Network Working Group
Request for Comments: 1990
Obsoletes: 1717
Category: Standards Track

K. Sklower
University of California, Berkeley
B. Lloyd
G. McGregor
Lloyd Internetworking
D. Carr
Newbridge Networks Corporation
T. Coradetti
Sidewalk Software

August 1996

The PPP Multilink Protocol (MP)

Status of this Memo

Abstract

This document proposes a method for splitting, recombining and sequencing datagrams across multiple logical data links. This work was originally motivated by the desire to exploit multiple bearer channels in ISDN, but is equally applicable to any situation in which multiple PPP links connect two systems, including async links. This is accomplished by means of new PPP [2] options and protocols.

The differences between the current PPP Multilink specification (RFC 1717) and this memo are explained in Section 11. Any system implementing the additional restrictions required by this memo will be backwards compatible with conforming RFC 1717 implementations.

Acknowledgements

The authors specifically wish to thank Fred Baker of ACC, Craig Fox of Network Systems, Gerry Meyer of Spider Systems, Dan Brennan of Penril Datability Networks, Vernon Schryver of SGI (for the comprehensive discussion of padding), and the members of the IP over Large Public Data Networks and PPP Extensions working groups, for much useful discussion on the subject.

Table of Contents

1. Introduction 2
1.1. Motivation 2
1.2. Functional Description 3
1.3. Conventions 4
2. General Overview 4
3. Packet Formats 7
3.1. Padding Considerations 10
4. Trading Buffer Space Against Fragment Loss 10
4.1. Detecting Fragment Loss 11
4.2. Buffer Space Requirements 12
5. PPP Link Control Protocol Extensions 13
5.1. Configuration Option Types 13
5.1.1. Multilink MRRU LCP option 14
5.1.2. Short Sequence Number Header Format Option 15
5.1.3. Endpoint Discriminator Option 15
6. Initiating use of Multilink Headers 19
7. Closing Member links 20
8. Interaction with Other Protocols 20
9. Security Considerations 21
10. References 21
11. Differences from RFC 1717 22
11.1. Negotiating Multilink, per se 22
11.2. Initial Sequence Number defined 22
11.3. Default Value of the MRRU 22
11.4. Config-Nak of EID prohibited 22
11.5. Uniformity of Sequence Space 22
11.6. Commencing and Abating use of Multilink Headers 23
11.7. Manual Configuration and Bundle Assignment 23
12. Authors' Addresses 24

1. Introduction

1.1. Motivation

Basic Rate and Primary Rate ISDN both offer the possibility of opening multiple simultaneous channels between systems, giving users additional bandwidth on demand (for additional cost). Previous proposals for the transmission of internet protocols over ISDN have stated as a goal the ability to make use of this capability, (e.g., Leifer et al., [1]).

There are proposals being advanced for providing synchronization between multiple streams at the bit level (the BONDING proposals); such features are not as yet widely deployed, and may require additional hardware for end system. Thus, it may be useful to have a purely software solution, or at least an interim measure.

There are other instances where bandwidth on demand can be exploited, such as using a dialup async line at 28,800 baud to back up a leased synchronous line, or opening additional X.25 SVCs where the window size is limited to two by international agreement.

The simplest possible algorithms of alternating packets between channels on a space available basis (which might be called the Bank Teller's algorithm) may have undesirable side effects due to reordering of packets.

By means of a four-byte sequencing header, and simple synchronization rules, one can split packets among parallel virtual circuits between systems in such a way that packets do not become reordered, or at least the likelihood of this is greatly reduced.

1.2. Functional Description

The method discussed here is similar to the multilink protocol described in ISO 7776 [4], but offers the additional ability to split and recombine packets, thereby reducing latency, and potentially increase the effective maximum receive unit (MRU). Furthermore, there is no requirement here for acknowledged-mode operation on the link layer, although that is optionally permitted.

Multilink is based on an LCP option negotiation that permits a system to indicate to its peer that it is capable of combining multiple physical links into a "bundle". Only under exceptional conditions would a given pair of systems require the operation of more than one bundle connecting them.

Multilink is negotiated during the initial LCP option negotiation. A system indicates to its peer that it is willing to do multilink by sending the multilink option as part of the initial LCP option negotiation. This negotiation indicates three things:

1. The system offering the option is capable of combining multiple physical links into one logical link;

2. The system is capable of receiving upper layer protocol data units (PDU) fragmented using the multilink header (described later) and reassembling the fragments back into the original PDU for processing;

3. The system is capable of receiving PDUs of size N octets where N is specified as part of the option even if N is larger than the maximum receive unit (MRU) for a single physical link.

Once multilink has been successfully negotiated, the sending system is free to send PDUs encapsulated and/or fragmented with the multilink header.

1.3. Conventions

The following language conventions are used in the items of specification in this document:

- MUST, SHALL or MANDATORY -- the item is an absolute requirement of the specification.
- SHOULD or RECOMMENDED -- the item should generally be followed for all but exceptional circumstances.
- MAY or OPTIONAL -- the item is truly optional and may be followed or ignored according to the needs of the implementor.

