifying multiple Authentication-Types may
      accept at most one of the negotiable authentication protocols and
      should send a Configure-Reject specifying all of the other
      specified authentication protocols.

      It is recommended that each PPP implementation support
      configuration of authentication parameters at least on a per-
      interface basis, if not a per peer entity basis. The parameters
      should specify which authetication techniques are minimally
      required as a prerequisite to establishment of a PPP connection,
      either for the specified interface or for the specified peer
      entity. Such configuration facilities are necessary to prevent an
      attacker from negotiating a reduced security authentication
      protocol, or no authentication at all, in an attempt to circumvent
      this authentication facility.

      If an implementation sends a Configure-Ack with this Configuration
      Option, then it is agreeing to authenticate with the specified
      protocol. An implementation receiving a Configure-Ack with this
      Configuration Option should expect the remote end to authenticate
      with the acknowledged protocol.

There is no requirement that authentication be full duplex or that
the same authentication protocol be used in both directions. It
is perfectly acceptable for different authentication protocols to
be used in each direction. This will, of course, depend on the
specific authentication protocols negotiated.

This document defines a simple Password Authentication Protocol in
Section 4. Development of other more secure protocols is
encouraged.

A summary of the Authentication-Type Configuration Option format is
shown below. The fields are transmitted from left to right.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |       Authentication-Type     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Data ...
+-+-+-+-+-+
```
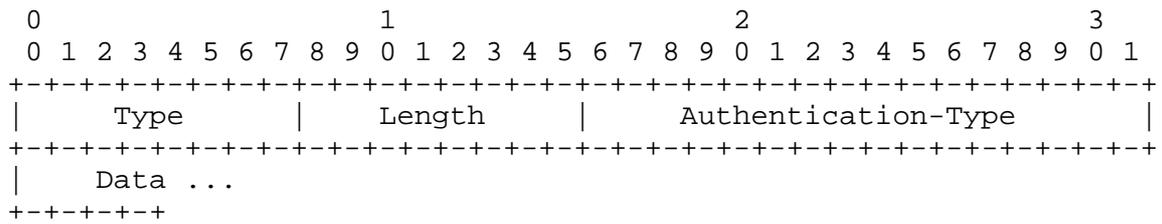
Type

   3

Length

   >= 4

Authentication-Type

   The Authentication-Type field is two octets and indicates the type
   of authentication protocol desired. Values for the
   Authentication-Type are always the same as the PPP Data Link Layer
   Protocol field values for that same authentication protocol. The
   most up-to-date values of the Authentication-Type field are
   specified in "Assigned Numbers" [2]. Initial values are assigned
   as follows:

      Value (in hex)   Protocol

      c023   Password Authentication Protocol

Data

   The Data field is zero or more octets and contains additional data
   as determined by the particular authentication protocol.

Default

    No authentication protocol necessary.

2.4. Magic-Number

    Description

        This Configuration Option provides a way to detect looped-back
        links and other Data Link Layer anomalies. This Configuration
        Option may be required by some other Configuration Options such as
        the Link-Quality-Monitoring Configuration Option.

        Before this Configuration Option is requested, an implementation
        must choose its Magic-Number. It is recommended that the Magic-
        Number be chosen in the most random manner possible in order to
        guarantee with very high probability that an implementation will
        arrive at a unique number. A good way to choose a unique random
        number is to start with an unique seed. Suggested sources of
        uniqueness include machine serial numbers, other network hardware
        addresses, time-of-day clocks, etc. Particularly good random
        number seeds are precise measurements of the inter-arrival time of
        physical events such as packet reception on other connected
        networks, server response time, or the typing rate of a human
        user. It is also suggested that as many sources as possible be
        used simultaneously.

        When a Configure-Request is received with a Magic-Number
        Configuration Option, the received Magic-Number should be compared
        with the Magic-Number of the last Configure-Request sent to the
        peer. If the two Magic-Numbers are different, then the link is
        not looped-back, and the Magic-Number should be acknowledged. If
        the two Magic-Numbers are equal, then it is possible, but not
        certain, that the link is looped-back and that this Configure-
        Request is actually the one last sent. To determine this, a
        Configure-Nak should be sent specifying a different Magic-Number
        value. A new Configure-Request should not be sent to the peer
        until normal processing would cause it to be sent (i.e., until a
        Configure-Nak is received or the Restart timer runs out).

        Reception of a Configure-Nak with a Magic-Number different from
        that of the last Configure-Nak sent to the peer proves that a link
        is not looped-back, and indicates a unique Magic-Number. If the
        Magic-Number is equal to the one sent in the last Configure-Nak,
        the possibility of a loop-back is increased, and a new Magic-
        Number should be chosen. In either case, a new Configure-Request
        should be sent with the new Magic-Number.

If the link is indeed looped-back, this sequence (transmit
Configure-Request, receive Configure-Request, transmit Configure-
Nak, receive Configure-Nak) will repeat over and over again. If
the link is not looped-back, this sequence may occur a few times,
but it is extremely unlikely to occur repeatedly. More likely,
the Magic-Numbers chosen at either end will quickly diverge,
terminating the sequence. The following table shows the
probability of collisions assuming that both ends of the link
select Magic-Numbers with a perfectly uniform distribution:

