Mobile Ad Hoc Networking (MANET)
Internet-Draft
Intended status: Experimental
Expires: August 18, 2014

M. Gerla
University of California, Los Angeles
S. Oh
UtopiaCompression Corporation
A. Colin de Verdiere
University of California, Los Angeles

February 14, 2014

ODMRP-ASYM
draft-gerla-manet-odmrp-asym-00

Abstract

This document describes ODMRP-ASYM, an extension for the On Demand Multicast Routing Protocol (ODMRP) aimed at taking advantage of unidirectional links rather than avoiding them.

Table of Contents

1. Introduction

Due to the nature of the wireless media used, MANETs typically exhibit a non-negligible proportion of asymmetric, or even unidirectional links, even more so when routers themselves make use of disparate transmission power (such as when using satellite communications). Most routing protocols make sure to avoid these links and use only the fully connected graph formed by the bidirectional links of the network, while taking advantage of these links can provide significant performance improvement, and even in some case allow data flows to reach weakly connected parts of the network.

This document specifies ODMRP-ASYM, an extension for the ODMRP protocol [ODMRP] that allows routers to make use of unidirectional links instead of avoiding them. It does so by enabling ODMRP-ASYM routers to discover alternative path to forward Join Reply messages to the multicast source, building the multicast mesh along the way.

2. Terminology and Notations

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

This document uses the terminology and notation defined in [ODMRP]. Additionally, it uses the terminology defined in Section 2.1 and the notational conventions defined in Section 2.2.

2.1. Terminology

This document defines and uses the following terminolgy:

ODMRP-ASYM Router - A router that implements this specification, in addition to implementing the original ODMRP specification, as described in [ODMRP]. An ODMRP-ASYM Router is equipped with at least one interface.

HC2SRC - Abbreviation for Hop-Count to Source; for a given ODMRP-ASYM Router, this refers to the number of hops separating this router from a multicast source, as determined by the Hop Count field of the last valid JQ message received from this source.

Loop - The Loop is the basic construct used by this specification to discover an alternate reverse route to a Multicast Source. A Loop consists in a chain of adjacent (i.e., each Router in the chain can receive messages from the previous one) ODMRP-ASYM Routers. It is closed if the first ODMRP-ASYM Router is the same as the last one in the chain; otherwise, it is open. Furthermore, a Loop has an Originator (the first Router in the chain) and a Destination.

Loop Originator - Refers to the ODMRP-ASYM Router, which has started the Loop Discovery process. It is the first router in the chain, and the last one when the Loop is closed.

Loop Destination - A Loop is built in order to discover an alternate route through which Join Reply messages can be forwarded towards a Multicast Source. This Mulicast Source is the Destination of the Loop.

Loop Summit - Is the closest (in terms of hop count) router to the Loop Destination among all the routers in a given Loop.

2.2. Notational conventions

This document defines and uses the following notational conventions:

tail - Loop Discovery and Loop Marking messages both carry an AddressList field. "tail" is defined such that tail(AddressList) is a list created by removing the first element in AddressList.

head - Conversely, head(AddressList) refers to the first element in AddressList.

length - length(AddressList) is the number of addresses in AddressList.

3. Applicability Statement

This protocol:

- o Is an extension of the ODMRP [ODMRP] protocol.
- o Enables ODMRP-ASYM routers to make use of unidirectional links for forwarding multicast data packets.
- o Discovers alternative routes on-demand, meaning that it does not impose any extra overhead on ODMRP when there are no unidirectional link present.

4. Protocol Overview and Functioning

The aim of this extension is for ODMRP-ASYM Routers to be able to make use of undirectional links between routers instead of blacklisting them. The objective is to use these links to transmit multicast packets, which would otherwise need to transit through longer paths, or would not reach loosely connected components, i.e., parts of the network that are only reachable through unidirectional links. This mechanism was originally described in [ODMRP_ASYM].

This objective is fulfilled by the following process:

1. Upon detection by an ODMRP-ASYM Router (the Loop Originator) that a Join Reply (towards one Multicast Source) has not been successfully delivered to an upstream router (through the acknowledgement mechanism described in [ODMRP], the downstream router triggers a Loop Discovery procedure as follows:

    1. The Loop Originator generates and broadcasts a Loop Discovery message advertising the Multicast Source as its destination.