2. General Overview

In order to establish communications over a point-to-point link, each end of the PPP link must first send LCP packets to configure the data link during Link Establishment phase. After the link has been established, PPP provides for an Authentication phase in which the authentication protocols can be used to determine identifiers associated with each system connected by the link.

The goal of multilink operation is to coordinate multiple independent links between a fixed pair of systems, providing a virtual link with greater bandwidth than any of the constituent members. The aggregate link, or bundle, is named by the pair of identifiers for two systems connected by the multiple links. A system identifier may include information provided by PPP Authentication [3] and information provided by LCP negotiation. The bundled links can be different physical links, as in multiple async lines, but may also be instances of multiplexed links, such as ISDN, X.25 or Frame Relay. The links may also be of different kinds, such as pairing dialup async links with leased synchronous links.

We suggest that multilink operation can be modeled as a virtual PPP link-layer entity wherein packets received over different physical link-layer entities are identified as belonging to a separate PPP network protocol (the Multilink Protocol, or MP) and recombined and sequenced according to information present in a multilink fragmentation header. All packets received over links identified as belonging to the multilink arrangement are presented to the same network-layer protocol processing machine, whether they have multilink headers or not.

The packets to be transmitted using the multilink procedure are encapsulated according to the rules for PPP where the following options would have been manually configured:

- No async control character Map
- No Magic Number
- No Link Quality Monitoring
- Address and Control Field Compression
- Protocol Field Compression
- No Compound Frames
- No Self-Describing-Padding

According to the rules specified in RFC1661, this means that an implementation MUST accept reassembled packets with and without leading zeroes present in the Protocol Field of the reassembled packet. Although it is explicitly forbidden below to include the Address and Control fields (usually, the two bytes FF 03) in the material to be fragmented, it is a good defensive programming practice to accept the packet anyway, ignoring the two bytes if present, as that is what RFC1661 specifies.

As a courtesy to implementations that perform better when certain alignment obtains, it is suggested that a determination be made when a bundle is created on whether to transmit leading zeroes by examining whether PFC has been negotiated on the first link admitted into a bundle. This determination should be kept in force so long as a bundle persists.

Of course, individual links are permitted to have different settings for these options. As described below, member links SHOULD negotiate Self-Describing-Padding, even though pre-fragmented packets MUST NOT be padded. Since the Protocol Field Compression mode on the member link allows a sending system to include a leading byte of zero or not at its discretion, this is an alternative mechanism for generating even-length packets.

LCP negotiations are not permitted on the bundle itself. An implementation MUST NOT transmit LCP Configure-Request, -Reject, -Ack, -Nak, Terminate-Request or -Ack packets via the multilink procedure, and an implementation receiving them MUST silently discard them. (By "silently discard" we mean to not generate any PPP packets in response; an implementation is free to generate a log entry registering the reception of the unexpected packet). By contrast, other LCP packets having control functions not associated with changing the defaults for the bundle itself are permitted. An implementation MAY transmit LCP Code-Reject, Protocol-Reject, Echo-Request, Echo-Reply and Discard-Request Packets.

The effective MRU for the logical-link entity is negotiated via an LCP option. It is irrelevant whether Network Control Protocol packets are encapsulated in multilink headers or not, or even over which link they are sent, once that link identifies itself as belonging to a multilink arrangement.

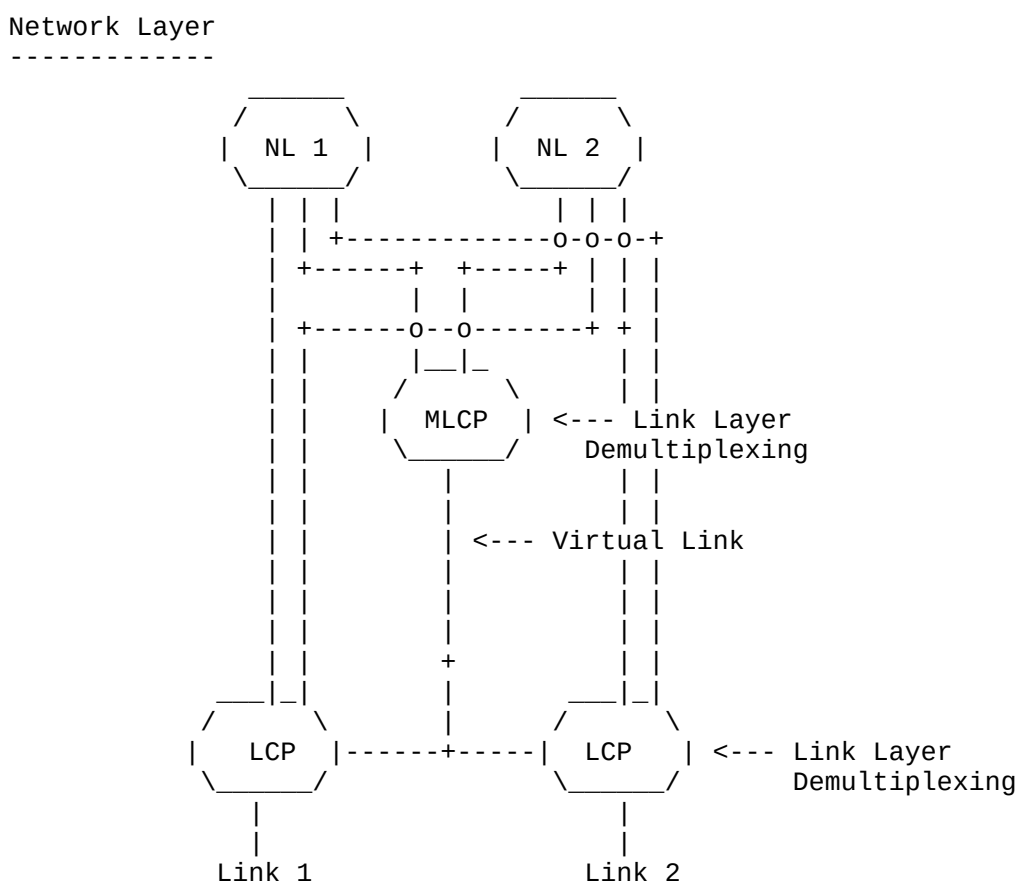
Note that network protocols that are not sent using multilink headers cannot be sequenced. (And consequently will be delivered in any convenient way).