```
    Number of Collisions         Probability
    --------------------     --------------------
             1               1/2**32    = 2.3 E-10
             2               1/2**32**2 = 5.4 E-20
             3               1/2**32**3 = 1.3 E-29
```

Good sources of uniqueness or randomness are required for this
divergence to occur. If a good source of uniqueness cannot be
found, it is recommended that this Configuration Option not be
enabled; Configure-Requests with the option should not be
transmitted and any Magic-Number Configuration Options which the
peer sends should be either acknowledged or rejected. In this
case, loop-backs cannot be reliably detected by the
implementation, although they may still be detectable by the peer.

If an implementation does transmit a Configure-Request with a
Magic-Number Configuration Option, then it MUST NOT respond with a
Configure-Reject if its peer also transmits a Configure-Request
with a Magic-Number Configuration Option. That is, if an
implementation desires to use Magic Numbers, then it MUST also
allow its peer to do so. If an implementation does receive a
Configure-Reject in response to a Configure-Request, it can only
mean that the link is not looped-back, and that its peer will not
be using Magic-Numbers. In this case, an implementation may act
as if the negotiation had been successful (as if it had instead
received a Configure-Ack).

The Magic-Number also may be used to detect looped-back links
during normal operation as well as during Configuration Option
negotiation. All Echo-Request, Echo-Reply, Discard-Request, and
Link-Quality-Report LCP packets have a Magic-Number field which
MUST normally be transmitted as zero, and MUST normally be ignored
on reception. However, once a Magic-Number has been successfully
negotiated, an LCP implementation MUST begin transmitting these
packets with the Magic-Number field set to its negotiated Magic-
Number. Additionally, the Magic-Number field of these packets may
be inspected on reception. All received Magic-Number fields should
be equal to either zero or the peer's unique Magic-Number,

depending on whether or not the peer negotiated one. Reception of
a Magic-Number field equal to the negotiated local Magic-Number
indicates a looped-back link. Reception of a Magic-Number other
than the negotiated local Magic-Number or or the peer's negotiated
Magic-Number, or zero if the peer didn't negotiate one, indicates
a link which has been (mis)configured for communications with a
different peer.

Procedures for recovery from either case are unspecified and may
vary from implementation to implementation. A somewhat
pessimistic procedure is to assume an LCP Physical-Layer-Down
event and make an immediate transition to the Closed state. A
further Active-Open event will begin the process of re-
establishing the link, which can't complete until the loop-back
condition is terminated and Magic-Numbers are successfully
negotiated. A more optimistic procedure (in the case of a loop-
back) is to begin transmitting LCP Echo-Request packets until an
appropriate Echo-Reply is received, indicating a termination of
the loop-back condition.

A summary of the Magic-Number Configuration Option format is shown
below. The fields are transmitted from left to right.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |          Magic-Number
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    Magic-Number (cont)         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
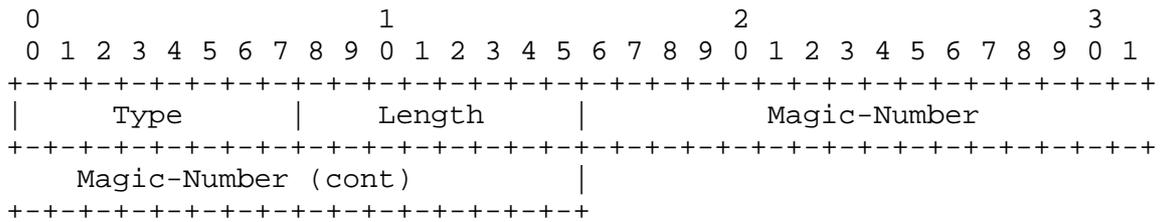
Type

    5

Length

    6

Magic-Number

    The Magic-Number field is four octets and indicates a number which
    is very likely to be unique to one end of the link. A Magic-
    Number of zero is illegal and must not be sent.

Default

    None.

2.5. Link-Quality-Monitoring

   Description

      On some links it may be desirable to determine when, and how
      often, the link is dropping data. This process is called Link
      Quality Monitoring and is implemented by periodically transmitting
      Link-Quality-Report packets as described in Section 3. The Link-
      Quality-Monitoring Configuration Option provides a way to enable
      the use of Link-Quality-Report packets, and also to negotiate the
      rate at which they are transmitted. By default, Link Quality
      Monitoring and the use of Link-Quality-Report packets is disabled.

   A summary of the Link-Quality-Monitoring Configuration Option format
   is shown below. The fields are transmitted from left to right.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Type      |    Length     |         Reporting-Period
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       Reporting-Period (cont)     |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
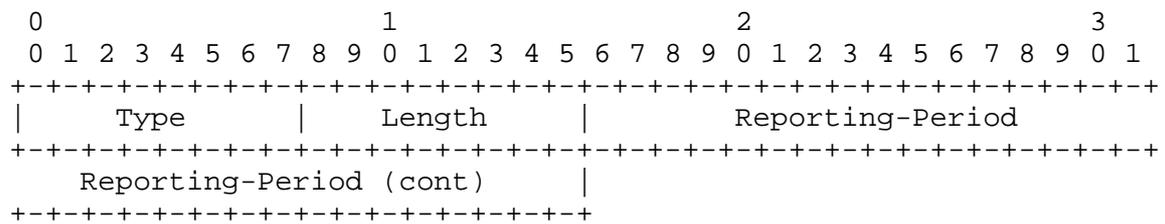
   Type

      6

   Length

      6

   Reporting-Period

      The Reporting-Period field is four octets and indicates the
      maximum time in micro-seconds that the remote end should wait
      between transmission of LCP Link-Quality-Report packets. A value
      of zero is illegal and should always be nak'd or rejected. An LCP
      implementation is always free to transmit LCP Link-Quality-Report
      packets at a faster rate than that which was requested by, and
      acknowledged to, the remote end.

   Default

      None

2.6. Protocol-Field-Compression

   Description

      This Configuration Option provides a way to negotiate the
      compression of the Data Link Layer Protocol field. By default,
      all implementations must transmit standard PPP frames with two
      octet Protocol fields. However, PPP Protocol field numbers are
      chosen such that some values may be compressed into a single octet
      form which is clearly distinguishable from the two octet form.
      This Configuration Option may be sent to inform the remote end
      that you can receive compressed single octet Protocol fields.
      Compressed Protocol fields may not be transmitted unless this
      Configuration Option has been received.

      As previously mentioned, the Protocol field uses an extension
      mechanism consistent with the ISO 3309 extension mechanism for the
      Address field; the Least Significant Bit (LSB) of each octet is
      used to indicate extension of the Protocol field. A binary "0" as
      the LSB indicates that the Protocol field continues with the
      following octet. The presence of a binary "1" as the LSB marks
      the last octet of the Protocol field. Notice that any number of
      "0" octets may be prepended to the field, and will still indicate
      the same value (consider the two representations for 3, 00000011
      and 00000000 00000011).

      In the interest of simplicity, the standard PPP frame uses this
      fact and always sends Protocol fields with a two octet
      representation. Protocol field values less than 256 (decimal) are
      prepended with a single zero octet even though transmission of
      this, the zero and most significant octet, is unnecessary.

      However, when using low speed links, it is desirable to conserve
      bandwidth by sending as little redundant data as possible. The
      Protocol Compression Configuration Option allows a trade-off
      between implementation simplicity and bandwidth efficiency. If
      successfully negotiated, the ISO 3309 extension mechanism may be
      used to compress the Protocol field to one octet instead of two.
      The large majority of frames are compressible since data protocols
      are typically assigned with Protocol field values less than 256.

      To guarantee unambiguous recognition of LCP packets, the Protocol
      field must never be compressed when sending any LCP packet. In
      addition, PPP implementations must continue to be robust and MUST
      accept PPP frames with double-octet, as well as single-octet,
      Protocol fields, and MUST NOT distinguish between them.

      When a Protocol field is compressed, the Data Link Layer FCS field

is calculated on the compressed frame, not the original
uncompressed frame.

A summary of the Protocol-Field-Compression Configuration Option
format is shown below. The fields are transmitted from left to
right.

```
 0                   1
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
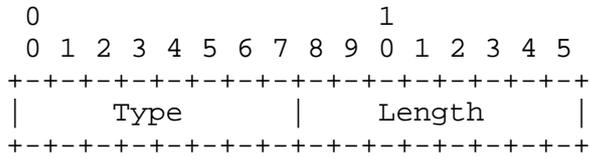
Type

   7

Length

   2

Default

   Disabled.