    2. The Loop Discovery message is retransmitted by intermediate routers. Each router updates the LD message so as to reflect the list of ODMRP-ASYM routers this message has transited through, as well as the Loop Summit, i.e. the intermediate router, which is closest (in terms of number of hops) to the Multicast source.

    3. Upon reception of a Loop Discovery message it generated, the Loop Originator verifies that it advertises a valid closed loop, i.e. that at least one router it transited through (the Loop Summit) is closer to the Multicast Source than itself. If it is the case, it starts thte Loop Marking procedure.

2. The Loop Marking procedure is as follows:

    1. The Loop Originator generates a Loop Marking message, advertising the list of routers that the LD message went through (the Loop), as well as the Loop Summit. The Loop Marking message is source-routed through the Loop.

    2. Each ODMRP-ASYM router, receiving the Loop Marking message, proceeds as follows:

        + If that Router's position in the Loop, as recorded by the Address List (see Section 6), is before the Loop Summit, the router forwards the LM message along the Loop.

+ If the router is the Loop Summit, it restarts the Join Reply procedure by generating a Join Reply message and forwarding it towards the Multicast Source, then forwards the LM message along the Loop.

+ Otherwise, i.e., if the router's position in the Loop is after the Loop Summit in the Loop, it adds or updates a Forwarding Tuple to its Forwarding Table, as described in [ODMRP].

5. Parameters and Constants

In addition to those defined in [ODMRP], this specification uses the parameters and consants defined in this section.

5.1. Router Parameters

This specification defines the following router parameters:

PENDING_LOOP_TIMEOUT is the minimum time a Loop tuple SHOULD be kept in the Pending Loop set after it was last refreshed.

DEFAULT_LD_HOP_LIMIT is the default value for the LD.HOPLIMIT field used by ODMRP-ASYM Routers generating an LD message.

5.2. Constants

This specification defines the following constants:

NO_SUMMIT - is a value, carried by the LoopSummit field of Loop Discovery and Loop Marking messages, meaning that the corresponding Loop currently has no Loop Summit or that the Loop Summit is not encoded in this message. For example, a Loop Marking message that has already transited through the Loop Summit does not carry its address anymore.

6. Packets and Messages

This section describes the protocol messages generated and processed by ODMRP-ASYM, according to the terminology defined in Section 2. In particular, this section describes the fields contained in each message. The specifics of the encoding are separated from this section. In particular, the encoding of these messages using [RFC5444] is described in Appendix B.

6.1. Loop Discovery

A Loop Discovery (LD) message is generated andn processed in order to discover and build a loop. It has the following fields:

LD.AddressLength is a 4 bit unsigned integer field, encoding the length of the addresses carried by this message as follows:

LD.AddressLength := the length of an address in octets - 1

LD.MulticastGroupAddress is an unsigned integer field, of length LD.AddressLength + 1 octets, encoding the address of the Multicast Group, to which this Join Reply is addressed

LD.Destination is an unsigned integer field, of length LD.AddressLength + 1 octets, encoding the address of the Loop Destination

LD.AddressList is an ordered list of the addresses of the routers that this message has traversed, including the router that has generated the message. This means that the Loop Originator is identified by head(LD.AddressList).

LD.LoopSummit is an unsigned integer field, representing the index in the LD.AddressList field of the address of the Loop Summit for the current loop. The addresses in LD.AddressList are indexed from 1 to length(AddressList). A value of NO_LOOPSUMMIT means that the loop described by this message does not have a Loop Summit.

LD.MinHC is an unsigned integer field, representing the HC2SRC value of the Loop Summit. This field MUST be set to 0 and ignored on reception if LD.LoopSummit = NO_LOOPSUMMIT.

LD.HOPLIMIT is an unsigned integer field, representing the maximum number of hops this message can traverse.

LD.HOPCOUNT is an unsigned integer field, representing the number of hops this message has already traversed.

6.2. Loop Marking

LM.AddressLength is a 4 bit unsigned integer field, encoding the length of the addresses carried by this message as follows:

LM.AddressLength := the length of an address in octets - 1

LM.MulticastGroupAddress is an unsigned integer field, of length LM.AddressLength + 1 octets, encoding the address of the Multicast Group, to which this Join Reply is addressed.

LM.SourceAddress is an unsigned integer field, of length LM.AddressLength + 1 octets, encoding the address of the Multicast Source, towards which the Loop Summit will transmit a Join Reply.

LM.AddressList is an ordered list of the addresses of the routers this message has to transit through. The Loop Marking message is effectively source-routed through these routers.