For example, consider the case in Figure 1. Link 1 has negotiated network layers NL 1, NL 2, and MP between two systems. The two systems then negotiate MP over Link 2.

Frames received on link 1 are demultiplexed at the data link layer according the PPP network protocol identifier and can be sent to NL 1, NL 2, or MP. Link 2 will accept frames with all network protocol identifiers that Link 1 does.

Frames received by MP are further demultiplexed at the network layer according to the PPP network protocol identifier and sent to NL 1 or NL 2. Any frames received by MP for any other network layer protocols are rejected using the normal protocol reject mechanism.

Figure 1. Multilink Overview.



3. Packet Formats

In this section we describe the layout of individual fragments, which are the "packets" in the Multilink Protocol. Network Protocol packets are first encapsulated (but not framed) according to normal PPP procedures, and large packets are broken up into multiple segments sized appropriately for the multiple physical links. Although it would otherwise be permitted by the PPP spec, implementations MUST NOT include the Address and Control Field in the logical entity to be fragmented. A new PPP header consisting of the Multilink Protocol Identifier, and the Multilink header is inserted before each section. (Thus the first fragment of a multilink packet in PPP will have two headers, one for the fragment, followed by the header for the packet itself).

Systems implementing the multilink procedure are not required to fragment small packets. There is also no requirement that the segments be of equal sizes, or that packets must be broken up at all. A possible strategy for contending with member links of differing transmission rates would be to divide the packets into segments proportion to the transmission rates. Another strategy might be to divide them into many equal fragments and distribute multiple fragments per link, the numbers being proportional to the relative speeds of the links.

PPP multilink fragments are encapsulated using the protocol identifier 0x00-0x3d. Following the protocol identifier is a four byte header containing a sequence number, and two one bit fields indicating that the fragment begins a packet or terminates a packet. After negotiation of an additional PPP LCP option, the four byte header may be optionally replaced by a two byte header with only a 12 bit sequence space. Address & Control and Protocol ID compression are assumed to be in effect. Individual fragments will, therefore, have the following format:

Figure 2: Long Sequence Number Fragment Format.

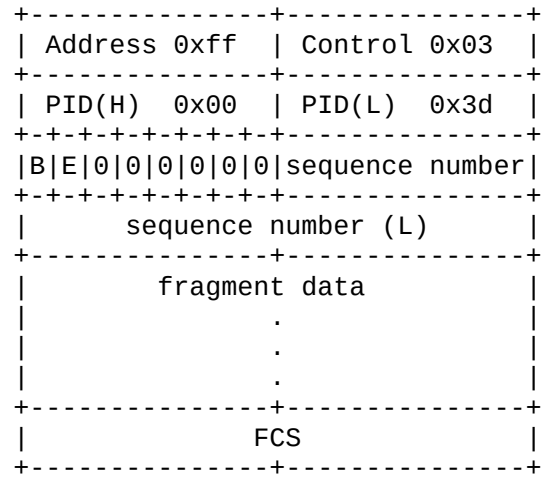
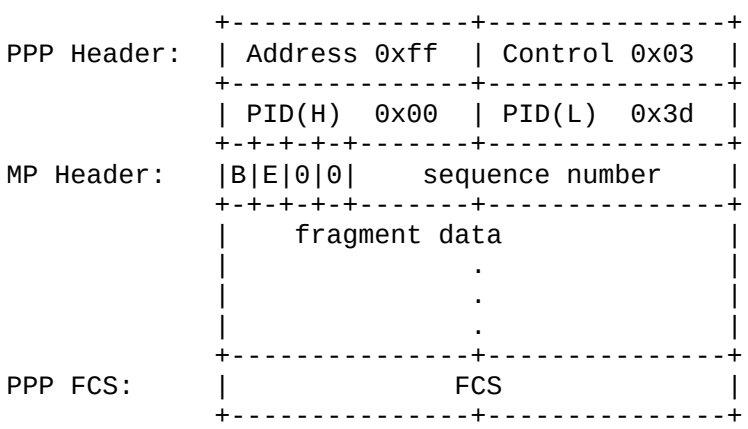


Figure 3: Short Sequence Number Fragment Format.