2.7. Address-and-Control-Field-Compression

    Description

        This Configuration Option provides a way to negotiate the
        compression of the Data Link Layer Address and Control fields. By
        default all implementations must transmit frames with Address and
        Control fields and must use the hexadecimal values 0xff and 0x03
        respectively. Since these fields have constant values, they are
        easily compressed. this Configuration Option may be used to
        inform the remote end that you can receive compressed Address and
        Control fields.

        Compressed Address and Control fields are formed by simply
        omitting them in all non-ambiguous cases. Ambiguous frames may
        not be compressed. Ambiguous cases result when the two octets
        following the Address and Control fields have values that could be
        interpreted as valid Address and Control fields (i.e., 0xff,
        0x03). This can happen when Protocol-Field-Compression is enabled
        and the Protocol field is compressed to one octet. If the
        Protocol value is 0xff, and the first octet of the Information
        field is 0x03, the result is ambiguous and the Address and Control
        fields must not be compressed on transmission.

        On reception, the Address and Control fields are decompressed by
        examining the first two octets. If they contain the values 0xff
        and 0x03, they are assumed to be the Address and Control fields.
        If not, it is assumed that the fields were compressed and were not
        transmitted.

        One additional case in which the Address and Control fields must
        never be compressed is when sending any LCP packet. This rule
        guarantees unambiguous recognition of LCP packets.

        When the Address and Control fields are compressed, the Data Link
        Layer FCS field is calculated on the compressed frame, not the
        original uncompressed frame.

     A summary of the Address-and-Control-Field-Compression configuration
     option format is shown below. The fields are transmitted from left
     to right.

         0   1
         0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
        +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
        | Type | Length |
        +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
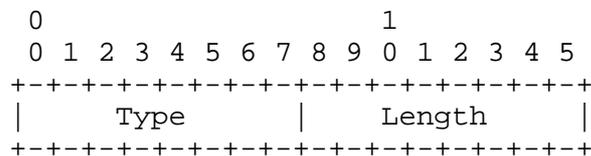
   Type

      8

   Length

      2

   Default

      Not compressed.

3. Link Quality Monitoring

   Data communications links are rarely perfect. Packets can be dropped
   or corrupted for various reasons (line noise, equipment failure,
   buffer overruns, etc.). Sometimes, it is desirable to determine
   when, and how often, the link is dropping data. Routers, for
   example, may want to temporarily allow another route to take
   precedence. An implementation may also have the option of
   disconnecting and switching to an alternate link. The process of
   determining data loss is called "Link Quality Monitoring".

3.1. Design Motivation

   There are many different ways to measure link quality, and even more
   ways to react to it. Rather than specifying a single scheme, Link
   Quality Monitoring is divided into a "mechanism" and a "policy". PPP
   fully specifies the "mechanism" for Link Quality Monitoring by
   defining the LCP Link-Quality-Report (LQR) packet and specifying a
   procedure for its use. PPP does NOT specify a Link Quality
   Monitoring "policy" -- how to judge link quality or what to do when
   it is inadequate. That is left as an implementation decision, and
   can be different at each end of the link. Implementations are
   allowed, and even encouraged, to experiment with various link quality
   policies. The Link Quality Monitoring mechanism specification
   insures that two implementations with different policies may
   communicate and interoperate.

   To allow flexible policies to be implemented, the PPP Link Quality
   Monitoring mechanism measures data loss in units of packets, octets,
   and Link-Quality-Reports. Each measurement is made separately for
   each half of the link, both inbound and outbound. All measurements
   are communicated to both ends of the link so that each end of the
   link can implement its own link quality policy for both its outbound
   and inbound links.

   Finally, the Link Quality Monitoring protocol is designed to be
   implementable on many different kinds of systems. Although it may be
   common to implement PPP (and especially Link Quality Monitoring) as a
   single software process, multi-process implementations with hardware
   support are also envisioned. The PPP Link Quality Monitoring
   mechanism provides for this by careful definition of the Link-
   Quality-Report packet format, and by specifiying reference points for
   all data transmission and reception measurements.

3.2. Design Overview

   Each Link Quality Monitoring implementation maintains counts of the
   number of packets and octets transmitted and successfully received,

and periodically transmits this information to its peer in a Link-
Quality-Report packet. These packets contain three sections: a
Header section, a Counters section, and a Measurements section.

The Header section of the packet consists of the normal LCP Link
Maintenance packet header including Code, Identifier, Length and
Magic-Number fields.

The Counters section of the packet consists of four counters, and
provides the information necessary to measure the quality of the
link. The LQR transmitter fills in two of these counters: Out-Tx-
Packets-Ctr and Out-Tx-Octets-Ctr (described later). The LQR
receiver fills in the two remaining counters: In-Rx-Packets-Ctr and
In-Rx-Octets-Ctr (described later). These counters are similar to
sequence numbers; they are constantly increasing to give a "relative"
indication of the number of packets and octets communicated across
the outbound link. By comparing the values in successive Link-
Quality-Reports, an LQR receiver can compute the "absolute" number of
packets and octets communicated across its inbound link. Comparing
these absolute numbers then gives an indication of an inbound link's
quality. Relative numbers, rather than absolute, are transmitted
because they greatly simplify link synchronization; an implementation
merely waits to receive two LQR packets.

The Measurements section of the packet consists of six state
variables: In-Tx-LQRs, Last-In-Id, In-Tx-Packets, In-Tx-Octets, In-
Rx-Packets, and In-Rx-Octets (described later). This section allows
an implementation to report inbound link quality measurements to its
peer (for which the report will instead indicate outbound link
quality) by transmitting the absolute, rather than relative, number
of LQRs, packets, and octets communicated across the inbound link.
These values are calculated by observing the Counters section of the
Link-Quality-Report packets received on the inbound link. Absolute
numbers may be used in this section without synchronization problems
because it is necessary to receive only one LQR packet to have valid
information.

Link Quality Monitoring is described in more detail in the following
sections. First, a description of the processes comprising the Link
Quality Monitoring mechanism is presented. This is followed by the
packet and byte counters maintained; the measurements, calculations,
and state variables used; the format of the Link-Quality-Report
packet; some policy suggestions; and, finally, an example link
quality calculation.

3.3. Processes

The PPP Link Quality Monitoring mechanism is described using a

"logical process" model. As shown below, there are five logical
processes duplicated at each end of the duplex link.

```
+---------+   +-------+   +----+ Outbound
|         |-->|  Mux  |-->| Tx |=========>
| Link-   |   +-------+   +----+
| Manager |
|         |   +-------+   +----+ Inbound
|         |<--| Demux |<--| Rx |<=========
+---------+   +-------+   +----+
```
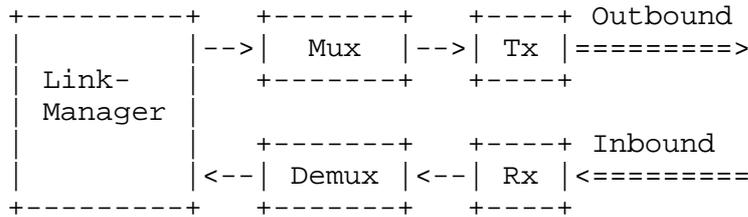
Link-Manager

   The Link-Manager process transmits and receives Link-Quality-
   Reports, and implements the desired link quality policy. LQR
   packets are transmitted at a constant rate, which is negotiated by
   the LCP Link-Quality-Monitoring Configuration Option. The Link-
   Manager process fills in only the Header and Measurements sections
   of the packet; the Counters section of the packet is filled in by
   the Tx and Rx processes.