LM.LoopSummit is an unsigned integer field, representing the index in the LD.AddressList field of the address of the Loop Summit for the current loop.

LM.SourceSequenceNumber is a 16 bit unsigned integer field, corresponding to the sequence number of the original Join Query message, that is, the Join Query message which was not successfully delivered and triggered the Loop Discovery process.

7. Information Bases

In additions to the information bases described in [ODMRP], each ODMRP-ASYM Router maintains a Distance Set and a Pending Loop Set, as described in the following sections. These information sets are given so as to facilitate description of message generation, forwarding and processing rules. An implementation may chose any representation or structure for when maintaining this information.

7.1. Distance Set

The Distance set contains distance tuples, recording the distance in hop counts to (active) Multicast sources, as recorded by received Join Query messages, and containing the following fields:

(D_source, D_hop_count, D_seq_num, D_exp_time)

Where:

D_source - is the address of the Multicast Source.

D_hop_count - is the distance, in hops, to the Multicast Source, as recorded by the most recent Join Query received that was originated by this source.

D_seq_num - is the sequence number of the Join Query message that updated this tuple.

D_exp_time - is the time at which the tuple MUST be considered expired and thus MUST NOT be taken into consideration by the operations of this protocol extension.

7.2. Pending Loops Set

The Pending Loops Set contains pending loop tuples, each recording information about an open Loop that this Router is part of, either because it has initiated the corresponding Loop Discovery process (i.e., this router is the Loop Originator) or because it has received and forwarded a corresponding Loop Discovery message. It contains the following fields:

(L_originator, L_destination, L_exp_time)

Where:

L_originator - is an address of the Loop Originator, as recorded by the corresponding Loop Discovery message

L_destination - is an address of the Loop Destination, as recorded by the corresponding Loop Discovery message

L_exp_time - is the time at which the tuple MUST be considered expired and thus MUST NOT be taken into consideration by the operationsn of this protocol extension.

8. Protocol Details

This protocol generates, processes and forwards Loop Discovery and Loop Marking messages, according to the following sections. This section makes use of the terminology defined in Section 9 of [ODMRP], as well as the following additional notation:

Unsuccessful Join Reply (UJR) Refers to the Join Reply, which unsuccessful delivery to the upstream router triggered the generation of a Loop Discovery. The abbreviation UJR is used to refer to this Join Reply's fields.

8.1. Loop Discovery

8.1.1. Invalid Loop Discoveries

A Loop Discovery Message, received by an ODMRP-ASYM Router, is invalid and MUST be discarded without further processing, and in particular MUST NOT be considered for forwarding, if:

o The address length carried by the received Loop Discovery Message (see Section 6) differs from the length of the addresses of the ODMRP-ASYM Router.

o In the received Loop Discovery Message, LD.HOPCOUNT > LD.HOPLIMIT, or LD.HOPCOUNT == LD.HOPLIMIT and LD.Originator is not an address of this Router.

o LD.Originator from the received Loop Discovery Message is an address of this Router, and there isn't any Pending Loop tuple in the Pending Loops set, such as:

   * L_originator is an address of this Router.

   * L_destination = LD.Destination.

8.1.2. Loop Discovery Generation

A Loop Discovery message SHOULD be generated upon detection that a Join Reply (the Unsuccessful Join Reply) was unable to be delivered to the upstream router . A Loop Discovery message is generated according to Section 6, with the following fields:

o LD.AddressLength := UJR.AddressLength.

o LD.MulticastGroupAddress := UJR.MulticastGroupAddress.

o LD.AddressList set to a list, containing as its only element an address of this ODMRP-ASYM Router.

o LD.LoopSummit := NO_SUMMIT.

o LD.MinHC := D_hop_count.

8.1.3. Loop Discovery Processing

Upon receiving a valid Loop Discovery message, an ODMRP-ASYM Router proceeds as follows:

1. If head(LD.AddressList) is an address of this Router (i.e. the Loop Discovery message advertises a closed Loop originated by this router), then:

    1. If there exists a Distance tuple (henceforth "corresponding Distance tuple") in the Distance set, such as:

        + D_source = LD.Destination

        + D_hop_count < LD.MinHC

       Then this Loop Discovery message advertises a valid closed Loop. Otherwise, the advertised loop is invalid. The Loop Discovery message MUST be discarded and MUST NOT be processed further.