The (B)eginning fragment bit is a one bit field set to 1 on the first fragment derived from a PPP packet and set to 0 for all other fragments from the same PPP packet.

The (E)nding fragment bit is a one bit field set to 1 on the last fragment and set to 0 for all other fragments. A fragment may have both the (B)eginning and (E)nding fragment bits set to 1.

The sequence field is a 24 bit or 12 bit number that is incremented for every fragment transmitted. By default, the sequence field is 24 bits long, but can be negotiated to be only 12 bits with an LCP configuration option described below.

Between the (E)nding fragment bit and the sequence number is a reserved field, whose use is not currently defined, which MUST be set to zero. It is 2 bits long when the use of short sequence numbers has been negotiated, 6 bits otherwise.

In this multilink protocol, a single reassembly structure is associated with the bundle. The multilink headers are interpreted in the context of this structure.

The FCS field shown in the diagram is inherited from the normal framing mechanism from the member link on which the packet is transmitted. There is no separate FCS applied to the reconstituted packet as a whole if transmitted in more than one fragment.

3.1. Padding Considerations

Systems that support the multilink protocol SHOULD implement Self-Describing-Padding. A system that implements self-describing-padding by definition will either include the padding option in its initial LCP Configure-Requests, or (to avoid the delay of a Configure-Reject) include the padding option after receiving a NAK containing the option.

A system that must pad its own transmissions but does not use Self-Describing-Padding when not using multilink, MAY continue to not use Self-Describing-Padding if it ensures by careful choice of fragment lengths that only (E)nding fragments of packets are padded. A system MUST NOT add padding to any packet that cannot be recognized as padded by the peer. Non-terminal fragments MUST NOT be padded with trailing material by any other method than Self-Describing-Padding.

A system MUST ensure that Self-Describing-Padding as described in RFC 1570 [11] is negotiated on the individual link before transmitting any multilink data packets if it might pad non-terminal fragments or if it would use network or compression protocols that are vulnerable to padding, as described in RFC 1570. If necessary, the system that adds padding MUST use LCP Configure-NAK's to elicit a Configure-Request for Self-Describing-Padding from the peer.

Note that LCP Configure-Requests can be sent at any time on any link, and that the peer will always respond with a Configure-Request of its own. A system that pads its transmissions but uses no protocols other than multilink that are vulnerable to padding MAY delay ensuring that the peer has Configure-Requested Self-Describing-Padding until it seems desireable to negotiate the use of Multilink itself. This permits the interoperability of a system that pads with older peers that support neither Multilink nor Self-Describing-Padding.

4. Trading Buffer Space Against Fragment Loss

In a multilink procedure one channel may be delayed with respect to the other channels in the bundle. This can lead to fragments being received out of order, thus increasing the difficulty in detecting the loss of a fragment. The task of estimating the amount of space required for buffering on the receiver becomes more complex because of this. In this section we discuss a technique for declaring that a fragment is lost, with the intent of minimizing the buffer space required, yet minimizing the number of avoidable packet losses.

4.1. Detecting Fragment Loss

On each member link in a bundle, the sender MUST transmit fragments with strictly increasing sequence numbers (modulo the size of the sequence space). This requirement supports a strategy for the receiver to detect lost fragments based on comparing sequence numbers. The sequence number is not reset upon each new PPP packet, and a sequence number is consumed even for those fragments which contain an entire PPP packet, i.e., one in which both the (B)eginning and (E)nding bits are set.

An implementation MUST set the sequence number of the first fragment transmited on a newly-constructed bundle to zero. (Joining a secondary link to an exisiting bundle is invisible to the protocol, and an implementation MUST NOT reset the sequence number space in this situation).

The receiver keeps track of the incoming sequence numbers on each link in a bundle and maintains the current minimum of the most recently received sequence number over all the member links in the bundle (call this M). The receiver detects the end of a packet when it receives a fragment bearing the (E)nding bit. Reassembly of the packet is complete if all sequence numbers up to that fragment have been received.

A lost fragment is detected when M advances past the sequence number of a fragment bearing an (E)nding bit of a packet which has not been completely reassembled (i.e., not all the sequence numbers between the fragment bearing the (B)eginning bit and the fragment bearing the (E)nding bit have been received). This is because of the increasing sequence number rule over the bundle. Any sequence number so detected is assumed to correspond to a fragment which has been lost.

An implementation MUST assume that if a fragment bears a (B)eginning bit, that the previously numbered fragment bore an (E)nding bit. Thus if a packet is lost bearing the (E)nding bit, and the packet whose fragment number is M contains a (B)eginning bit, the implementation MUST discard fragments for all unassembled packets through M-1, but SHOULD NOT discard the fragment bearing the new (B)eginning bit on this basis alone.

