Network Working Group
Request for Comments: 2125
Category: Standards Track

C. Richards
Shiva Corporation
K. Smith
Ascend Communications, Inc.

March 1997

The PPP Bandwidth Allocation Protocol (BAP)
The PPP Bandwidth Allocation Control Protocol (BACP)

Status of this Memo

Abstract

This document proposes a method to manage the dynamic bandwidth allocation of implementations supporting the PPP multilink protocol [2]. This is done by defining the Bandwidth Allocation Protocol (BAP), as well as its associated control protocol, the Bandwidth Allocation Control Protocol (BACP). BAP can be used to manage the number of links in a multilink bundle. BAP defines datagrams to co-ordinate adding and removing individual links in a multilink bundle, as well as specifying which peer is responsible for which decisions regarding managing bandwidth during a multilink connection.

Table of Contents

1. Introduction

As PPP multilink implementations become increasingly common, there is a greater need for some conformity in how to manage bandwidth over such links. BACP and BAP provide a flexible yet robust way of managing bandwidth between 2 peers. BAP does this by defining Call-Control packets and a protocol that allows peers to co-ordinate the actual bandwidth allocation and de-allocation. Phone number deltas may be passed in the Call-Control packets to minimize the end user's configuration.

1.1. Specification of Requirements

In this document, several words are used to signify the requirements of the specification. These words are often capitalized.

MUST This word, or the adjective "required", means that the definition is an absolute requirement of the specification.

MUST NOT This phrase means that the definition is an absolute prohibition of the specification.

SHOULD This word, or the adjective "recommended", means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications must be understood and carefully weighed before choosing a different course.

MAY This word, or the adjective "optional", means that this item is one of an allowed set of alternatives. An implementation which does not include this option MUST be prepared to interoperate with another implementation which does include the option.

1.2. Terminology

This document frequently uses the following terms:

peer The other end of the point-to-point link

silently discard
This means the implementation discards the packet without further processing. The implementation SHOULD provide the capability of logging the error, including the contents of the silently discarded packet, and SHOULD record the event in a statistics counter.

BOD (bandwidth on demand)
BOD refers to the ability of a system to allocate and remove links in a multilink system to change the bandwidth of a multilink bundle. This may be done in response to changing line conditions and it also may be done in response to changing resource conditions. In either case, changing bandwidth dynamically during a multilink connection is referred to as BOD.

2. New LCP Configuration Option

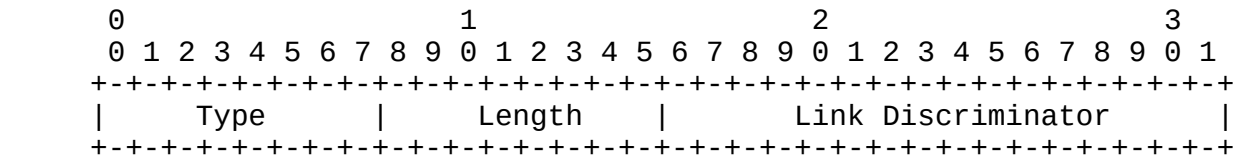
Implementations MUST implement LCP as defined in [1]. LCP MUST be in the Network-Layer Protocol phase before BACP can be negotiated.

2.1. Link Discriminator

Description

This LCP Configuration Option is used to declare a unique discriminator for the link that the option is sent over. This option MUST be negotiated by LCP on every link. BAP uses the link discriminator to differentiate the various links in a multilink bundle. Each link in a multilink bundle MUST have a unique discriminator. The discriminator is independent for each peer, so each link may have 2 different LCP Link Discriminator values, one for each peer. When the Link Discriminator is sent in a BAP packet, the transmitter sends the Link Discriminator Option value received from its peer in the peer’s LCP Configure Request packet.

A summary of the Link Discriminator LCP Option format is shown below. The fields are transmitted from left to right.



Type

23 for Link Discriminator option.

Length

4

Link Discriminator

The Link Discriminator field is 2 octets in Length, and it contains a unique identifier used to indicate a particular link in a multilink bundle. The Link Discriminator for a link MUST be unique among the Link Discriminators assigned by this endpoint for this bundle. The Link Discriminator MAY be assigned in a sequential, monotonically increasing manner.

3. BACP Operation

BACP uses the same packet exchange mechanism as the Link Control Protocol defined in [1]. BACP packets MUST NOT be exchanged until PPP has reached the Network-Layer Protocol phase. BACP packets received before this phase is reached should be silently discarded.

BACP is negotiated once per multilink bundle. If BACP is negotiated on any of the links in a multilink bundle, it is opened for all of the links in the bundle.