    2. Generate a new Loop Marking message according to Section 8.2.2

    3. Set the corresponding Pending Loop tuple as expired, by setting P_exp_time to current_time - 1. The Loop Discovery message is not processed further, and in particular MUST NOT be considered for forwarding.

    * L_originator == head(LD.AddressList).

    * L_destination == LD.Destination.

2. The Loop Discovery message is then considered for forwarding, according to Section 8.1.4.

8.1.4. Loop Discovery Forwarding

A Loop Discovery message, considered for forwarding, MUST be updated as follows, prior to being transmitted:

1. Append this Router's address to LD.AddressList.

2. Find the Distance Tuple defined by:

    * D_source == LD.Destination.

3. If such a tuple exists, and if D_hop_count < LD.MinHC, then update the Loop Discovery message as follows:

    1. LD.MinHC := D_hop_count.

2. LD.LoopSummit := length(LD.AddressList).

4. LD.HOPCOUNT := LD.HOPCOUNT + 1.

8.1.5. Loop Discovery Transmission

A Loop Discovery message MUST be broadcast on all participating ODMRP interfaces.

8.2. Loop Marking

8.2.1. Invalid Loop Marking messages

A Loop Marking Message, received by an ODMRP-ASYM Router, is invalid and MUST be discarded without further processing, and in particular MUST NOT be considered for forwarding, if:

- o The address length carried by the Loop Marking Message (see Section 6) differs from the length of the addresses of the ODMRP-ASYM Router.

- o head(LM.AddressList) of the received Loop Marking Message is not an address of this Router.

8.2.2. Loop Marking Generation

A Loop Marking Message MUST be generated by an ODMRP-ASYM Router upon receiving a Loop Discovery message advertising a valid closed Loop originated by this router, as described in Section 8.1.3. A Loop Marking message MUST be generated according to Section 6 with the following fields, using LD as a shortcut for the corresponding Loop Discovery message:

- o LM.AddressLength := LD.AddressLength.

- o LM.MulticastGroupAddress := LD.MulticastGroupAddress.

- o LM.SourceAddress := LD.Destination.

- o LM.AddressList := LD.AddressList.

- o LM.LoopSummit := LD.LoopSummit.

- o LM.SequenceNumber := D_seq_num (from the corresponding Distance tuple)

8.2.3. Loop Marking Processing

Upon receiving a valid Loop Marking message, an ODMRP-ASYM Router proceeds as follows:

1. If LM.LoopSummit == 1 (i.e. this Router is the Loop Summit for this Loop):

    1. Find the Routing Tuple (corresponding Routing Tuple) in the Routing Set such as:

        + R_source = LM.SourceAddress.

    2. If no such tuple exists, the Loop Marking message is invalid and MUST be discarded without any further processing.

    3. Create a new Join Reply with the following fields:

        + JR.AddressLength := LM.AddressLength.

        + JR.MulticastGroupAddress := LM.MulticastGroupAddress.

        + JR.AckRequired := 0.

        + JR.SourceAddress := LM.SourceAddress.

        + JR.SequenceNumber := R_seq_num.

        + JR.NextHopAddress := R_next_hop.

        The new Join Reply is then considered for forwarding, according to [ODMRP].

2. If LM.LoopSummit == 1 or LM.LoopSummit == NO_SUMMIT, then:

    1. Find the Forwarding Tuple (matching Forwarding Tuple) in the Forwarding Table such as:

        + F_multicast_group == LM.MulticastGroupAddress.

        + F_source == LM.SourceAddress.

    2. If no matching Forwarding Tuple is found, create a Forwarding Tuple with:

        + F_multicast_group := LM.MulticastGroupAddress.

+ F_source := LM.SourceAddress.

+ F_seq_num := -1.

+ F_exp_time := FG_TIMEOUT.

3. The matching Forwarding Tuple, existing or new, is compared with the matching Routing Tuple: if F_seq_num <= LM.SourceSequenceNumber:

   1. Set F_seq_num to LM.SourceSequenceNumber.

   2. Set F_exp_time to FG_TIMEOUT.

3. The Loop Marking message is then considered for forwarding, according to Section 8.2.4.

8.2.4. Loop Marking Message Forwarding

A Loop Marking message, considered for forwarding, MUST be updated as follows:

- o Remove the first address of LM.AddressList, i.e. set LM.AddressList to tail(LM.AddressList). If LM.AddressList becomes empty, the Loop Marking message MUST be discarded and MUST NOT be processed further.

