Network Working Group
Internet Draft
Updates: 4204, 4207, 4209, 5818
Category: Standards Track

Dan Li
Huawei
D. Ceccarelli
Ericsson
Lou Berger
LabN

Expires: January 2013

July 30, 2012

Link Management Protocol Behavior Negotiation and Configuration Modifications

draft-ietf-ccamp-lmp-behavior-negotiation-06.txt

Abstract

The Link Management Protocol (LMP) is used to coordinate the properties, use, and faults of data links in Generalized Multiprotocol Label Switching (GMPLS) networks. This document defines an extension to LMP to negotiate capabilities and indicate support for LMP extensions. The defined extension is compatible with non-supporting implementations.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

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

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

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html

This Internet-Draft will expire on January 29, 2013.

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Table of Contents

1. Introduction

The Link Management Protocol (LMP) [RFC4204] has been successfully deployed in Generalized Multiprotocol Label Switching (GMPLS)-controlled networks.

New LMP behaviors and protocol extensions have been introduced in a number of IETF documents as set out later in this section. It is likely that future extensions will be made to support additional functions.

In a network, if one LMP node supports a new behavior or protocol extension but its adjacent node does not, it is beneficial to have a protocol mechanism to discover the capabilities of peer nodes so that the right protocol extensions can be selected and the correct features can be enabled. There are no such procedures defined in the base LMP specification [RFC4204]. [RFC4209] defined a specific mechanism to identify support for the functions defined in that document. This document defines an LMP extension to support the identification of supported LMP functions in a generic fashion, as well as how a node supporting these extensions would communicate with legacy nodes.

In [RFC4204], the basic behaviors have been defined around the use of the standard LMP messages, which include Config, Hello, Verify, Test, LinkSummary, and ChannelStatus. Per [RCF4204], these behaviors MUST be supported when LMP is implemented, and the message types from 1 to 20 have been assigned by IANA for these messages. Support for all functions required by [RFC4204] is assumed by this document.

In [RFC4207], the SONET/SDH technology-specific behavior and information for LMP is defined. The Trace behavior is added to LMP, and the message types from 21 to 31 were assigned by IANA for the messages that provide the TRACE function. The Trace function has been extended for the support of OTNs (Optical Transport Networks) in [LMP-TEST].

In [RFC4209], extensions to LMP are defined to allow it to be used between a peer node and an adjacent Optical Line System (OLS). The LMP object class type and sub-object class name have been extended to support DWDM behavior.

In [RFC5818], the data channel consistency check behavior is defined, and the message types from 32 to 34 have been assigned by IANA for messages that provide this behavior.

It is likely that future extensions to LMP for other functions or technologies will require the definition of further LMP messages.

This document describes an LMP extension, which is referred to as behavior negotiation, which enables nodes at the ends of a link to identify the LMP messages and functions supported by the adjacent node. The extension makes use of a new CONFIG object. The use of this new object does not preclude the use of existing or yet to be defined CONFIG object.

This document also modifies the format of messages that carry CONFIG object to allow for multiple objects. Multiple CONFIG objects allow behavior negotiation concurrent with existing usage of the CONFIG object, i.e., HelloConfig C-Type defined in [RFC4204] and LMP_WDM_CONFIG C-Type defined in [RFC4209]. This document modifies the ConfigAck message to include CONFIG objects so that acceptable parameters are explicitly identified. It also describes how a node which supports the extensions defined in this document interacts with a legacy LMP node.

2. LMP Message Modifications

LMP Config, ConfigNack and ConfigAck messages are modified by this document to allow for the inclusion of multiple CONFIG objects. The Config and ConfigNack messages were only defined to carry on CONFIG object in [RFC4204]. The ConfigAck message, which was defined without carrying any CONFIG objects in [RFC4204], is modified to enable explicit identification of negotiated configuration parameters. The inclusion of CONFIG objects in ConfigAck messages is triggered by the use of the BehaviorConfig object (defined below) in a received Config message.

2.1. Modified Message Formats

The format of the Config message as updated by this document is as follows:

```
<Config Message> ::= <Common Header> <LOCAL_CCID> <MESSAGE_ID>
                     <LOCAL_NODE_ID> <CONFIG> [ <CONFIG> ... ]
```

The format of the ConfigAck message as updated by this document is as follows:

```
<ConfigAck Message> ::= <Common Header> <LOCAL_CCID> <LOCAL_NODE_ID>
                        <REMOTE_CCID> <MESSAGE_ID_ACK>
                        <REMOTE_NODE_ID>[ <CONFIG> ... ]
```

The format of the ConfigNack message as updated by this document is as follows:

```
<ConfigNack Message> ::= <Common Header> <LOCAL_CCID>
                         <LOCAL_NODE_ID>  <REMOTE_CCID>
                         <MESSAGE_ID_ACK> <REMOTE_NODE_ID>
                         <CONFIG> [ <CONFIG> ... ]
```

2.2. Processing

Nodes which support the extensions defined in this document MAY include multiple CONFIG objects when sending a Config, ConfigAck and ConfigNack message. A maximum of a single object of any particular C-type SHALL be included. A node which receives a message with multiple CONFIG objects of the same C-type SHALL process the first object of a particular C-type and ignore any subsequent CONFIG objects of the same C-type. Unless specified as part of the CONFIG object definition, ordering of CONFIG objects is not significant.