Mux

   The Mux process multiplexes packets from the various protocols
   (e.g., LCP, IP, XNS, etc.) into a single, sequential, and
   prioritized stream of packets. Link-Quality-Report packets MUST
   be given the highest possible priority to insure that link quality
   information is communicated in a timely manner.

Tx

   The Tx process maintains the counters Out-Tx-Packets-Ctr and Out-
   Tx-Octets-Ctr which are used to measure the amount of data which
   is transmitted on the outbound link. When Tx processes a Link-
   Quality-Report packet, it inserts the values of these counters
   into the Counters section of the packet. Because these counters
   represent relative, rather than absolute, values, the question of
   when to update the counters, before or after they are inserted
   into a Link-Quality-Report packet, is left as an implementation
   decision. However, an implementation MUST make this decision the
   same way every time. The Tx process MUST follow the Mux process
   so that packets are counted in the order transmitted to the link.

Rx

   The Rx process maintains the counters In-Rx-Packets-Ctr and In-
   Rx-Octets-Ctr which are used to measure the amount of data which
   is received by the inbound link. When Rx processes a Link-

Quality-Report packet, it inserts the values of these counters
into the Counters section of the packet. Again, the question of
when to update the counters, before or after they are inserted
into a Link-Quality-Report packet, is left as an implementation
decision which MUST be made consistently the same way.

   Demux

      The Demux process demultiplexes packets for the various protocols.
      The Demux process MUST follow the Rx process so that packets are
      counted in the order received from the link.

3.4. Counters

   In order to fill in the Counters section of a Link-Quality-Report
   packet, Link Quality Monitoring requires the implementation of one
   8-bit unsigned, and four 32-bit unsigned, monotonically increasing
   counters. These counters may be reset to any initial value before
   the first Link-Quality-Report is transmitted, but MUST NOT be reset
   again until LCP has left the Open state. Counters wrap to zero when
   their maximum value is reached (for 32 bit counters: 0xffffffff + 1 =
   0).

   Out-Identifier-Ctr

      Out-Identifier-Ctr is an 8-bit counter maintained by the Link-
      Manager process which increases by one for each transmitted Link-
      Quality-Report packet.

   Out-Tx-Packets-Ctr

      Out-Tx-Packets-Ctr is a 32-bit counter maintained by the Tx
      process which increases by one for each transmitted Data Link
      Layer packet.

   Out-Tx-Octets-Ctr

      Out-Tx-Octets-Ctr is a 32-bit counter maintained by the Tx process
      which increases by one for each octet in a transmitted Data Link
      Layer packet. All octets which are included in the FCS
      calculation MUST be counted, as should the FCS octets themselves.
      All other octets MUST NOT be counted.

   In-Rx-Packets-Ctr

      In-Rx-Packets-Ctr is a 32-bit counter maintained by the Rx process
      which increases by one for each successfully received Data Link
      Layer packet. Packets with incorrect FCS fields or other problems

MUST not be counted.

    In-Rx-Octets-Ctr

        In-Rx-Octets-Ctr is a 32-bit counter maintained by the Rx process
        which increases by one for each octet in a successfully received
        Data Link Layer packet. All octets which are included in an FCS
        calculation MUST be counted, as should the FCS octets themselves.
        All other octets MUST NOT be counted.

3.5. Measurements, Calculations, State Variables

    In order to fill in the Measurements section of a Link-Quality-Report
    packet, Link Quality Monitoring requires the Link-Manager process to
    make a number of calculations and keep a number of state variables.
    These calculations are made, and these state variables updated, each
    time a Link-Quality-Report packet is received from the inbound link.

    In-Tx-LQRs

        In-Tx-LQRs is an 8-bit state variable which indicates the number
        of Link-Quality-Report packets which the peer had to transmit in
        order for the local end to receive exactly one LQR. In-Tx-LQRs
        defines the length of the "period" over which In-Tx-Packets, In-
        Tx-Octets, In-Rx-Packets, and In-Rx-Octets were measured. In-Tx-
        LQRs is calculated by subtracting Last-In-Id from the received
        Identifier. If more than 255 LQRs in a row are lost, In-Tx-LQRs
        will be ambiguous since the Identifier field and all state
        variables based on it are only 8 bits. It is assumed that the
        Link Quality Monitoring policy will be robust enough to handle
        this case (it should probably close down the link long before this
        happens).

    Last-In-Id

        Last-In-Id is an 8-bit state variable which stores the value of
        the last received Identifier. Last-In-Id should be updated after
        In-Tx-LQRs has been calculated.

    In-Tx-Packets

        In-Tx-Packets is a 32-bit state variable which indicates the
        number of packets which were transmitted on the inbound link
        during the last period. In-Tx-Packets is calculated by
        subtracting Last-Out-Tx-Packets-Ctr from the received Out-Tx-
        Packets-Ctr.

   Last-Out-Tx-Packets-Ctr

      Last-Out-Tx-Packets-Ctr is a 32-bit state variable which stores
      the value of the last received Out-Tx-Packets-Ctr. Last-Out-Tx-
      Packets-Ctr should be updated after In-Tx-Packets has been
      calculated.

   In-Tx-Octets

      In-Tx-Octets is a 32-bit state variable which indicates the number
      of octets which were transmitted on the inbound link during the
      last period. In-Tx-Octets is calculated by subtracting Last-Out-
      Tx-Octets-Ctr from the received Out-Tx-Octets-Ctr.

   Last-Out-Tx-Octets-Ctr

      Last-Out-Tx-Octets-Ctr is a 32-bit state variable which stores the
      value of the last received Out-Tx-Octets-Ctr. Last-Out-Tx-
      Octets-Ctr should be updated after In-Tx-Octets has been
      calculated.

   In-Rx-Packets

      In-Rx-Packets is a 32-bit state variable which indicates the
      number of packets which were received on the inbound link during
      the last period. In-Rx-Packets is calculated by subtracting
      Last-In-Rx-Packets-Ctr from the received In-Rx-Packets-Ctr.

   Last-In-Rx-Packets-Ctr

      Last-In-Rx-Packets-Ctr is a 32-bit state variable which stores the
      value of the last received In-Rx-Packets-Ctr. Last-In-Rx-
      Packets-Ctr should be updated after In-Rx-Packets has been
      calculated.

   In-Rx-Octets

      In-Rx-Octets is a 32-bit state variable which indicates the number
      of octets which were received on the inbound link during the last
      period. In-Rx-Octets is calculated by subtracting Last-In-Rx-
      Octets-Ctr from the received In-Rx-Octets-Ctr.

