Mobile Ad Hoc Networking (MANET)   M. Gerla
Internet-Draft   University of California, Los
Intended status: Experimental   Angeles
Expires: May 20, 2015   S. Oh
                                          UtopiaCompression Corporation
                                                   A. Colin de Verdiere
                                            University of California, Los
                                                                 Angeles
                                                      November 16, 2014

                                ODMRP-ASYM
                     draft-gerla-manet-odmrp-asym-01

Abstract

   This document describes ODMRP-ASYM, an extension for the On Demand
   Multicast Routing Protocol [ODMRP], which enables routers to use
   unidirectional links to route multicast messages.

Status of this Memo

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

   Internet-Drafts are working documents of the Internet Engineering
   Task Force (IETF). Note that other groups may also distribute
   working documents as Internet-Drafts. The list of current Internet-
   Drafts is at http://datatracker.ietf.org/drafts/current/.

   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."

   This Internet-Draft will expire on May 20, 2015.

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 powers (such as when using satellite
   communications). Most routing protocols make sure to avoid these
   links, typically by detecting and blacklisting them, and use only the
   fully connected graph formed by the bidirectional links of the
   network, even though taking advantage of these links can provide
   significant performance improvement, and even in some cases allow
   data to flow between weakly connected parts of the network, i.e.,
   parts of the network which are only connected by unidirectional
   links.

   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 paths to forward Join Reply messages
   to the multicast source, building the forwarding group 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 MUST use all its ODMRP
      Interfaces for the operations of this protocol. In other words,
      an ODMRP_ASYM Router MUST NOT select only a subset of its ODMRP
      Interfaces over which to process and generate ODMRP_ASYM control
      messages.

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 path 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 Discovery and Loop Marking messages (see
   Section 6) represent a Loop by an ordered list of one address per
   ODMRP-ASYM Router belonging to the Loop.

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 - The objective of building a Loop is to find an ODMRP-
   ASYM router, strictly closer (in terms of hop count) to the Loop
   Destination than the Loop Originator. A Loop can contain zero or
   more of such routers. If the Loop contains at least one such
   router, the closest to the Loop Destination is the Loop Summit.

Original Join Reply - Refers to the JR message, which failed
   transmission triggered the Loop Discovery process, as described in
   Section 9.

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.

l[n] -   is the nth element of list l. Lists in this specification
   are 1-based, meaning that they are indexed from 1 to length(list).

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 on the path between the Multicast
      Sources and Receivers.

4. Protocol Overview and Functioning

   The objective of this extension is to enable ODMRP Routers to make
   use of unidirectional links for forwarding multicast packets. In
   doing so, it may allow data to transit through shorter paths, and in
   some cases permit packets to be exchanged between weakly connected
   components of the network, i.e., parts of the network that are only
   connected by 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 Loop
       Originator 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, if there is one among
           the routers in the Loop.

   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 is
       closer to the Multicast Source than itself. If that is the
       case, it starts the 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 built
              with information carried within the LM 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 (see [ODMRP]
              section 10), as if it had received the corresponding Join
              Reply message.

5. Parameters and Constants

   In addition to those described in [ODMRP], this specification uses
   the parameters and constants 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. NO_SUMMIT has a value of 0.

6. Packets and Messages

   This section describes the protocol messages generated and processed
   by ODMRP-ASYM, according to the terminology defined in Section 2.
   The objective of this section is to describe the meaning and content
   of the fields contained in each message. The specifics the encoding
   of these messages using [RFC5444] is described in Section 7.

6.1. Join Query

   In addition to the fields specified in [ODMRP] section 7, Join Query
   messages generated by ODMRP-ASYM router have an optional hop count
   field (noted as JQ.HopCount), encoded by way of the <msg-hop-count>
   field of the JQ message header, as specified in [RFC5444]. This
   field MUST be included in every JQ message originated by an ODMRP-
   ASYM router, but MUST NOT be added to transmitted JQ messages if the
   received JQ message did not contain this field (see Section 9.1.3).

6.2. Loop Discovery

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

   LD.AddressLength encodes the length of the addresses carried by this
      message as follows:

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

LD.MulticastGroupAddress encodes the address of the Multicast Group,
   to which this Loop Discovery is addressed.

LD.Destination encodes the address of the Loop Destination.

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

LD.LoopSummit represenets the index in the LD.AddressList field of
   an address of the corresponding Loop Summit, if it exists. In
   other words, LD.LoopSummit is either NO_LOOPSUMMIT is the Loop has
   no Summit, or such that LD.AddressList[LD.LoopSummit] is an
   address of this Loop's Summit.

LD.MinHC represents 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 represents the maximum number of hops this message can
   traverse.

LD.HOPCOUNT represents the number of hops this message has already
   traversed.

6.3. Loop Marking

LM.AddressLength encodes the length of the addresses carried by this
   message as follows:

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

LM.MulticastGroupAddress encodes the address of the Multicast Group,
   to which this Loop Marking message is addressed.