The Bandwidth Allocation Control Protocol is exactly the same as the Link Control Protocol [1] with the following exceptions:

Data Link Layer Protocol Field

Exactly one BACP packet is encapsulated in the Information field of PPP Data Link Layer frames where the Protocol field indicates Type hex c02b (Bandwidth Allocation Control Protocol).

Code field

Only Codes 1 through 7 (Configure-Request, Configure-Ack, Configure-Nak, Configure-Reject, Terminate-Request, Terminate-Ack and Code-Reject) are used. Other Codes should be treated as unrecognized and should result in Code-Rejects.

Configuration Option Types

BACP has a distinct set of Configuration Options, which are defined in the next section.

4. BACP Configuration Options

BACP Configuration Options allow negotiation of desirable BACP parameters. These options are used in Config-Request, Config-Ack, Config-Nak, and Config-Reject packets. BACP uses the same Configuration Option format defined for LCP [1], with a seperate set of Options.

Current values of BACP Configuration Options are assigned as follows:

1 Favored-Peer

4.1. Favored-Peer

Description

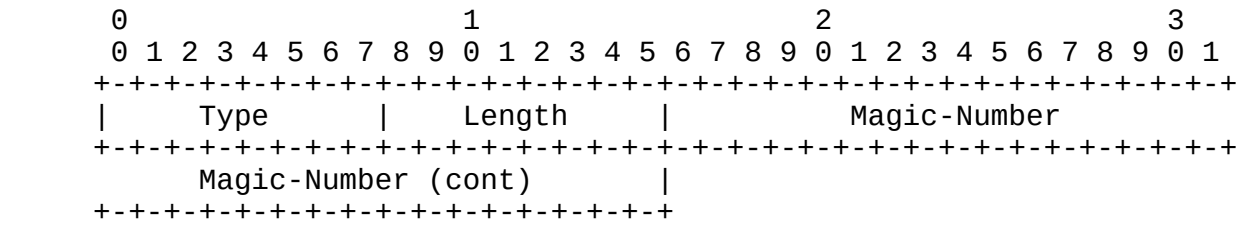
This Configuration Option is used to determine which peer is favored in the event of a race condition in which 2 peers simultaneously transmit the same BAP request. Each peer negotiates a 4 octet magic number, which is successfully negotiated when the 2 Magic-Numbers are different. The favored peer is the peer that transmits the lowest Magic-Number in its Favored-Peer Configuration Option.

The Favored-Peer Configuration Option MUST be implemented.

BACP will usually be negotiated after only one link of a multilink bundle has reached the Network-Layer Protocol phase. In this situation, it is acceptable for the peer that initiated the connection to use a Magic-Number of 1, and the peer that responded to the connection to use a Magic-Number of 0xFFFFFFFF. If a multilink bundle has been established with links that were originated by each peer, or if it is not clear which peer has initiated a link (on a leased line, for example), then a random number MUST be used for the Magic-Number. Refer to the description of the LCP Magic-Number Configuration Option in [1] for an explanation of how to create a useful random number.

When a Configure-Request is received with a Favored-Peer Configuration Option, the received Magic-Number is compared with the Magic-Number of the last Configure-Request sent to the peer. If the two Magic-Numbers are different, then the Favored-Peer negotiation has been successful, and the Favored-Peer Option SHOULD be acknowledged. If the two Magic-Numbers are equal, a Configure-Nak MUST 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 (that is, until a Configure-Nak is received or the Restart timer runs out).

A summary of the Favored-Peer Option format is shown below. The fields are transmitted from left to right.



Type

1 for Favored-Peer

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 always be Nak'd.

5. BAP Operation

5.1. Link Management

BAP defines packets, parameters and negotiation procedures to allow two endpoints to negotiate gracefully adding and dropping links from a multilink bundle. An implementation can:

- Request permission to add a Link to a bundle (Call-Request)
- Request that the peer add a link to a bundle via a callback (Callback-Request)
- Negotiate with the peer to drop a link from a bundle (this implies that the peer can refuse) (Link-Drop-Query-Request)

After BACP reaches the opened state, either peer MAY request that another link be added to the bundle by sending a BAP Call- or Callback-Request packet. A Call-Request packet is sent if the implementation wishes to originate the call for the new link, and a Callback-Request packet is sent if the implementation wishes its peer to originate the call for the new link. The implementation receiving a Call- or Callback-Request MUST respond with a Call- or Callback-Response with a valid Response Code.

After BACP reaches the opened state, either peer MAY request that a link be dropped from the bundle. A BAP Link-Drop-Query-Request packet is sent to the peer to negotiate dropping a link. The peer MUST respond with a Link-Drop-Query-Response. If the peer is agreeable to dropping the link the implementation MUST issue an LCP Terminate-Request to initiate dropping the link.