Nodes which support the extensions defined in this document MUST include a BehaviorConfig type object when sending a Config message to a neighbor whose support for the extensions is either known or unknown. (But not when the neighbor is known to not support the extensions.) Inclusion of other CONFIG objects in a Config message is at the discretion of the message sender, and is based on the rules defined by as part of CONFIG object definition. Nodes MAY include, HelloConfig, LMP_WDM_CONFIG, BehaviorConfig object types in a single message.

Inclusion of multiple CONFIG objects in a ConfigNack message is based on the processing of a received Config message. Per [RFC4204] "Parameters where agreement was reached MUST NOT be included in the ConfigNack Message." As such, a ConfigNack message MUST NOT include CONFIG objects which are acceptable and MUST include any CONFIG objects which are not acceptable. When a CONFIG object is included in a ConfigNack message, per [RFC4204], the object is to include "acceptable alternate values for negotiable parameters".

When sending a ConfigAck message, nodes supporting the extensions defined in this document MUST include all CONFIG objects received in the corresponding Config message when that message includes a CONFIG object of type BehaviorConfig.

3. LMP Behavior Negotiation

The Config message is used in the control channel negotiation phase of LMP [RFC4204]. The LMP behavior negotiation procedure is defined in this document as an addition to this phase.

The Config message is defined in Section 12.3.1 of [RFC4204] and carries the CONFIG object (class name 6) as defined in Section 13.6 of [RFC4204].

Two class types have been defined:

- C-Type = 1, HelloConfig, defined in [RFC4204]

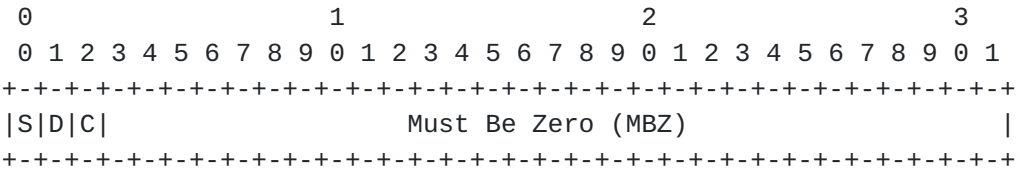
- C-Type = 2, LMP_WDM_CONFIG, defined in [RFC4209]

This document defines a third C-Type to report and negotiate LMP mechanisms and behaviors. Its usage indicates support for the extensions defined in this document.

3.1. BehaviorConfig C-Type Format

Class = 6

- C-Type = (To be assigned by IANA), BehaviorConfig



Flags:

S: 1 bit

This bit indicates support for the Trace behavior of SONET/SDH technology-specific defined in [RFC4207].

D: 1 bit

This bit indicates support for the DWDM behavior defined in [RFC4209].

C: 1 bit

This bit indicates support for the data channel consistency check behavior defined in [RFC5818].

Must Be Zero (MBZ): Variable length

The remaining bits in the flags field MUST be set to zero (0). The number of bits present is based on the Length field of the LMP object header and MUST include enough bits so the Length field MUST be at least 8, and MUST be a multiple of 4.

Other bits may be defined in future documents, in which case the number of MBZ bits field is expected to change.

3.2. Processing

The inclusion of a BehaviorConfig type object in a message is discussed above in Section 2.2.

When sending a BehaviorConfig type object, the N-bit (negotiable) in the LMP object header MUST be set (N=1) in the LMP object header.

When sending a BehaviorConfig type object in Config and ConfigNack messages, the flags field SHOULD be set based on the supported capabilities of the sending node. When sending a ConfigAck message, the flags field MUST be set to the value received in the corresponding Config message.

When receiving a BehaviorConfig type object, the node compares the flags field against its capacities. Any bit set in the MBZ portion of the flags field MUST be interpreted as unacceptable. Processing related to unacceptable values in CONFIG objects is defined in [RFC4204] and is not modified by this document.