   Last-In-Rx-Octets-Ctr

      Last-In-Rx-Octets-Ctr is a 32-bit state variable which stores the
      value of the last received In-Rx-Octets-Ctr. Last-In-Rx-Octets-
      Ctr should be updated after In-Rx-Octets has been calculated.

Measurements-Valid

Measurements-Valid is a 1-bit boolean state variable which
indicates whether or not the In-Tx-Packets, In-Tx-Octets, In-Rx-
Packets, and In-Rx-Octets state variables contain valid
measurements. These measurements cannot be considered valid until
two or more Link-Quality-Report packets have been received on the
inbound link. This bit should be reset when LCP reaches the Open
state and should be set after the receipt of exactly two LQRs.

3.6. Link-Quality-Report Packet Format

A Summary of the Link-Quality-Report packet format is shown below.
The fields are transmitted from left to right. The Code, Identifier,
Length, and Magic-Number fields make up the normal LCP Link
Maintenance packet header; the In-Tx-LQRS, Last-In-Id, V, In-Tx-
Packets, In-Tx-Octets, In-Rx-Packets, In-Rx-Octets fields contain
digested absolute measurements; and the Out-Tx-Packets-Ctr, Out-Tx-
Octets-Ctr, In-Rx-Packets-Ctr, and In-Rx-Octets-Ctr fields contain
raw relative counts. Note that as transmitted over the link, this
packet format does not include the In-Rx-Packets-Ctr and In-Rx-
Octets-Ctr fields which are logically appended to the packet by the
Rx process after reception on the inbound link.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Code      |  Identifier   |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Magic-Number                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   In-Tx-LQRs   |   Last-In-Id  |          Reserved          |V|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        In-Tx-Packets                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         In-Tx-Octets                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        In-Rx-Packets                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         In-Rx-Octets                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Out-Tx-Packets-Ctr                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Out-Tx-Octets-Ctr                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/                                                               /
/                                                               /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       In-Rx-Packets-Ctr                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        In-Rx-Octets-Ctr                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
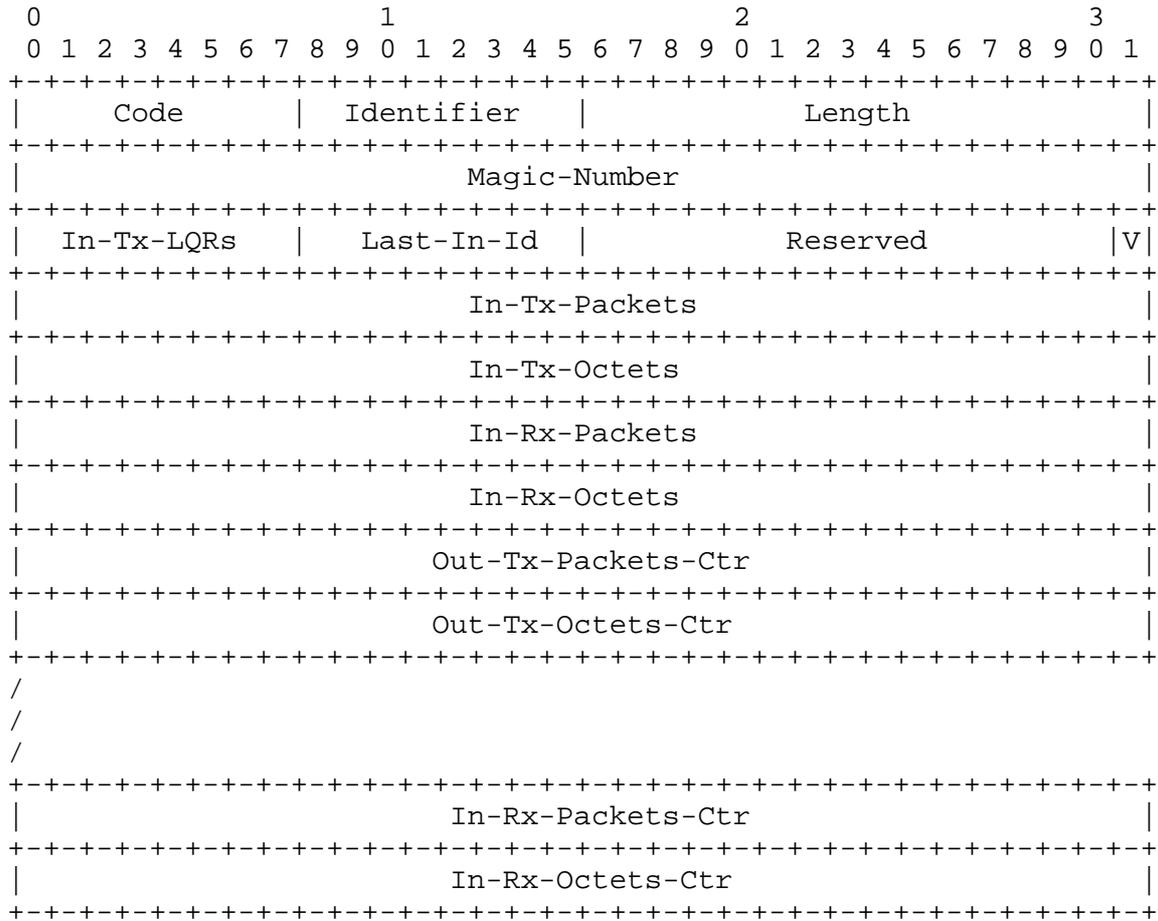
Code

   12 for Link-Quality-Report.

Identifier

   The Identifier field is one octet and indicates the sequence
   number for this Link-Quality-Report. The Identifier field is
   copied from the Out-Identifier-Ctr counter on transmission. On
   reception, the Identifier field is used to calculate In-Tx-LQRs
   and is then stored in Last-In-Id.

   The Link-Quality-Report Identifier sequence number space MUST be
   separate from that of all other LCP packets; for example,
   transmission of an LCP Echo-Request must not cause the Out-
   Identifier-Ctr counter to be incremented.

Length

   The Length field is two octets and indicates the length of the LQM
   packet including the Code, Identifier, Length and all defined
   fields. Octets outside the range of the length field should be
   treated as Data Link Layer padding and should be ignored on
   reception. In order for the correct In-Tx-Octets and In-Rx-Octets
   values to be calculated, Link-Quality-Reports MUST be consistently
   transmitted with the same amount of padding.

Magic-Number

   The Magic-Number field is four octets and aids in detecting
   looped-back links. Unless modified by a Configuration Option, the
   Magic-Number MUST always be transmitted as zero and MUST always be
   ignored on reception. If Magic-Numbers have been negotiated,
   incoming LQM packets should be checked to make sure that the local
   end is not seeing its own Magic-Number and thus a looped-back
   link.

In-Tx-LQRs

   The In-Tx-LQRs field is one octet and indicates the number of
   periods covered by the Measurements section of this Link-Quality-
   Report. The In-Tx-LQRs field is copied from the In-Tx-LQRs state
   variable on transmission.