If an implementation wishes to force dropping a link without negotiation, it should simply send an LCP Terminate-Request packet on the link (without sending any BAP Link-Drop-Query-Request).

After an LCP Terminate-Request is sent an implementation SHOULD stop transmitting data packets on that link, but still continue to receive and process data packets normally until receipt of a Terminate-Ack from the peer. The receiver of an LCP Terminate-Request SHOULD stop transmitting packets before issuing the Terminate-Ack. This procedure will insure that no data is lost in either direction.

5.2. Bandwidth Management

BAP allows two peer implementations to manage the bandwidth available to the protocols using the multilink bundle by negotiating when to add and drop links (See Link Management). Use of the negotiation features of BAP makes it unnecessary to require a 'common' algorithm for determining when to add and remove links in a multilink bundle.

BOD decisions can be based on link utilization. A BAP implementation may monitor its transmit traffic, both transmit and receive traffic, or choose not to monitor traffic in either direction. If a server system implements bi-directional monitoring, it will allow BOD operation with a client that does not monitor traffic in either direction, which will minimize the end-user's configuration. When an implementation decides that it is time to remove a link due to traffic monitoring, it MUST transmit a Link-Drop-Query-Request to inquire if the peer agrees to drop a link from the current multilink bundle. When an implementation receives a Link-Drop-Query-Request, it SHOULD base its response on the traffic it is monitoring. It MUST NOT base its response solely on its receive data heuristics.

The operation of the Link-Drop-Query-Request and -Response datagrams causes a link in a multilink bundle to be left up as long as either implementation that is monitoring link utilization determines that it is necessary.

BOD decisions can also be based on the resources (e.g., physical port, B-channel, etc.) available to an implementation. For example, an implementation might remove a link from a multilink bundle to answer an incoming voice call, or might add a link when a line becomes free due to the termination of a separate PPP call on another port. An implementation MUST use an LCP Terminate-Request to remove a link due to a resource condition.

5.3. BAP Packets

All of the BAP Request and Indication packets require a Response packet in response before taking any action.

An implementation MUST set a timer when sending a Request or Indication packet. The value of this timer SHOULD depend on the type and speed of the link or links in use. Upon expiration of this timer, the implementation MUST retransmit the request or indication, with an identical identification number. This procedure will insure that the peer receives the proper request or indication even if a packet is lost during transmission. If a response packet is lost the peer will realize that this is not a new request or indication packet.

If the number of retransmissions exceeds the number supported by the implementation for this packet, the implementation MAY take appropriate recovery action. For example, if no response to a Link-Drop-Query-Request is received after 2 retransmissions, an implementation MAY initiate dropping the link by sending an LCP Terminate-Request for that link.

Since BAP packets help determine the amount of bandwidth available to an implementation, PPP SHOULD give them priority over other data packets when transmitting. This will help insure the prompt addition and removal of links in a multilink bundle. This is especially important when adding links to a bundle due to bandwidth constraints.

5.4. Race Conditions

In order to resolve race conditions, an implementation MUST implement the BACP Favored-Peer Configuration Option.

A race condition can occur if both implementations send a Call-Request, Callback-Request or Link-Drop-Query-Request at the same time. These race conditions should be solved as follows:

If each implementation sends a Call-Request or Callback-Request at the same time, the implementation with the lowest BACP Favored-Peer Magic-Number value SHOULD be favored.

If each implementation sends a Link-Drop-Query-Request at the same time, the same scheme SHOULD be used as for Call-Requests.

5.5. BAP Datagram Format

Description

Before any BAP packets may be communicated, PPP MUST reach the Network-Layer Protocol phase, and BACP MUST reach the opened state.

Exactly one BAP packet is encapsulated in the Information field of PPP Data Link Layer frames where the Protocol field indicates type hex c02d (Bandwidth Allocation Protocol).

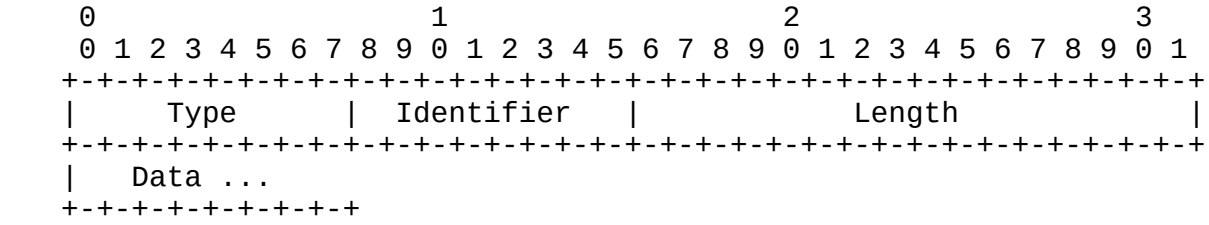
Because ISDN Terminal Adapters sometimes are used to do multilink with a non-multilink aware client, BAP datagrams MUST NOT be compressed or encrypted. Otherwise, the ISDN TA may not be able to properly intercept BAP datagrams needed to control the multilink connection. This refers to compression of the whole datagram; Address-and-Control-Field-Compression and Protocol-Field-Compression are allowed if properly negotiated.