4. Backward Compatibility

The required use of the BehaviorConfig type CONFIG object enables nodes which support the extensions defined in this document to explicitly identify when a neighboring node does not. When a non-supporting node receives a Config message with the BehaviorConfig type CONFIG object or multiple CONFIG objects its behavior is likely to be one of the following behaviors:

a) Reject the Config message because of the unknown BehaviorConfig object type and send a ConfigNack message which includes the unsupported C-type.

b) Reject the message because of multiple CONFIG objects and send a ConfigNack message which includes all but one of the CONFIG objects.

c) Silently ignore the one or more of the CONFIG object, and respond with a ConfigAck message that does not include any CONFIG objects.

d) Treat the message as malformed, and discard it without any response.

Behaviors (a) and (b) result in ConfigNack messages with a BehaviorConfig type object whose contents are identical to what was sent in the Config message. Behavior (c) results in a ConfigAck message without a BehaviorConfig type CONFIG object. In each of these cases, the node SHOULD explicitly identify that the LMP neighbor does not support the extensions defined in this document.

Behavior (d) results in no response at all. When the node reaches the, [RFC4204] defined, "retry limit", the node SHOULD infer that the LMP neighbor does not support the extensions defined in this document.

Once a node identifies a neighbor as not supporting the extensions defined in this document, the node SHOULD follow previously defined Config message usage.

5. Security Considerations

[RFC4204] describes how LMP messages between peers can be secured, and these measures are equally applicable to messages carrying the new CONFIG object defined in this document.

The procedures described in this document do not of itself constitute a security risk since they do not cause any change in network state. It would be possible, if the messages were intercepted or spoofed to cause bogus alerts in the management plane, or to cause LMP peers to consider that they could or could not operate protocol extensions, and so the use of the LMP security measures are RECOMMENDED.

Note, however, that [RFC4204] refers to [RFC2401], which has been replaced by [RFC4301]. Also, the reference to IKEv2 in [RFC4301] is out of date, and the current reference for IKEv2 is [RFC5996].

6. IANA Considerations

6.1. New LMP Class Type

IANA maintains the "Link Management Protocol (LMP)" registry which has a subregistry called "LMP Object Class name space and Class type (C-Type)".

IANA is requested to make an assignment from this registry as follows:

6 CONFIG [RFC4204]

CONFIG Object Class type name space:

C-Type	Description	Reference
3(suggested)	BehaviorConfig	[This.I-D]

6.2. New Capabilities Registry

IANA is requested to create a new subregistry of the "Link Management Protocol (LMP)" registry to track the Behaviour Configuration bits defined in Section 2 of this document. It is suggested that this registry be called "LMP Behaviour Configuration Flags".

Allocations from this registry are by Standards Action.

Bits in this registry are numbered from zero as the most significant bit (transmitted first). The number of bits that can be present is limited by the length field of the CONFIG object which gives rise to (255 x 32)-8 = 8152. IANA is strongly recommended to allocate new bits with the lowest available unused number.

The registry is initially populated as follows:

Bit Number	Bit Name	Meaning	Reference
0	S	SONET/SDH Test support	[This.ID]
1	D	DWDM support	[This.ID]
2	C	Data Channel consistency check support	[This.ID]

7. Contributors

Diego Caviglia
Ericsson
Via A. Negrone 1/A 16153
Genoa Italy
Phone: +39 010 600 3736
Email: diego.caviglia@ericsson.com

8. Acknowledgments

Thanks to Adrian Farrel and Richard Graveman for their useful comments.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2401] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005

[RFC5996] C. Kaufman, P. Hoffman, Y. Nir, P. Eronen, "Internet Key Exchange Protocol: IKEv2", RFC 5996, September 2010.

[RFC4204] J. Lang, Ed., "Link Management Protocol (LMP)", RFC 4204, October 2005.

[RFC4207] J. Lang, Ed., "Synchronous Optical Network (SONET)/ Synchronous Digital Hierarchy (SDH) Encoding for Link Management Protocol (LMP) Test Messages", RFC 4207, October 2005.

[RFC4209] A. Fredette, Ed., "Link Management Protocol (LMP) for Dense Wavelength Division Multiplexing (DWDM) Optical Line Systems", RFC 4209, October 2005.

[RFC5818] D. Li, Ed., "Data Channel Status Confirmation Extensions for the Link Management Protocol", RFC 5818, April 2010.

9.2. Informative References

[LMP TEST] D. Ceccarelli, Ed., "Link Management Protocol (LMP) Test Messages Extensions for Evolutive Optical Transport Networks (OTN)" draft-ceccarelli-ccamp-gmpls-g709-lmp-test-04.txt, July, 2012.

10. Authors' Addresses

Dan Li
Huawei Technologies
F3-5-B R&D Center, Huawei Industrial Base,
Shenzhen 518129 China
Phone: +86 755-289-70230
Email: danli@huawei.com

Daniele Ceccarelli
Ericsson
Via A. Negrone 1/A
Genova - Sestri Ponente
Italy
Email: daniele.ceccarelli@ericsson.com

Lou Berger
LabN Consulting, L.L.C.
Email: lberger@labn.net