Last-In-Id

   The Prev-In-Id field is one octet and indicates the age of the
   Measurements section of this Link-Quality-Report. The Last-In-Id
   field is copied from the Last-In-Id field on transmission. On
   reception, the Last-In-Id field may be compared with the Out-
   Identifier-Ctr to determine how many, if any, outbound Link-
   Quality-Reports have been lost.

V

   The V field is 1 bit and indicates whether or not the Measurements
   section of this Link-Quality-Report is valid. The V field is
   copied from the Measurements-Valid state variable on transmission.
   If the V field is not set to 1, then the In-Tx-LQRs, Last-In-Id,
   In-Tx-Packets, In-Tx-Octets, In-Rx-Packets and In-Rx-Octets fields
   should be ignored.

Reserved

   The Reserved field is 15 bits and is intended to pad the remaining

packet fields to even four-octet boundaries for the convenience of
hardware implementations. The Reserved field should always be
transmitted as zero and ignored on reception.

   In-Tx-Packets

      The In-Tx-Packets field is four octets and indicates the number of
      packets transmitted on the inbound link of the Link-Quality-Report
      transmitter during the last measured period. The In-Tx-Packets
      field is copied from the In-Tx-Packets state variable on
      transmission.

   In-Tx-Octets

      The In-Tx-Octets field is four octets and indicates the number of
      octets transmitted on the inbound link of the Link-Quality-Report
      transmitter during the last measured period. The In-Tx-Octets
      field is copied from the In-Tx-Octets state variable on
      transmission.

   In-Rx-Packets

      The In-Rx-Packets field is four octets and indicates the number of
      packets received on the inbound link of the Link-Quality-Report
      transmitter during the last measured period. The In-Rx-Packets
      field is copied from the In-Rx-Packets state variable on
      transmission.

   In-Rx-Octets

      The In-Rx-Octets field is four octets and indicates the number of
      octets received on the inbound link of the Link-Quality-Report
      transmitter during the last measured period. The In-Rx-Octets
      field is copied from the In-Rx-Octets state variable on
      transmission.

   Out-Tx-Packets

      The Out-Tx-Packets field is four octets and is used to calculate
      the number of packets transmitted on the outbound link of the
      Link-Quality-Report transmitter during a period. The Out-Tx-
      Packets field is copied from the Out-Tx-Packets-Ctr counter on
      transmission.

   Out-Tx-Octets

      The Out-Tx-Octets field is four octets and is used to calculate
      the number of octets transmitted on the outbound link of the

Link-Quality-Report transmitter during a period. The Out-Tx-
Octets field is copied from the Out-Tx-Octets-Ctr counter on
transmission.

In-Rx-Packets

The In-Rx-Packets field is four octets and is used to calculate
the number of packets received on the inbound link of the Link-
Quality-Report receiver during a period. The In-Rx-Packets field
is copied from the In-Rx-Packets-Ctr counter on reception. The
In-Rx-Packets is not shown because it is not actually transmitted
over the link. Rather, it is logically appended (in an
implementation dependent manner) to the packet by the
implementation's Rx process.

In-Rx-Octets

The In-Rx-Octets field is four octets and is used to calculate the
number of octets received on the inbound link of the Link-
Quality-Report receiver during a period. The In-Rx-Octets field
is copied from the In-Rx-Octets-Ctr counter on reception. The
In-Rx-Octets is not shown because it is not actually transmitted
over the link. Rather, it is logically appended (in an
implementation dependent manner) to the packet by the
implementation's Rx process.

3.7. Policy Suggestions

Link-Quality-Report packets provide a mechanism to determine the link
quality, but it is up to each implementation to decide when the link
is usable. It is recommended that this policy implement some amount
of hysteresis so that the link does not bounce up and down. A
particularly good policy is to use a K out of N algorithm. In such
an algorithm, there must be K successes out of the last N periods for
the link to be considered of good quality.

Procedures for recovery from poor quality links are unspecified and
may vary from implementation to implementation. A suggested approach
is to immediately close all other Network-Layer protocols (i.e.,
cause IPCP to transmit a Terminate-Req), but to continue transmitting
Link-Quality-Reports. Once the link quality again reaches an
acceptable level, Network-Layer protocols can be reconfigured.

3.8. Example

An example may be helpful. Assume that Link-Manager implementation A
transmits a Link-Quality-Report which is received by Link-Manager
implementation B at time t0 with the following values:

```
   Out-Tx-Packets     5
   Out-Tx-Octets    100
   In-Rx-Packets      3
   In-Rx-Octets      70
```

   Assume that A then transmits 20 IP packets with 200 octets, of which
   15 packets and 150 octets are received by B. At time t1, A transmits
   another LQR which is received by B as follows:

```
   Out-Tx-Packets   26 (5 old, plus 20 IP, plus 1 LQR)
   Out-Tx-Octets   342 (42 for LQR)
   In-Rx-Packets    19
   In-Rx-Octets    262
```

   Implementation B can now calculate the number of packets and octets
   transmitted, received and lost on its inbound link as follows:

```
   In-Tx-Packets   =  26 -   5 =  21
   In-Tx-Octets    = 342 - 100 = 242
   In-Rx-Packets   =  10 -   3 =  16
   In-Rx-Octets    = 262 -  70 = 192
   In-Lost-Packets =  21 -  16 =   5
   In-Lost-Octets  = 242 - 192 =  50
```

   After doing these calculations, B evaluates the measurements in what
   ever way its implemented policy specifies. Also, the next time that
   B transmits an LQR to A, it will report these values in the
   Measurements section, thereby allowing A to evaluate these same
   measurements.

4. Password Authentication Protocol

   The Password Authentication Protocol (PAP) may be used to
   authenticate a peer by verifying the identity of the remote end of
   the link. Use of the PAP must first be negotiated using the LCP
   Authentication-Type Configuration Option. Successful negotiation
   adds an additional Authentication phase to the Link Control Protocol,
   after the Link Quality Determination phase, and before the Network
   Layer Protocol Configuration Negotiation phase. PAP packets received
   before the Authentication phase is reached should be silently
   discarded. The Authentication phase is exited once an Authenticate-
   Ack packet is sent or received.

   PAP is intended for use primarily by hosts and routers that connect
   via switched circuits or dial-up lines to a PPP network server. The
   server can then use the identification of the connecting host or
   router in the selection of options for network layer negotiations or
   failing authentication, drop the connection.

   Note that PAP is not a strong authentication method. Passwords are
   passed over the circuit in the clear and there is no protection from
   repeated trial and error attacks. Work is currently underway on more
   secure authentication methods for PPP and other protocols. It is
   strongly recommended to switch to these methods when they become
   available.