The maximum length of a BAP packet transmitted over a PPP link is the same as the maximum length of the Information field of a PPP data link layer frame.

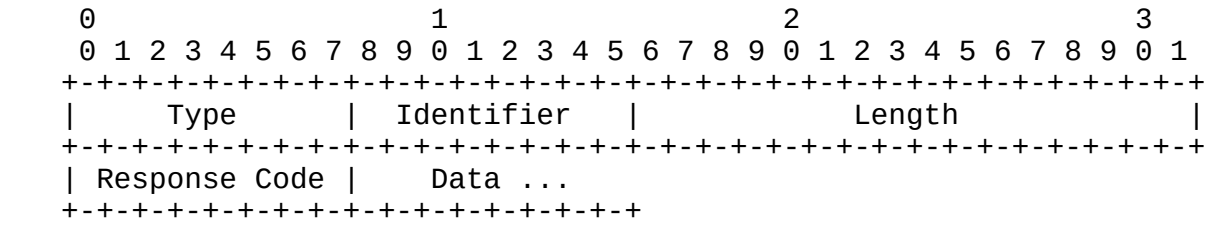
Bandwidth Allocation Protocol datagrams can be catagorized as either Request, Indication or Response packets. Every Request and Indication datagram has a corresponding Response packet. Request and Indication datagrams have a slightly different format from Response datagrams, as the Response datagrams include a Response Code octet.

All of the BAP datagrams MUST be supported by an implementation. However, that does not mean an implementation must support all BAP datagram actions. An implementation MAY send a Request-Rej to a Request that it does not implement.

A summary of the Bandwidth Allocation Protocol datagram Request and Indication packet format is shown below. The fields are transmitted from left to right.



A summary of the Bandwidth Allocation Protocol datagram Response packet format is shown below. The fields are transmitted from left to right.



Type

The Type field is one octet and identifies the type of BAP datagram packet. Datagram types are defined as follows. This field is coded in binary coded hexadecimal.

01	Call-Request
02	Call-Response
03	Callback-Request
04	Callback-Response
05	Link-Drop-Query-Request
06	Link-Drop-Query-Response
07	Call-Status-Indication
08	Call-Status-Response

The various types of BAP datagrams are explained in the following sections.

Identifier

The Identifier field is one octet and is binary coded. It aids in matching Requests and Indications with Responses. Call-Status-Indication packets MUST use the same Identifier as was used by the original Call-Request or Callback-Request that was used to initiate the call. All other Request or Indication packets MUST use a unique Identifier for each new Request or Indication. All Response packets MUST use the same Identifier as the Identifier in the Request or Indication packet being responded to. When re-transmitting a request or indication, the Identifier MUST be the same as the Identifier used on the previous transmission of the request or indication.

Length

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

Response Code

The Response Code is only present in Response datagrams. It is binary coded and can have the following values:

00000000	Request-Ack
00000001	Request-Nak
00000010	Request-Rej
00000011	Request-Full-Nak

The Request-Ack Response Code is sent to indicate that the Request or Indication command is valid and was successfully received by an implementation. The Request-Nak Response Code is sent to indicate that the Request command was received, but an implementation does

not want the requested action performed at this time. If a Response containing a Request-Nak Response Code is received, the original Request MAY be retried after an implementation determines that sufficient time has elapsed. The Request-Rej Response Code is sent to indicate that the Request command received by an implementation is not implemented (i.e., if reception of a particular request type is not supported by the peer.) The Request-Full-Nak Response Code is sent to indicate that the Request command was received, but an implementation does not want the requested action performed. The Request-Full-Nak is used to indicate that an implementation has reached the maximum (for a Call- or Callback-Request) or the minimum (for a Link-Drop-Query-Request) bandwidth configured or available for this multilink bundle. If a Response containing a Request-Full-Nak Response Code is received, the original Request SHOULD NOT be retried until the total bandwidth of the multilink bundle has changed.