- o If LM.LoopSummit != NO_SUMMIT, then update LM.LoopSummit such that LM.LoopSummit := LM.LoopSummit - 1.

The Loop Marking message is then transmitted in unicast to the ODMRP-ASYM Router, identified by head(LM.AddressList).

9. Security Considerations

This document does currently not specify any security considerations.

10. IANA considerations

The IANA is requested to assign two new message types for Loop Discovery and Loop Marking messages, as well as one Message-Type-Specific TLV Type and one Message-Type-Specific Address block TLV registry for each of those message types, as specified below.

10.1. Loop Discovery registries

IANA is requested to create a registry for Message-Type-specific Message TLVs for LD messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 1.

Type	Description	Allocation policy
128	LOOPSUMMIT	
129	MINHC	
130-223	Unassigned	Expert review

Table 1: Loop Discovery Message-Type-Specific TLV types

Allocation of the LOOPSUMMIT and MINHC TLVs from the LD Message-Type-specific Message TLV types in Table 1 will create two new Type Extension registries, with assignments specified in Table 2 and Table 3.

Type Extension	Description	Allocation policy
0-255	Unassigned	Expert review

Table 2: LD Message TLV Type assignment: LOOPSUMMIT

Type Extension	Description	Allocation policy
0-255	Unassigned	Expert review

Table 3: LD Message TLV Type assignment: MINHC

IANA is requested to create a registry for Message-Type-specific address block TLVs for LD messages, in accordance with section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 4.

Type	Description	Allocation policy
128	ADDR-TYPE	
129-223	Unassigned	Expert review

Table 4: Loop Discovery Message-Type-specific address block TLV types

Allocation of the ADDR-TYPE TLV from the LD Message-Type-specific TLV Address block TLV Types will create a new Type extension registry, with assignments specified in Table 5.

Type Extension	Description	Allocation policy
0	MULTICAST-GROUP-ADDRESS	
1	DESTINATION-ADDRESS	
2	ADDRESS-LIST	
3-255		Expert review

Table 5: LD Message Address block TLV Type assignments: ADDR-TYPE

10.2. Loop Marking registries

IANA is requested to create a registry for Message-Type-specific Message TLVs for LM messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 6.

Type	Description	Allocation policy
128	LOOPSUMMIT	
129-223	Unassigned	Expert review

Table 6: Loop Discovery Message-Type-Specific TLV types

Allocation of the LOOPSUMMIT TLV from the LM Message-Type-specific TLV Types will create a new Type extension registr, with assignments specified in Table 7.

Type extension	Description	Allocation policy
0-255	Unassigned	Expert review

Table 7: LM Message Types assignments: LOOPSUMMIT

IANA is requested to create a registry for Message-Type-specific address block TLVs for LM messages, in accordance with section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 8.

Type	Description	Allocation policy
128	ADDR-TYPE	
129-223	Unassigned	Expert review

Table 8: Loop Marking Message-Type-specific address block TLV types

Allocation of the ADDR-TYPE TLV from the LM Message-Type-specific TLV Address block TLV Types will create a new Type extension registry, with assignments specified in Table 9.

Type Extension	Description	Allocation policy
0	MULTICAST-GROUP-ADDRESS	
1	MULTICAST-SOURCE-ADDRESS	
2	ADDRESS-LIST	
3-255		Expert review

Table 9: LD Message Address block TLV Type assignments: ADDR-TYPE

11. Acknowledgements

The authors would like to thank Yeng-Zhong Lee, Joon-Sang Park, and Yunjung Yi for their work on the original protocol, as published in [ODMRP_ASYM].

12. References

12.1. Normative References

[ODMRP] Yi, Y., Lee, S., Su, W., Gerla, M., and A. Colin de Verdiere, "The On Demand Multicast Routing Protocol", draft-gerla-manet-odmrp (work in progress).

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

[RFC5444] Clausen, T., Dearlove, C., Dean, J., and C. Adjih, "Generalized MANET Packet/Message Format", RFC 5444, February 2009.

12.2. Informative References

[ODMRP_ASYM] Gerla, M., Lee, Y., Park, J., and Y. Yi, "On Demand Multicast Routing With Unidirectional Links".