LM.SourceAddress encodes the address of the Multicast Source,
   towards which the Loop Summit will transmit a Join Reply.

LM.AddressList is an ordered list of addresses of the routers this
   message has yet to transit through, i.e., head(LM.AddressList) is
   an address of the next router that should receive this LM. The
   Loop Marking message is effectively source-routed through these
   routers.

LM.LoopSummit represents the index in the LD.AddressList field of an
    address of the corresponding Loop Summit, if it is present in
    LM.AddressList. In other words, LD.LoopSummit is either
    NO_LOOPSUMMIT if the LM has already transited through the Loop
    Summit, or such that LD.AddressList[LD.LoopSummit] is an address
    of this Loop's Summit.

LM.SourceSequenceNumber corresponds to the sequence number carried
    by the Original Join Reply message.

7. RFC5444 Encoding

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

7.1. 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 1
   shows the mapping between the Loop Discovery elements described in
   Section 6.2 and their encoding. Each LD message MUST contain exactly
   one of each of these elements.

LD Element	RFC5444 Element
LD.AddressLength	<msg-addr-length>
LD.MulticastGroupAddress	Address in address block + TLV
LD.Destination	Address in address block + TLV
LD.AddressList	Addresses in address block + TLV
LD.LoopSummit	LOOPSUMMIT Message-type-specific TLV
LD.MinHC	MINHC Message-type-specific TLV
LD.HOPLIMIT	<msg-hop-limit>
LD.HOPCOUNT	<msg-hop-count>

Table 1: Loop Discovery Message Elements

   The following considerations apply for when encoding the elements
   described by Table 1:

   LD.AddressLength - Encodes addresses that are between 1 and 16 bytes
      long.

LD.MulticastGroupAddress - Is encoded by way of an address block
    with associated message-type-specific TLV of type ADDR-TYPE and
    value MULTICAST-GROUP-ADDRESS.

LD.Destination - Is encoded by way of an address block with
    associated message-type-specific TLV of type ADDR-TYPE and value
    DESTINATION-ADDRESS.

LD.AddressList - Is encoded by way of an address block with <num-
    addr> = length(LD.AddressList), containing the addresses in order.
    The address block has an associated message-type-specific TLV of
    type ADDR-TYPE and value ADDRESS-LIST.

LD.LoopSummit - 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 = NO_LOOPSUMMIT, <thasvalue> is
    cleared, and the <length> and <value> fields are omitted.

LD.HOPLIMIT - Is encoded by way of the <msg-hop-limit> field of the
    LD message header.

LD.HOPLIMIT - Is encoded by way of the <msg-hop-count> field of the
    LD message header.

7.2. 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 2
    shows the mapping between the Loop Marking elements described in
    Section 6.3 and their encoding. Each LD message MUST contain exactly
    one of each of these elements.

LM Element	RFC5444 Element
LM.AddressLength	<msg-addr-length>
LM.MulticastGroupAddress	Address in address block + TLV
LM.SourceAddress	Address in address block + TLV
LM.AddressList	Addresses in address block + TLV
LM.LoopSummit	LOOPSUMMIT Message-type-specific TLV
LM.SourceSequenceNumber	<msg-seq-num>

Table 2: Loop Marking Message Elements

    The following considerations apply for when encoding the elements

described by Table 2:

LM.AddressLength - Encodes addresses that are between 1 and 16 bytes
    long.

LM.MulticastGroupAddress - 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 - Is encoded by way of an address block with
    associated message-type-specific TLV of type ADDR-TYPE and value
    SOURCE-ADDRESS.

LM.AddressList - Is encoded by way of an address block with <num-
    addr> = length(LM.AddressList), containing the addresses in order.
    The address block has an associated message-type-specific TLV of
    type ADDR-TYPE and value ADDRESS-LIST.

LM.LoopSummit - 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 = NO_LOOPSUMMIT, <thasvalue> is
    cleared, and the <length> and <value> fields are omitted.

8. 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 to maintain this information.

8.1. Distance Set

   The Distance set contains distance tuples, recording the distance in
   hop count 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, carried by the
       corresponding Join Query message.

   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.

8.2. 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.

9. 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 10 of
   [ODMRP], as well as the following additional notation:

   OJR Refers to the corresponding Original Join Reply, as defined in
      Section 2.

9.1. Join Query

   ODMRP-ASYM requires that JQ messages carry a HopCount field, in order
   to maintain the Distance Set. This section specifies the additional
   processing required.

9.1.1. Join Query generation

   In addition to the fields described in [ODMRP] section 10.1.2, Join
   Query messages, originated by ODMRP-ASYM Routers, MUST be generated
   with the JQ.HopCount field present and set to 0.