Data

The Data field is variable in length, and will usually contain the list of zero or more BAP Options that the sender desires to transmit. The format of BAP Options is described in a later chapter.

5.5.1. Call-Request

Before originating a call to add another link to a multilink bundle, an implementation MUST transmit a Call-Request packet. This will inform the receiver of the request to add another link to the bundle and give the receiver a chance to inform the implementation of the phone number of a free port that can be called.

The options field MUST include the Link-Type option. The options field MAY include the No-Phone-Number and/or the Reason options.

Upon reception of a Call-Request, a Call-Response datagram MUST be transmitted.

5.5.2. Call-Response

An implementation MUST transmit a Call-Response datagram in response to a received Call-Request datagram. If the Call-Request is acceptable, the Call-Response MUST have a Response Code of Request-Ack. The Phone-Delta option MUST be included in a Call-Response packet with a Response Code of Request-Ack unless the Call-Request included the No-Phone-Number option. The options field MAY include the Reason and/or Link-Type options.

5.5.3. Callback-Request

An implementation that wants its peer to originate another link to add to the multilink bundle MUST transmit a Callback-Request packet to its peer. This will inform the receiver of the request to add another link to the bundle along with the number to be called.

The options field MUST include the Link-Type and Phone-Delta options. The Reason option MAY also be included.

Upon reception of a Callback-Request, a Callback-Response datagram MUST be transmitted.

5.5.4. Callback-Response

An implementation MUST transmit a Callback-Response datagram in response to a received Callback-Request datagram. If the Callback-Request is acceptable, the Callback-Response MUST have a Response Code of Request-Ack. A Callback-Response packet MAY include the Link-Type option.

5.5.5. Link-Drop-Query-Request

An implementation that determines that a link is no longer needed and wishes to negotiate dropping it (e.g., based on a throughput BOD decision), MUST transmit a Link-Drop-Query-Request packet. The options field MUST include the Link-Discriminator option (containing the receiver’s Link-Discriminator), and MAY include the Reason option.

Upon reception of a Link-Drop-Query-Request, an implementation MUST transmit a Link-Drop-Query-Response datagram. The Response-Code will be Request-Ack if it agrees to drop the link; if it does not agree to drop the link the Response-Code will be Request-Nak or Request-Full-Nak. After the receipt of a Link-Drop-Query-Response with a Response Code of Request-Ack, the transmitter of the Link-Drop-Query-Request MUST initiate tear down of the indicated link by sending an LCP Terminate-Request packet on the designated link.

5.5.6. Link-Drop-Query-Response

An implementation transmits a Link-Drop-Query-Response datagram in response to a received Link-Drop-Query-Request datagram. If the implementation agrees (e.g., based on its throughput BOD algorithm) to reduce the bandwidth of the multilink bundle, then the Response Code MUST be set to Request-Ack.

The Reason option MAY be included in the Link-Drop-Query-Response packet.

The Link-Drop-Query-Request datagram MUST be supported, as well as the underlying implementation to respond to it. This means that a Link-Drop-Query-Response with a Response Code of Request-Rej MUST NOT be transmitted in response to a Link-Drop-Query-Request.

5.5.7. Call-Status-Indication

After an implementation attempts to add a link to a bundle as the result of a Call-Request or a Callback-Request, it MUST send a Call-Status-Indication packet to its peer to indicate if the attempt to add the link succeeded or failed. One Indication MUST be sent for each attempt made. For each Call-Status-Indication packet transmitted with the Call-Status Option Action octet set to Retry, a subsequent Call-Status-Indication packet MUST be sent to indicate the success or failure of the retry. The Call-Status option MUST be included to inform the receiver of the status of the attempt to add a link and the action the implementation will take in case of failure. The reason option MAY also be included in the Call-Status-Indication packet.

Upon reception of a Call-Status-Indication packet which indicates a failure, an implementation may log the failure and reason code. Upon reception of any Call-Status-Indication packet, a Call-Status-Response datagram MUST be transmitted.

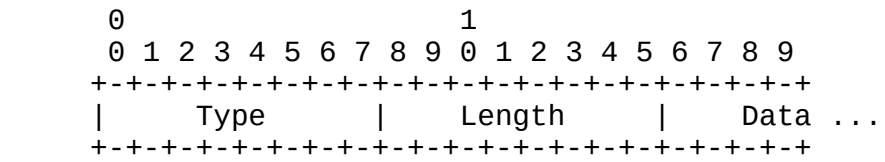
5.5.8. Call-Status-Response

An implementation transmits a Call-Status-Response datagram in response to a received Call-Status-Indication datagram. The Response Code field MUST be set to Request-Ack in this packet. The Reason option MAY be included in this packet.