Appendix A. Illustrations

This section shows examples of ODMRP-ASYM control messages encoded using . specifies that a packet is formed by a packet header, an optional TLV block and zero or more messages. ODMRP-ASYM does not use any packet TLV, and the minimal packet header required by ODMRP-ASYM does not differ from the one required by ODMRP (see [ODMRP], Appendix B).

A.1. Loop Discovery message

LD messages are instances of [RFC5444] messages. This section illustrates an example of LD message.

The LD message's header has the bits 1 (mhashoplimit) and 2 (mhashopcount) set, indicating that the hop count and hop limit fields are present, but not the originator address and sequence number. Its address length field is set to 3, indicating that the addresses carried by this message are 3 + 1 = 4 octets long. The overall message length is 56 octets.

Both the LD.LOOPSUMMIT and LD.MINHC field are required, and encoded by the two TLVs with corresponding types that the message carries, with respective values LS and MHC.

The message has 3 address blocks. The first two encode LD.MulticastGroupAddress and LD.Destination respectively, and have a flag octet (ABF) value of 0, hence with no Tail or Head section, and a Mid section of length 4 octets. The third address block encodes LD.AddressList, and contains 4 addresses, sharing a 3 octets prefix (Head), as specified by the Field octet value of 128 (bit 0 set). These address blocks have each one associated Message-Type-specific Address block TLV of type ADDR-TYPE and type extension 0 (MULTICAST-GROUP-ADDRESS), 1 (DESTINATION-ADDRESS) and 2 (ADDRESS-LIST) respectively.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Loop Discovery |0 1 1 0| MAL=3 |       Message Size = 56       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Hop limit   |   Hop count   |         TLVs Length = 8       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   LOOPSUMMIT  |0 0 0 1 0 0 0 0|   Length = 1  |   Value = LS  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     MINHC     |0 0 0 1 0 0 0 0|   Length = 1  |  Value = MHC  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Num Addrs = 1 |    ABF = 0    |         Multicast Group    ...|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|...         Address            |   Addr-TLV blk Length = 3     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   ADDR-TYPE   |1 0 0 0 0 0 0 0|      0        |  Num Addrs = 1|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    ABF = 0    |                      Destination           ...|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|... Address    |   Addr-TLV blk Length = 3     |    ADDR-TYPE  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|1 0 0 0 0 0 0 0|       1       | Num Addrs = 3 |1 0 0 0 0 0 0 0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Head length = 3|                     Head                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Addr1     |     Addr2     |     Addr3     |     Addr4     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Addr-TLV blk length = 3    |   ADDR-TYPE   |1 0 0 0 0 0 0 0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      2        |
+-+-+-+-+-+-+-+-+
```

Figure 1

A.2. Loop Marking message

LM messages are instances of [RFC5444] messages. This section illustrates an example of LM message.

The LM message's header has only the bit 3 (mhasseqnum) set, indicating that the sequence number field is present, but that the originator address, hop count and hop limit fields are omitted. Its address length field is set to 3, indicating that the addresses carried by this message are 3 + 1 = 4 octets long. The overall message length is 52 octets.

The message contains one Message-Type-specific TLV, of Type LOOPSUMMIT and value LS = LM.LoopSummit.

The message has 3 address blocks. The first two encode LM.MulticastGroupAddress and LM.Destination respectively, and have a flag octet (ABF) value of 0, hence with no Tail or Head section, and a Mid section of length 4 octets. The third address block encodes LM.AddressList, and contains 4 addresses, sharing a 3 octets prefix (Head), as specified by the Field octet value of 128 (bit 0 set). These address blocks have each one associated Message-Type-specific Address block TLV of type ADDR-TYPE and type extension 0 (MULTICAST-GROUP-ADDRESS), 1 (MULTICAST-SOURCE-ADDRESS) and 2 (ADDRESS-LIST) respectively.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |  Loop Marking |0 0 0 1| MAL=3 |       Message Size = 52       |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |     Source Sequence Number    |         TLVs Length = 4       |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |   LOOPSUMMIT  |0 0 0 1 0 0 0 0|   Length = 1  |   Value = LS  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    | Num Addrs = 1 |    ABF = 0    |        Multicast Group     ...|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |...         Address            |   Addr-TLV blk Length = 3     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |   ADDR-TYPE   |1 0 0 0 0 0 0 0|       0       |  Num Addrs = 1|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    ABF = 0    |                     Source                 ...|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |... Address    |   Addr-TLV blk Length = 3     |    ADDR-TYPE  |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |1 0 0 0 0 0 0 0|       1       | Num Addrs = 3 |1 0 0 0 0 0 0 0|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |Head length = 3|                    Head                       |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |     Addr1     |     Addr2     |     Addr3     |     Addr4     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    Addr-TLV blk length = 3    |   ADDR-TYPE   |1 0 0 0 0 0 0 0|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      2        |
    +-+-+-+-+-+-+-+-+
```

Figure 2

Appendix B. RFC5444 Encoding

This section describes the encoding of ODMRP_ASYM messages using [RFC5444].