The detection of a lost fragment, whose sequence number was deduced to be U, causes the receiver to discard all fragments up to the lowest numbered fragment with an ending bit (possibly deduced) greater than or equal to U. However, the quantity M may jump into the middle of a chain of packets which can be successful completed.

Fragments may be lost due to corruption of individual packets or catastrophic loss of the link (which may occur only in one direction). This version of the multilink protocol mandates no specific procedures for the detection of failed links. The PPP link quality management facility, or the periodic issuance of LCP echo-requests could be used to achieve this.

Senders SHOULD avoid keeping any member links idle to maximize early detection of lost fragments by the receiver, since the value of M is not incremented on idle links. Senders SHOULD rotate traffic among the member links if there isn't sufficient traffic to overflow the capacity of one link to avoid idle links.

Loss of the final fragment of a transmission can cause the receiver to stall until new packets arrive. The likelihood of this may be decreased by sending a null fragment on each member link in a bundle that would otherwise become idle immediately after having transmitted a fragment bearing the (E)nding bit, where a null fragment is one consisting only of a multilink header bearing both the (B)egin and (E)nding bits (i.e., having no payload). Implementations concerned about either wasting bandwidth or per packet costs are not required to send null fragments and may elect to defer sending them until a timer expires, with the marginally increased possibility of lengthier stalls in the receiver. The receiver SHOULD implement some type of link idle timer to guard against indefinite stalls.

The increasing sequence per link rule prohibits the reallocation of fragments queued up behind a failing link to a working one, a practice which is not unusual for implementations of ISO multilink over LAPB [4].

4.2. Buffer Space Requirements

There is no amount of buffering that will guarantee correct detection of fragment loss, since an adversarial peer may withhold a fragment on one channel and send arbitrary amounts on the others. For the usual case where all channels are transmitting, you can show that there is a minimum amount below which you could not correctly detect packet loss. The amount depends on the relative delay between the channels, (D[channel-i,channel-j]), the data rate of each channel, R[c], the maximum fragment size permitted on each channel, F[c], and the total amount of buffering the transmitter has allocated amongst the channels.

When using PPP, the delay between channels could be estimated by using LCP echo request and echo reply packets. (In the case of links of different transmission rates, the round trip times should be adjusted to take this into account.) The slippage for each channel

is defined as the bandwidth times the delay for that channel relative to the channel with the longest delay, S[c] = R[c] * D[c,c-worst]. (S[c-worst] will be zero, of course!)

A situation which would exacerbate sequence number skew would be one in which there is extremely bursty traffic (almost allowing all channels to drain), and then where the transmitter would first queue up as many consecutively numbered packets on one link as it could, then queue up the next batch on a second link, and so on. Since transmitters must be able to buffer at least a maximum- sized fragment for each link (and will usually buffer up at least two) A receiver that allocates any less than S[1] + S[2] + ... + S[N] + F[1] + ... + F[N], will be at risk for incorrectly assuming packet loss, and therefore, SHOULD allocate at least twice that.

5. PPP Link Control Protocol Extensions

If reliable multilink operation is desired, PPP Reliable Transmission [6] (essentially the use of ISO LAPB) MUST be negotiated prior to the use of the Multilink Protocol on each member link.

Whether or not reliable delivery is employed over member links, an implementation MUST present a signal to the NCP's running over the multilink arrangement that a loss has occurred.

Compression may be used separately on each member link, or run over the bundle (as a logical group link). The use of multiple compression streams under the bundle (i.e., on each link separately) is indicated by running the Compression Control Protocol [5] but with an alternative PPP protocol ID.

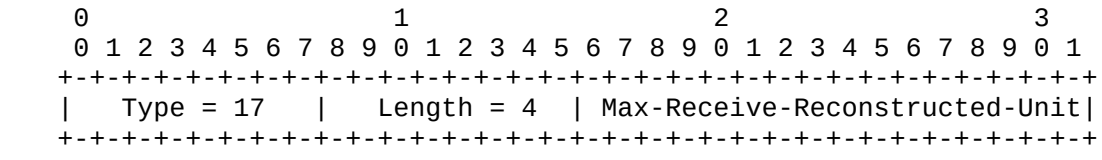
5.1. Configuration Option Types

The Multilink Protocol introduces the use of additional LCP Configuration Options:

- Multilink Maximum Received Reconstructed Unit
- Multilink Short Sequence Number Header Format
- Endpoint Discriminator

5.1.1. Multilink MRRU LCP option

Figure 4: Multilink MRRU LCP option



The presence of this LCP option indicates that the system sending it implements the PPP Multilink Protocol. If not rejected, the system will construe all packets received on this link as being able to be processed by a common protocol machine with any other packets received from the same peer on any other link on which this option has been accepted.