6. BAP Datagram Options

BAP Datagram Options are used in various BAP packets. Their use in various packets is as defined below. The format of these options loosely follows the formatting conventions of LCP Configuration Options. When there are multiple BAP Options in one BAP packet, the options MAY be transmitted in any order.

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



Type

The type field is one octet, and indicates the type of the BAP Datagram Option. This field is binary coded Hexadecimal. The following options are currently defined:

01 Link-Type
02 Phone-Delta
03 No-Phone-Number-Needed
04 Reason
05 Link-Discriminator
06 Call-Status

Length

The Length field is one octet, and indicates the length of this BAP Option including the Type, Length, and Data fields.

Data

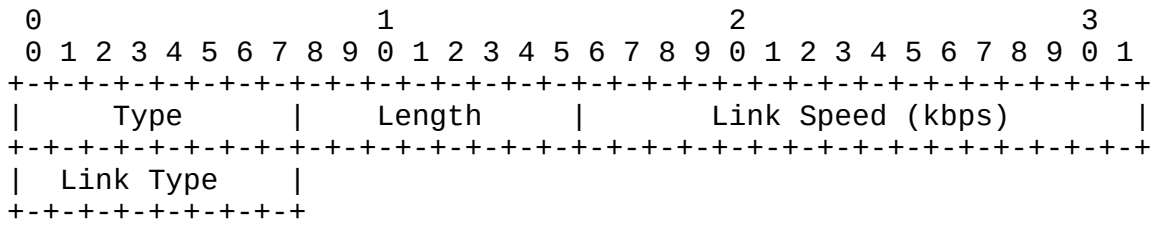
The Data field is zero or more octets, and contains information specific to the BAP Option. The format and length of the Data field is determined by the Type and Length fields.

6.1. Link-Type

Description

This option indicates the general type of link indicated for the operation being performed. This option does not indicate a specific link type, rather it gives some general characteristics of the desired link type. This option MAY be used along with other knowledge (i.e., the type of the other link(s) in the bundle or user configuration) to determine the type of link desired to be used in the operation. It MUST be included in a Call- or Callback-Request, and MAY be included in a Call- or Callback-Response.

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



Type

01 for Link-Type.

Length

The Length field is one octet, and indicates the length of this BAP Option including the Type, Length and Link Type fields.

Link Speed

The Link Speed field is 2 octets, and indicates the requested speed of the desired link in kilobits per second. This field is coded as 2 binary coded hexadecimal octets, with the most significant octet sent first.

Link Type

The Link Type field is a bit mask. It is 1 octet in length. Bit 0 of the Link Type field corresponds to bit 39 of the Link-Type BAP Option as described above. If a bit is set, it indicates support of the corresponding link type. If the link indicated is different than the supported link types, no bit will be set. Otherwise, at least one bit MUST be set. If an implementation supports more than one link type, more than one bit MAY be set.

Bit	Link type
0	ISDN
1	X.25
2	analog
3	switched digital (non-ISDN)
4	ISDN data over voice
5-7	reserved

If the Length field contains more bits than are defined by this specification, then any bits that are not defined should be

ignored. In order to allow for future expansion of this field, it is important to properly support receiving a Link Type field longer than what is defined by this specification. If the Length field is shorter than the number of bits defined, then the implementation should set all bits not received to 0.

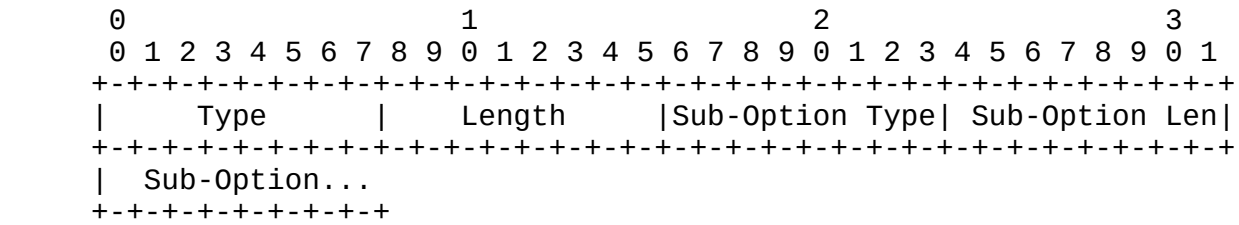
6.2. Phone-Delta

Description

The BAP Phone-Delta Option is used by an implementation to give its peer the information needed to make a call. Due to the difficulty of determining which dialing prefixes (if any) are necessary to dial a given phone number/national destination code/country code combination, the phone number to be dialed will be based on a previously known number. This MAY be the original number used to establish the first link of the multilink bundle, a number configured by the user, the phone number used to make a callback connection, or a number determined in some other way. The Phone-Delta Option will consist of a Subscriber-Number Sub-Option along with a Unique-Digits Sub-Option that indicates how many of the digits of the Subscriber-Number are unique among the ports in use, previously used, and to be used in the multilink bundle. There is also an optional Phone-Number-Sub-Address Sub-Option.