B.1. Loop Discovery Encoding

This protocol defines the Loop Discovery message type. Hence, according to [RFC5444], all Loop Discovery messages are generated, processed and transmitted following this specification. Table 10 shows the mapping between the Loop Discovery elements described in Section 6.1 and their encoding. Each LD message MUST contain exactly one of each of these elements.

LD Element	RFC5444 Element	Considerations
LD.AddressLength	<msg-addr-length>	Encodes from 1 to 16 bytes addresses.
LD.MulticastGroupAdd r ess	Address in address block + TLV	Is encoded by way of an address block with associated message-type-specific TLV of type ADDR-TYPE and value MULTICAST-GROUP-ADDRE SS.
LD.Destination	Address in address block + TLV	Is encoded by way of an address block with associated message-type-specific TLV of type ADDR-TYPE and value DESTINATION-ADDRESS.
LD.AddressList	Addresses in address block + TLV	Is encoded by way of an address block with <num-addr> = length(LD.AddressList), containing the addresses in order. The address block ha san associated message-type-specifi cTLV of type ADDR-TYP Eand value ADDRESS-LIST.

LD.LoopSummit	LOOPSUMMIT Message-type-specific TLV	Is encoded by way of a message-type-specific TLV of type LOOPSUMMIT, with all the flags cleared except <thasvalue>, <length> = 1 and where <value> is the index of the Loop Summit in LD.AddressList. If LD.LoopSummit = 0, <thasvalue> is cleared, and the <length> and <value> fields are omitted.
LD.MinHC	MINHC Message-type-specific TLV	
LD.HOPLIMIT	<msg-hop-limit>	Is encoded by way of the <msg-hop-limit> field of the LD message header.
LD.HOPCOUNT	<msg-hop-count>	

Table 10: Loop Discovery Message Elements

B.2. Loop Marking Encoding

This protocol defines the Loop Marking message type. Hence, according to [RFC5444], all Loop Marking messages are generated, processed and transmitted following this specification. Table 11 shows the mapping between the Loop Marking elements described in Section 6.2 and their encoding. Each LD message MUST contain exactly one of each of these elements.

LM Element	RFC5444 Element	Considerations
LM.AddressLength	<msg-addr-length>	Encodes from 1 to 16 bytes addresses
LM.MulticastGroupAddr ess	Address in address block + TLV	Is encoded by way of an address block with associated message-type-specific TLV of type ADDR-TYPE and value MULTICAST-GROUP-ADDRESS.
LM.SourceAddress	Address in address block + TLV	Is encoded by way of an address block with associated message-type-specific TLV of type ADDR-TYPE and value SOURCE-ADDRESS.
LM.AddressList	Addresses in address block + TLV	Is encoded by way of an address block with <num-addr> = length(LM.AddressList), containing the addresses in order. The address block ha san associated message-type-specifi cTLV of type ADDR-TYP Eand value ADDRESS-LIST.

LM.LoopSummit	LOOPSUMMIT Message-type-specific TLV	Is encoded by way of a message-type-specific TLV of type LOOPSUMMIT, with all the flags cleared except <thasvalue>, <length> = 1 and where <value> is the index of the Loop Summit in LM.AddressList. If LM.LoopSummit = 0, <thasvalue> is cleared, and the <length> and <value> fields are omitted.
LM.SourceSequenceNumber	<msg-seq-num>	

Table 11: Loop Marking Message Elements

Authors' Addresses

Mario Gerla
University of California, Los Angeles
3732F Boelter Hall
Computer Science Department
University of California
Los Angeles, CA 90095-1596,
USA

Phone: +1 310 825-4367
Email: gerla@cs.ucla.edu

Soon Young Oh
UtopiaCompression Corporation

Email: soon@utopiacompression.com

Axel Colin de Verdiere
University of California, Los Angeles

Email: axel@axelcdv.com