4.1. Packet Format

   Exactly one Password Authentication Protocol packet is encapsulated
   in the Information field of PPP Data Link Layer frames where the
   protocol field indicates type hex c023 (Password Authentication
   Protocol). A summary of the Password Authentication Protocol packet
   format is shown below. The fields are transmitted from left to
   right.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Code      |  Identifier   |            Length             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    Data ...
   +-+-+-+-+-+
```
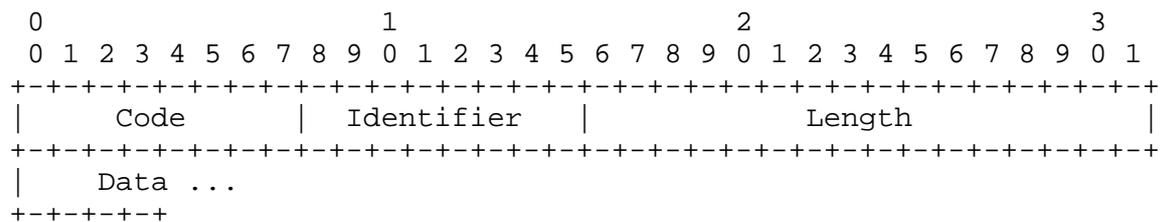
   Code

      The Code field is one octet and identifies the type of PAP packet.
      PAP Codes are assigned as follows:

```
     1          Authenticate
     2          Authenticate-Ack
     3          Authenticate-Nak
```

   Identifier

      The Identifier field is one octet and aids in matching requests
      and replies.

   Length

      The Length field is two octets and indicates the length of the PAP
      packet including the Code, Identifier, Length and Data fields.
      Octets outside the range of the Length field should be treated as
      Data Link Layer padding and should be ignored on reception.

   Data

      The Data field is zero or more octets. The format of the Data
      field is determined by the Code field.

4.2. Authenticate

   Description

      The Authenticate packet is used to begin the Password
      Authentication Protocol. An implementation having sent a LCP
      Configure-Ack packet with an Authentication-Type Configuration
      Option further specifying the Password Authentication Protocol
      must send an Authenticate packet during the Authentication phase.
      An implementation receiving a Configure-Ack with said
      Configuration Option should expect the remote end to send an
      Authenticate packet during this phase.

      An Authenticate packet is sent with the Code field set to 1
      (Authenticate) and the Peer-ID and Password fields filled as
      desired.

      Upon reception of an Authenticate, some type of Authenticate reply
      MUST be transmitted.

   A summary of the Authenticate packet format is shown below. The
   fields are transmitted from left to right.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Code      |   Identifier  |            Length             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Peer-ID Length|  Peer-Id ...
   +-+-+-+-+-+-+-+-+-+-+-+-+
   | Passwd-Length |  Password  ...
   +-+-+-+-+-+-+-+-+-+-+-+-+
```
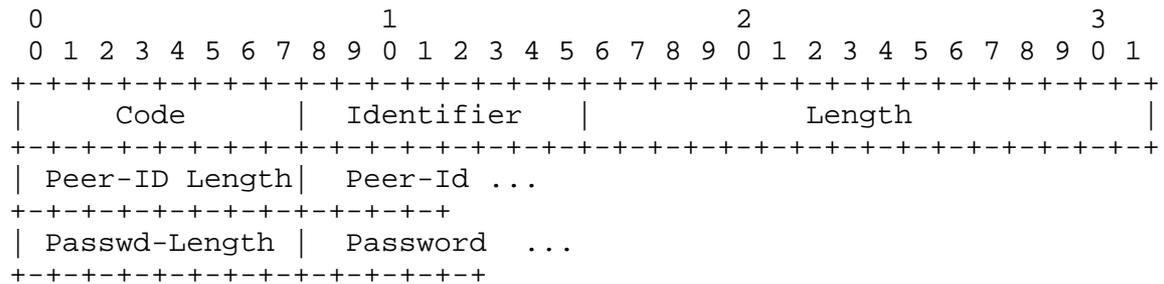
   Code

      1 for Authenticate.

   Identifier

      The Identifier field is one octet and aids in matching requests
      and replies. The Identifier field should be changed each time a
      Authenticate is transmitted which is different from the preceding
      request.

   Peer-ID-Length

      The Peer-ID-Length field is one octet and indicates the length of
      the Peer-ID field

   Peer-ID

      The Peer-ID field is zero or more octets and indicates the name of
      the peer to be authenticated.

   Passwd-Length

      The Passwd-Length field is one octet and indicates the length of
      the Password field

   Password

      The Password field is zero or more octets and indicates the
      password to be used for authentication.

   4.3. Authenticate-Ack

      Description

         If the Peer-ID/Password pair received in an Authenticate is both
         recognizable and acceptable, then a PAP implementation should
         transmit a PAP packet with the Code field set to 2 (Authenticate-
         Ack), the Identifier field copied from the received Authenticate,
         and the Message field optionally filled with an ASCII message.

      A summary of the Authenticate-Ack packet format is shown below. The
      fields are transmitted from left to right.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Code      |  Identifier   |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Msg-Length   |  Message  ...
+-+-+-+-+-+-+-+-+-+-+-+-+-
```
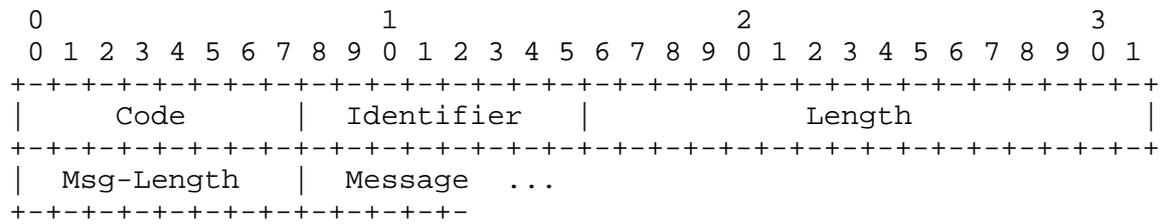
      Code

         2 for Authenticate-Ack.

      Identifier

         The Identifier field is one octet and aids in matching requests
         and replies. The Identifier field MUST be copied from the
         Identifier field of the Authenticate which caused this
         Authenticate-Ack.

      Msg-Length

         The Msg-Length field is one octet and indicates the length of the
         Message field

      Message

         The Message field is zero or more octets and indicates an ASCII
         message.

4.4. Authenticate-Nak

   Description

      If the Peer-ID/Password pair received in a Authenticate is not
      recognizable or acceptable, then a PAP implementation should
      transmit a PAP packet with the Code field set to 3 (Authenticate-
      Nak), the Identifier field copied from the received Authenticate,
      and the Message field optionally filled with an ASCII message.

   A summary of the Authenticate-Nak packet format is shown below. The
   fields are transmitted from left to right.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Code      |   Identifier  |            Length             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Msg-Length   |   Message  ...
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-
```
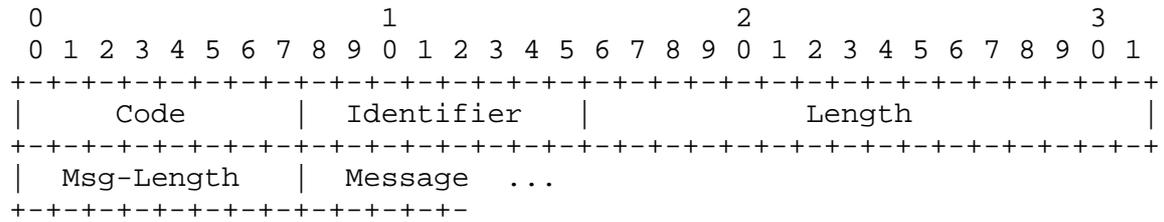
   Code

      3 for Authenticate-Nak.