The Max-Receive-Reconstructed unit field is two octets, and specifies the maximum number of octets in the Information fields of reassembled packets. A system MUST be able to receive the full 1500 octet Information field of any reassembled PPP packet although it MAY attempt to negotiate a smaller, or larger value. The number 1500 here comes from the specification for the MRU LCP option in PPP; if this requirement is changed in a future version of RFC 1661, the same rules will apply here.

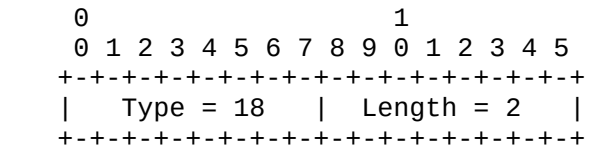
A system MUST include the LCP MRRU option in every LCP negotiation intended to instantiate a bundle or to join an existing bundle. If the LCP MRRU option is offered on a link which is intended to join an existing bundle, a system MUST offer the same Max-Receive-Reconstruct-Unit value previously negotiated for the bundle.

A system MUST NOT send any multilink packets on any link unless its peer has offered the MMRU LCP option and the system has configure-Ack'ed it during the most recent LCP negotiation on that link. A system MAY include the MMRU LCP option in a configure-NAK, if its peer has not offered it (until, according to PPP rules, the peer configure-Reject's it).

Note: the MRRU value conveyed im this option corresponds to the MRU of the bundle when conceptualized as a PPP entity; but the rules for the Multilink MRRU option are different from the LCP MRU option, as some value MUST be offered in every LCP negotiation, and that confirmation of this option is required prior to multilink interpretation.

5.1.2. Short Sequence Number Header Format Option

Figure 5: Short Sequence Number Header Format Option

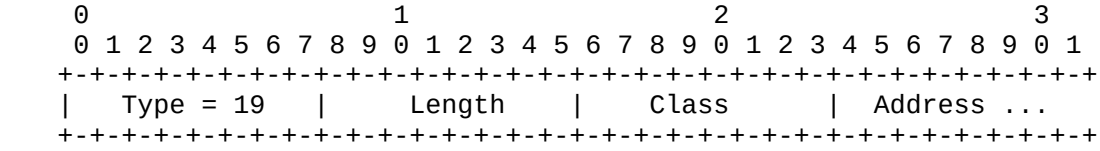


This option advises the peer that the implementation wishes to receive fragments with short, 12 bit sequence numbers. When a peer system configure-Ack's this option, it MUST transmit all multilink packets on all links of the bundle with 12 bit sequence numbers or configure-Reject the option. If 12 bit sequence numbers are desired, this option MUST be negotiated when the bundle is instantiated, and MUST be explicitly included in every LCP configure request offered by a system when the system intends to include that link in an existing bundle using 12 bit sequence numbers. If this option is never negotiated during the life of a bundle, sequence numbers are 24 bits long.

An implementation wishing to transmit multilink fragments with short sequence numbers MAY include the multilink short sequence number in a configure-NAK to ask that the peer respond with a request to receive short sequence numbers. The peer is not compelled to respond with the option.

5.1.3. Endpoint Discriminator Option

Figure 7: Endpoint Discriminator Option



The Endpoint Discriminator Option represents identification of the system transmitting the packet. This option advises a system that the peer on this link could be the same as the peer on another existing link. If the option distinguishes this peer from all others, a new bundle MUST be established from the link being negotiated. If this option matches the class and address of some other peer of an existing link, the new link MUST be joined to the bundle containing the link to the matching peer or MUST establish a new bundle, depending on the decision tree shown in (1) through (4) below.

To securely join an existing bundle, a PPP authentication protocol [3] must be used to obtain authenticated information from the peer to prevent a hostile peer from joining an existing bundle by presenting a falsified discriminator option.

This option is not required for multilink operation. If a system does not receive the Multilink MRRU option, but does receive the Endpoint Discriminator Option, and there is no manual configuration providing outside information, the implementation MUST NOT assume that multilink operation is being requested on this basis alone.

As there is also no requirement for authentication, there are four sets of scenarios:

(1) No authentication, no discriminator:
All new links MUST be joined to one bundle, unless there is manual configuration to the contrary.
It is also permissible to have more than one manually configured bundle connecting two given systems.

(2) Discriminator, no authentication:
Discriminator match -> MUST join matching bundle,
discriminator mismatch -> MUST establish new bundle.

(3) No discriminator, authentication:
Authenticated match -> MUST join matching bundle,
authenticated mismatch -> MUST establish new bundle.

(4) Discriminator, authentication:
Discriminator match and authenticated match -> MUST join bundle,
discriminator mismatch -> MUST establish new bundle,
authenticated mismatch -> MUST establish new bundle.

The option contains a Class which selects an identifier address space and an Address which selects a unique identifier within the class address space.

This identifier is expected to refer to the mechanical equipment associated with the transmitting system. For some classes, uniqueness of the identifier is global and is not bounded by the scope of a particular administrative domain. Within each class, uniqueness of address values is controlled by a class dependent policy for assigning values.