An implementation MAY include more than one Phone-Delta option in a response. This indicates that there is more than one phone number that can be used for the requested operation. The Phone-Delta option MUST appear in a Callback-Request. It also MUST appear in a Call-Response with a Response Code set to Request-Ack if the Call-Request did not contain the No-Phone-Number option. It MAY be included in the Call-Status-Indication packet.

A summary of the Phone-Delta BAP Option format is shown below. The fields are transmitted from left to right.



Type

02 for Phone-Delta.

Length

The Length field is one octet, and indicates the length of this BAP Option including the Type, Length, and Sub-Option fields.

Sub-Option Type

The following Sub-Option Types are defined for the Phone-Delta option.

01 Unique-Digits
02 Subscriber-Number
03 Phone-Number-Sub-Address

Sub-Option Length

The Sub-Option Length field is one octet, and indicates the length of this BAP Sub-Option including the Sub-Option Type, Sub-Option Length, and Sub-Option fields.

6.2.1. Phone-Delta Sub-Options

Unique-Digits

The Unique-Digits Sub-Option field consists of one octet that is a count of the number of rightmost digits of the Subscriber-Number that are different from the set of phone numbers of the ports used in this multilink connection. (For example, if the first port of a multilink bundle has a phone number of 123456789, and an implementation wanted its peer to call a port with a phone number of 123456888, the Unique-Digits octet would be 3.) If the Phone-Number-Sub-Address Sub-Option is present, the Unique-Digits Sub-Option MUST NOT include any of the Sub Address digits in its count of different rightmost digits.

This field is required.

Subscriber-Number

This field is the phone number of the port that should be called by the peer. Any digits that precede the rightmost unique digits of the Subscriber-Number are provided for informational purposes only, and do not need to be included in this field. This field is an ASCII string and MUST contain only ASCII characters indicating valid phone number digits. This field is required.

Phone-Number-Sub-Address

This field is the sub address of the port to be called by the peer. This sub-option SHOULD only be used for an ISDN call. This field is an ASCII string and only contains valid phone number digits. This field is optional.

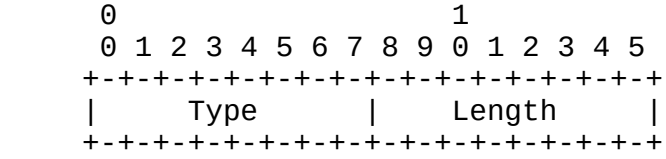
6.3. No-Phone-Number-Needed

Description

The No-Phone-Number option indicates that the calling implementation is already configured with the phone number of its multilink peer and the answering implementation MUST NOT include the Phone Number option in the response. This may be for security reasons, for configuration reasons, or for any other reason.

This option MAY be used in a Call-Request packet.

A summary of the No-Phone-Number BAP Option format is shown below. The fields are transmitted from left to right.



Type

03 for No-Phone-Number.

Length

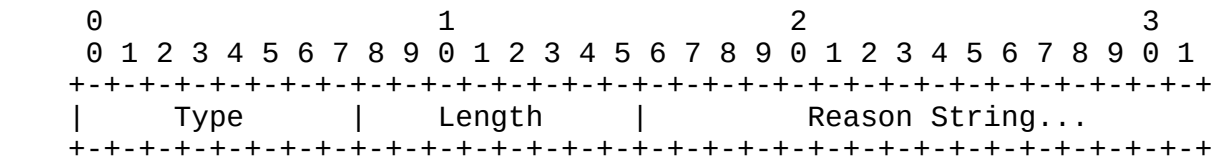
2

6.4. Reason

Description

This option is used to indicate a reason for the Request or Response. It is meant to be used for informational purposes only. This option MAY be used in any BAP packet.

A summary of the Reason BAP Option format is shown below. The fields are transmitted from left to right.



Type

04 for Reason.

Length

The Length field is one octet, and indicates the length of this BAP Option including the Type, Length and Reason String fields.