   Identifier

      The Identifier field is one octet and aids in matching requests
      and replies. The Identifier field MUST be copied from the
      Identifier field of the Authenticate which caused this
      Authenticate-Nak.

   Msg-Length

      The Msg-Length field is one octet and indicates the length of the
      Message field

   Message

      The Message field is zero or more octets and indicates an ASCII
      message.

5. IP Control Protocol (IPCP) Configuration Options

IPCP Configuration Options allow negotiatiation of desirable Internet
Protocol parameters. Negotiable modifications proposed in this document
include IP addresses and compression protocols.

The initial proposed values for the IPCP Configuration Option Type field
(see [1]) are assigned as follows:

    1   IP-Addresses
    2   Compression-Type

5.1. IP-Addresses

   Description

      This Configuration Option provides a way to negotiate the IP
      addresses to be used on each end of the link. By default, no IP
      addresses are assigned to either end. An address specified as
      zero shall be interpreted as requesting the remote end to specify
      the address. If an implementation allows the assignment of
      multiple IP addresses, then it may include multiple IP Address
      Configuration Options in its Configure-Request packets. An
      implementation receiving a Configure-Request specifying multiple
      IP Address Configuration Options may send a Configure-Reject
      specifying one or more of the specified IP Addresses. An
      implementation which desires that no IP addresses be assigned
      (such as a "half-gateway") may reject all IP Address Configuration
      Options.

   A summary of the IP-Addresses Configuration Option format is shown
   below. The fields are transmitted from left to right.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |       Source-IP-Address
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   Source-IP-Address (cont)     |   Destination-IP-Address
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 Destination-IP-Address (cont)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
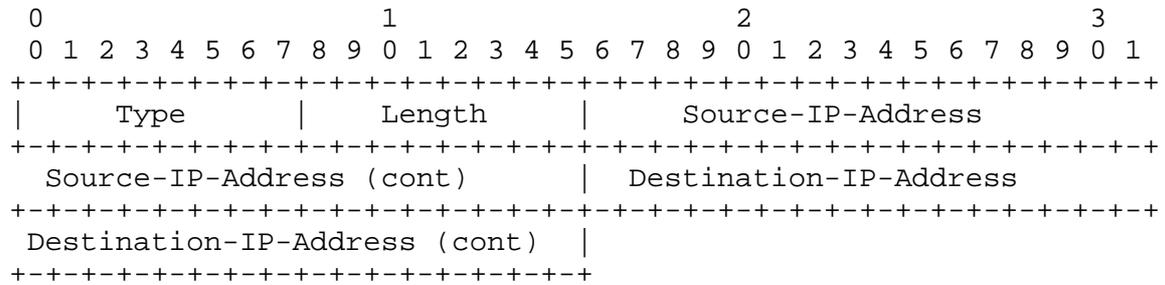
   Type

      1

   Length

      10

   Source-IP-Address

      The four octet Source-IP-Address is the desired local address of
      the sender of a Configure-Request. In a Configure-Ack,
      Configure-Nak or Configure-Reject, the Source-IP-Address is the
      remote address of the sender, and is thus a local address with
      respect to the Configuration Option receiver.

Destination-IP-Address

   The four octet Destination-IP-Address is the remote address with
   respect to the sender of a Configure-Request. In a Configure-Ack,
   Configure-Nak or Configure-Reject, the Destination-IP-Address is
   the local address of the sender, and is thus a remote address with
   respect to the Configuration Option receiver.

Default

   No IP addresses assigned.

   5.2. Compression-Type

      Description

         This Configuration Option provides a way to negotiate the use of a
         specific compression protocol. By default, compression is not
         enabled.

      A summary of the Compression-Type Configuration Option format is
      shown below. The fields are transmitted from left to right.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      Type     |    Length     |        Compression-Type       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Data ...
   +-+-+-+-+-+
```
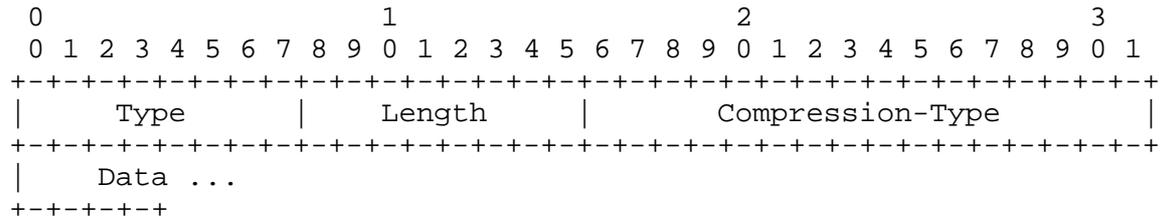
      Type

         2

      Length

         >= 4

      Compression-Type

         The Compression-Type field is two octets and indicates the type of
         compression protocol desired. Values for the Compression-Type are
         always the same as the PPP Data Link Layer Protocol field values
         for that same compression protocol. The most up-to-date values of
         the Compression-Type field are specified in "Assigned Numbers"
         [2]. Initial values are assigned as follows:

            Value (in hex)   Protocol

            0037   Van Jacobson Compressed TCP/IP

      Data

         The Data field is zero or more octets and contains additional data
         as determined by the compression protocol indicated in the
         Compression-Type field.

   Default

      No compression protocol enabled.

References

   [1]   Perkins, D., "The Point-to-Point Protocol for the Transmission
          of Multi-Protocol of Datagrams Over Point-to-Point Links", RFC
          1171, July, 1990.

   [2]   Reynolds, J., and J. Postel, "Assigned Numbers", RFC 1060,
          USC/Information Sciences Institute, March 1990.

Security Considerations

   Security issues are discussed in Section 2.3.

Author's Address

   This proposal is the product of the Point-to-Point Protocol Working
   Group of the Internet Engineering Task Force (IETF). The working
   group can be contacted via the chair:

      Russ Hobby
      UC Davis
      Computing Services
      Davis, CA 95616

      Phone: (916) 752-0236

      EMail: rdhobby@ucdavis.edu

   Questions about this memo can also be directed to:

      Drew D. Perkins
      Carnegie Mellon University
      Networking and Communications
      Pittsburgh, PA 15213

      Phone: (412) 268-8576

      EMail: ddp@andrew.cmu.edu

Acknowledgments