Each endpoint may chose an identifier class without restriction. Since the objective is to detect mismatches between endpoints erroneously assumed to be alike, mismatch on class alone is sufficient. Although no one class is recommended, classes which have

universally unique values are preferred.

This option is not required to be supported either by the system or the peer. If the option is not present in a Configure-Request, the system MUST NOT generate a Configure-Nak of this option for any reason; instead it SHOULD behave as if it had received the option with Class = 0, Address = 0. If a system receives a Configure-Nak or Configure-Reject of this option, it MUST remove it from any additional Configure-Request.

The size is determined from the Length field of the element. For some classes, the length is fixed, for others the length is variable. The option is invalid if the Length field indicates a size below the minimum for the class.

An implementation MAY use the Endpoint Discriminator to locate administration or authentication records in a local database. Such use of this option is incidental to its purpose and is deprecated when a PPP Authentication protocol [3] can be used instead. Since some classes permit the peer to generate random or locally assigned address values, use of this option as a database key requires prior agreement between peer administrators.

The specification of the subfields are:

Type
     19 = for Endpoint Discriminator

Length
     3 + length of Address

Class
     The Class field is one octet and indicates the identifier address space. The most up-to-date values of the LCP Endpoint Discriminator Class field are specified in the most recent "Assigned Numbers" RFC [7]. Current values are assigned as follows:

     0 Null Class

     1 Locally Assigned Address

     2 Internet Protocol (IP) Address

     3 IEEE 802.1 Globally Assigned MAC Address

     4 PPP Magic-Number Block

5 Public Switched Network Directory Number

Address

The Address field is one or more octets and indicates the identifier address within the selected class. The length and content depend on the value of the Class as follows:

Class 0 - Null Class

Maximum Length: 0

Content:
This class is the default value if the option is not present in a received Configure-Request.

Class 1 - Locally Assigned Address

Maximum Length: 20

Content:

This class is defined to permit a local assignment in the case where use of one of the globally unique classes is not possible. Use of a device serial number is suggested. The use of this class is deprecated since uniqueness is not guaranteed.

Class 2 - Internet Protocol (IP) Address

Fixed Length: 4

Content:

An address in this class contains an IP host address as defined in [8].

Class 3 - IEEE 802.1 Globally Assigned MAC Address

Fixed Length: 6

Content:

An address in this class contains an IEEE 802.1 MAC address in canonical (802.3) format [9]. The address MUST have the global/local assignment bit clear and MUST have the multicast/specific bit clear. Locally assigned MAC addresses should be represented using Class 1.

Class 4 - PPP Magic-Number Block

Maximum Length: 20

Content:

This is not an address but a block of 1 to 5 concatenated 32 bit PPP Magic-Numbers as defined in [2]. This class provides for automatic generation of a value likely but not guaranteed to be unique. The same block MUST be used by an endpoint continuously during any period in which at least one link is in the LCP Open state. The use of this class is deprecated.

Note that PPP Magic-Numbers are used in [2] to detect unexpected loopbacks of a link from an endpoint to itself. There is a small probability that two distinct endpoints will generate matching magic-numbers. This probability is geometrically reduced when the LCP negotiation is repeated in search of the desired mismatch, if a peer can generate uncorrelated magic-numbers.

As used here, magic-numbers are used to determine if two links are in fact from the same peer endpoint or from two distinct endpoints. The numbers always match when there is one endpoint. There is a small probability that the numbers will match even if there are two endpoints. To achieve the same confidence that there is not a false match as for LCP loopback detection, several uncorrelated magic-numbers can be combined in one block.

Class 5 - Public Switched Network Directory Number

Maximum Length: 15

Content:

An address in this class contains an octet sequence as defined by I.331 (E.164) representing an international telephone directory number suitable for use to access the endpoint via the public switched telephone network [10].

6. Initiating use of Multilink Headers

When the use of the Multilink protocol has been negotiated on a link (say Y), and the link is being added to a bundle which currently contains a single existing link (say X), a system MUST transmit a Multilink-encapsulated packet on X before transmitting any Multilink-

encapsulated packets on Y.

Since links may be added and removed from a bundle without destroying the state associated with it, the fragment should be assigned the appropriate (next) fragment number. As noted earlier, the first fragment transmitted in the life of a bundle is assigned fragment number 0.

7. Closing Member links

Member links may be terminated according to normal PPP LCP procedures using LCP Terminate-Request and Terminate-Ack packets on that member link. Since it is assumed that member links usually do not reorder packets, receipt of a terminate ack is sufficient to assume that any multilink protocol packets ahead of it are at no special risk of loss.

Receipt of an LCP Terminate-Request on one link does not conclude the procedure on the remaining links.

So long as any member links in the bundle are active, the PPP state for the bundle persists as a separate entity. However, if the there is a unique link in the bundle, and all the other links were closed gracefully (with Terminate-Ack), an implementation MAY cease using multilink
headers.

If the multilink procedure is used in conjunction with PPP reliable transmission, and a member link is not closed gracefully, the implementation should expect to receive packets which violate the increasing sequence number rule.