Reason String

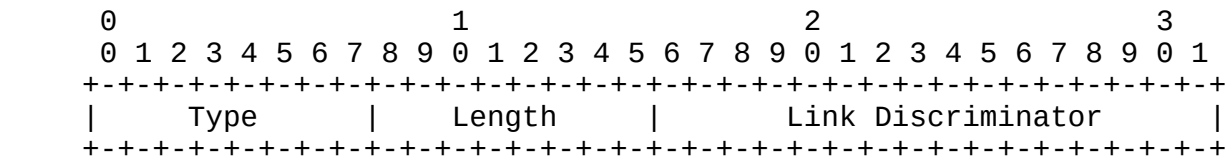
This is an ASCII string. The content of the field is implementation dependent. An implementation MAY ignore the Reason String field.

6.5. Link-Discriminator

Description

The Link-Discriminator option MUST be used in a Link-Drop-Query-Request datagram. This option is used to inform the receiver of a Link-Drop-Query-Request of which link will be dropped.

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



Type

05 for Link-Discriminator

Length

4

Link Discriminator

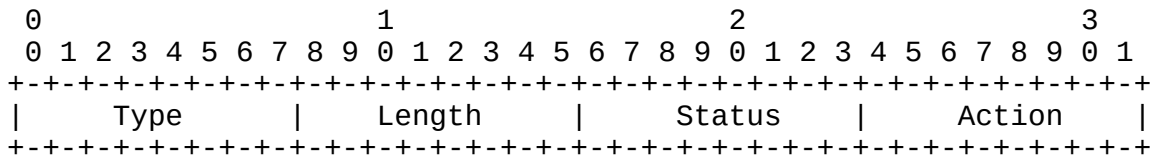
The Link Discriminator field is 2 octets in length. It contains the Link Discriminator that was contained in the LCP Link-Discriminator Configuration Option sent by the receiver of the packet containing the Link Discriminator.

6.6. Call-Status

Description

The Call-Status option MUST be used in a Call-Status-Indication datagram. This option is used to inform the receiver of the Call-Status-Indication datagram of the status of the completed call attempt, as well as a possible action that will be taken (if the call failed).

A summary of the Call-Status BAP Option format is shown below. The fields are transmitted from left to right.



Type

06 for Call-Status.

Length

4

Status

The Status field is 1 octet in length. If the call was successful, the value MUST be set to 0. A non-zero value indicates a call failure. A value of 255 indicates a non-specific failure, and a more specific call status MAY be indicated by using the same number as the Q.931 cause value (i.e., 1 is unassigned number, 17 is user busy, etc.)

Action

The Action octet indicates what action the calling implementation is taking after a failed call. If the call was sucessful, the Action octet MUST be set to 0.

The Action octet can have the following values:

0 - No retry
1 - Retry

Appendix

List of BAP datagrams and associated fields.

datagram	mandatory fields	allowed options
Call-Request	Link-Type	No-Phone-Number
Call-Response		Phone-Delta
		Link-Type
Callback-Request	Link-Type	
	Phone-Delta	
Callback-Response		Link-Type
Link-Drop-Query-Request	Link-Discriminator	
Link-Drop-Query-Response		
Call-Status-Indication	Call-Status	Phone-Delta
Call-Status-Response		

The Reason option is allowed to be included with any BAP datagram.

History of BACP

The first version of BACP was written by Craig Richards of Shiva Corporation. This version was enhanced and improved by the MPCP Working Group, a collaborative effort of 3Com, Ascend, Bay Networks, Cisco, Microsoft, Shiva, US Robotics and Xylogics.

Acknowledgements

Kevin Smith of Ascend for his contributions based on his work on the MP+ Specification. Gerry Meyer and Robert Myhill of Shiva for their early comments and improvements. Andy Nicholson of Microsoft for his improvements to the bandwidth management scheme. Dana Blair and Andy Valencia of Cisco, Cheng Chen and Dan Brennan of 3Com for their good ideas as part of the MPCP Working Group. All of the members of the MPCP working group for their ability to work with their competitors with enthusiasm to produce a better protocol for the industry.

Security Considerations

Security issues are not discussed in this memo.

References

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

[2] Sklower, Lloyd, McGregor, Carr & Coradetti, "The PPP Multilink Protocol", RFC 1990, University of California, Berkeley, Lloyd Internetworking, Newbridge Networks Corporation, Sidewalk Software, August 1996.

Chair's Address

The working group can be contacted via the current chair:

Karl Fox
Ascend Communications
3518 Riverside Drive, Suite 101
Columbus, Ohio 43221

(614)451-1883

EMail: karl@ascend.com

Editors' Addresses

Craig Richards
Shiva Corporation
28 Crosby Drive
Bedford, MA 01730
VOICE +1 617 270 8419
FAX +1 617 270 8599

EMail: crich@us.shiva.com

Kevin Smith
Ascend Communications, Inc.
1275 Harbor Bay Parkway
Alameda, CA 94501
CA

EMail: kevin@ascend.com