8. Interaction with Other Protocols

In the common case, LCP, and the Authentication Control Protocol would be negotiated over each member link. The Network Protocols themselves and associated control exchanges would normally have been conducted once, on the bundle.

In some instances it may be desirable for some Network Protocols to be exempted from sequencing requirements, and if the MRU sizes of the link did not cause fragmentation, those protocols could be sent directly over the member links.

Although explicitly discouraged above, if there were several member links connecting two implementations, and independent sequencing of two protocol sets were desired, but blocking of one by the other was not, one could describe two multilink procedures by assigning

multiple endpoint identifiers to a given system. Each member link, however, would only belong to one bundle. One could think of a physical router as housing two logically separate implementations, each of which is independently configured.

A simpler solution would be to have one link refuse to join the bundle, by sending a Configure-Reject in response to the Multilink LCP option.

9. Security Considerations

Operation of this protocol is no more and no less secure than operation of the PPP authentication protocols [3]. The reader is directed there for further discussion.

10. References

[1] Leifer, D., Sheldon, S., and B. Gorsline, "A Subnetwork Control Protocol for ISDN Circuit-Switching", University of Michigan (unpublished), March 1991.

[2] Simpson, W., Editor, "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, Daydreamer, July 1994.

[3] Lloyd, B., and W. Simpson, "PPP Authentication Protocols", RFC 1334, Lloyd Internetworking, Daydreamer, October 1992.

[4] International Organisation for Standardization, "HDLC - Description of the X.25 LAPB-Compatible DTE Data Link Procedures", International Standard 7776, 1988

[5] Rand, D., "The PPP Compression Control Protocol (CCP)", PPP Extensions Working Group, RFC 1962, June 1996.

[6] Rand, D., "PPP Reliable Transmission", RFC 1663, Novell, July 1994

[7] Reynolds, J., and J. Postel, "Assigned Numbers", STD 2, RFC 1700, USC/Information Sciences Institute, October 1994.

[8] Postel, J., Editor, "Internet Protocol - DARPA Internet Program Protocol Specification", STD 5, RFC 791, USC/Information Sciences Institute, September 1981.

[9] Institute of Electrical and Electronics Engineers, Inc., "IEEE Local and Metropolitan Area Networks: Overview and Architecture", IEEE Std. 802-1990, 1990.

[10] The International Telegraph and Telephone Consultative Committee (CCITT), "Numbering Plan for the ISDN Area", Recommendation I.331 (E.164), 1988.

[11] Simpson, W., Editor, "PPP LCP Extensions", RFC 1570, Daydreamer, January 1994.

11. Differences from RFC 1717

This section documents differences from RFC 1717. There are restrictions placed on implementations that were absent in RFC 1717; systems obeying these restrictions are fully interoperable with RFC 1717 - compliant systems.

11.1. Negotiating Multilink, per se

RFC 1717 permitted either the use of the Short Sequence Number Header Format (SSNHF) or the Maximum Reconstructed Receive Unit (MRRU) options by themselves to indicate the intent to negotiate multilink. This specification forbids the use of the SSNHF option by itself; but does permit the specific of both options together. Any implementation which otherwise conforms to rfc1717 and also obeys this restriction will interoperate with any RFC 1717 implementation.

11.2. Initial Sequence Number defined

This specification requires that the first sequence number transmitted after the virtual link has reached to open state be 0.

11.3. Default Value of the MRRU

This specfication removes the default value for the MRRU, (since it must always be negotiated with some value), and specifies that an implementation must be support an MRRU with same value as the default MRU size for PPP.

11.4. Config-Nak of EID prohibited

This specification forbids the config-Naking of an EID for any reason.

11.5. Uniformity of Sequence Space

This specification requires that the same sequence format be employed on all links in a bundle.

11.6. Commencing and Abating use of Multilink Headers

This memo specifies how one should start the use of Multilink Headers when a link is added, and under what circumstances it is safe to discontinue their use.

11.7. Manual Configuration and Bundle Assignment

The document explicitly permits multiple bundles to be manually configured in the absence of both the Endpoint Descriminator and any form of authentication.

13. Authors' Addresses

Keith Sklower
Computer Science Department
384 Soda Hall, Mail Stop 1776
University of California
Berkeley, CA 94720-1776

Phone: (510) 642-9587
EMail: sklower@CS.Berkeley.EDU

Brian Lloyd
Lloyd Internetworking
3031 Alhambra Drive
Cameron Park, CA 95682

Phone: (916) 676-1147
EMail: brian@lloyd.com

Glenn McGregor
Lloyd Internetworking
3031 Alhambra Drive
Cameron Park, CA 95682

Phone: (916) 676-1147
EMail: glenn@lloyd.com

Dave Carr
Newbridge Networks Corporation
600 March Road
P.O. Box 13600
Kanata, Ontario,
Canada, K2K 2E6

Phone: (613) 591-3600
EMail: dcarr@Newbridge.COM

Tom Coradetti
Sidewalk Software
1190 Josephine Road
Roseville, MN 55113

Phone: (612) 490 7856
EMail: 70761.1664@compuserve.com