9.1.2. Additional Join Query processing

   Upon receiving a valid Join Query message, an ODMRP-ASYM router MUST
   proceed as follows, after executing the steps specified in [ODMRP],
   section 10.1.3:

   1. Find the Distance tuple in the Distance Set, such as:

       * D_source = JQ.SourceAddress

       And update it in the following way:

       * D_hop_count := JQ.HopCount

       * D_seq_num := JQ.SequenceNumber

       * D_exp_time := current-time + ROUTE_TIMEOUT

   2. If no such tuple exists, create one with the following fields:

       * D_source = JQ.SourceAddress

       * D_hop_count := JQ.HopCount

       * D_seq_num := JQ.SequenceNumber

       * D_exp_time := current-time + ROUTE_TIMEOUT

9.1.3. Join Query transmission

   A JQ message, considered for forwarding, MUST be updated as follows,
   after following the process specified in [ODMRP], section 10.1.4:

   o If the HopCount field is present, update it so that JQ.HopCount :=
      JQ.HopCount + 1.

9.2. Loop Discovery

9.2.1. 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 this
      Router.

   o LD.HOPCOUNT > LD.HOPLIMIT, or LD.HOPCOUNT = LD.HOPLIMIT and
      LD.Originator is not an address of this Router.

   o LD.Originator 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.

9.2.2. 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 Router.

   o LD.LoopSummit := NO_SUMMIT.

   o LD.MinHC := D_hop_count.

9.2.3. 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 9.3.2

   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.

   2. Else, find the Pending Loop tuple (denoted "matching Pending Loop
       tuple") such that:

       * L_originator = head(LD.AddressList).

       * L_destination = LD.Destination.

       If it exists, update it such that L_exp_time :=
       PENDING_LOOP_TIMEOUT. Else, create one with the required fields
       set as above, and L_exp_time := PENDING_LOOP_TIMEOUT.

   3. The Loop Discovery message is then considered for forwarding,
       according to Section 9.2.4.

9.2.4. Loop Discovery Forwarding

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

   1. Append an address of this Router to LD.AddressList. The appended
       address MUST be an address of an ODMRP interface of this Router,
       but MAY be chosen freely among these addresses if there are more
       than one.

   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.

9.2.5. Loop Discovery Transmission

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

9.3. Loop Marking

9.3.1. Invalid Loop Marking messages

   A Loop Marking message (LM), 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) is not an address of this Router.

9.3.2. Loop Marking Generation

   A Loop Marking Message MUST be generated by an ODMRP-ASYM Router upon
   receiving a Loop Discovery (referenced as LD in the following)
   message advertising a valid closed Loop originated by this router, as
   described in Section 9.2.3. A Loop Marking message MUST be generated
   according to Section 6, with the following contents, computed from
   the corresponding Distance tuple and received LD 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)

9.3.3. Loop Marking Processing

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

   1. If LM.LoopSummit = 1, this Router is the Loop Summit for this
       Loop, and MUST do the following:

       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 this
       Router is either the Loop Summit, or after the Loop Summit in the
       corresponding Loop. Hence, it MUST join the corresponding
       forwarding group by adding or updating an entry in its forwarding
       table, by proceeding as follows:

       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, then update the tuple as follows:

> 1. F_seq_num := LM.SourceSequenceNumber.
>
> 2. Set F_exp_time := current-time + FG_TIMEOUT.

3. The Loop Marking message is then considered for forwarding, according to Section 9.3.4.

9.3.4. Loop Marking Message Forwarding

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

o Remove the first address from 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, or LM.LoopSummit := NO_LOOPSUMMIT if LM.LoopSummit = 1.

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

10. Security Considerations

As an extension of ODMRP, this protocol inherits from all the security considerations described in [ODMRP]. This document does not currently describe additional security concerns or specify any other security measure.

11. 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.

11.1. 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 3.

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

            Table 3: 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 3 will create two new Type
   Extension registries, with assignments specified in Table 4 and
   Table 5.

```
         +----------------+-------------+------------------+
         | Type Extension | Description | Allocation policy |
         +----------------+-------------+------------------+
         |     0-255      |  Unassigned |  Expert review   |
         +----------------+-------------+------------------+
```

            Table 4: LD Message TLV Type assignment: LOOPSUMMIT

```
         +----------------+-------------+------------------+
         | Type Extension | Description | Allocation policy |
         +----------------+-------------+------------------+
         |     0-255      |  Unassigned |  Expert review   |
         +----------------+-------------+------------------+
```

               Table 5: 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 6.

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

Table 6: 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 7.

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

Table 7: LD Message Address block TLV Type assignments: ADDR-TYPE

11.2. 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 8.

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

Table 8: 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 9.

```
+---------------+------------+------------------+
| Type extension | Description | Allocation policy |
+---------------+------------+------------------+
|     0-255     | Unassigned |  Expert review   |
+---------------+------------+------------------+
```

                Table 9: 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 10.

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

         Table 10: 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 11.

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

        Table 11: LD Message Address block TLV Type assignments: ADDR-TYPE

12. 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].

13. References

13.1. 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.

13.2. 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 [RFC5444]. [RFC5444] 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)
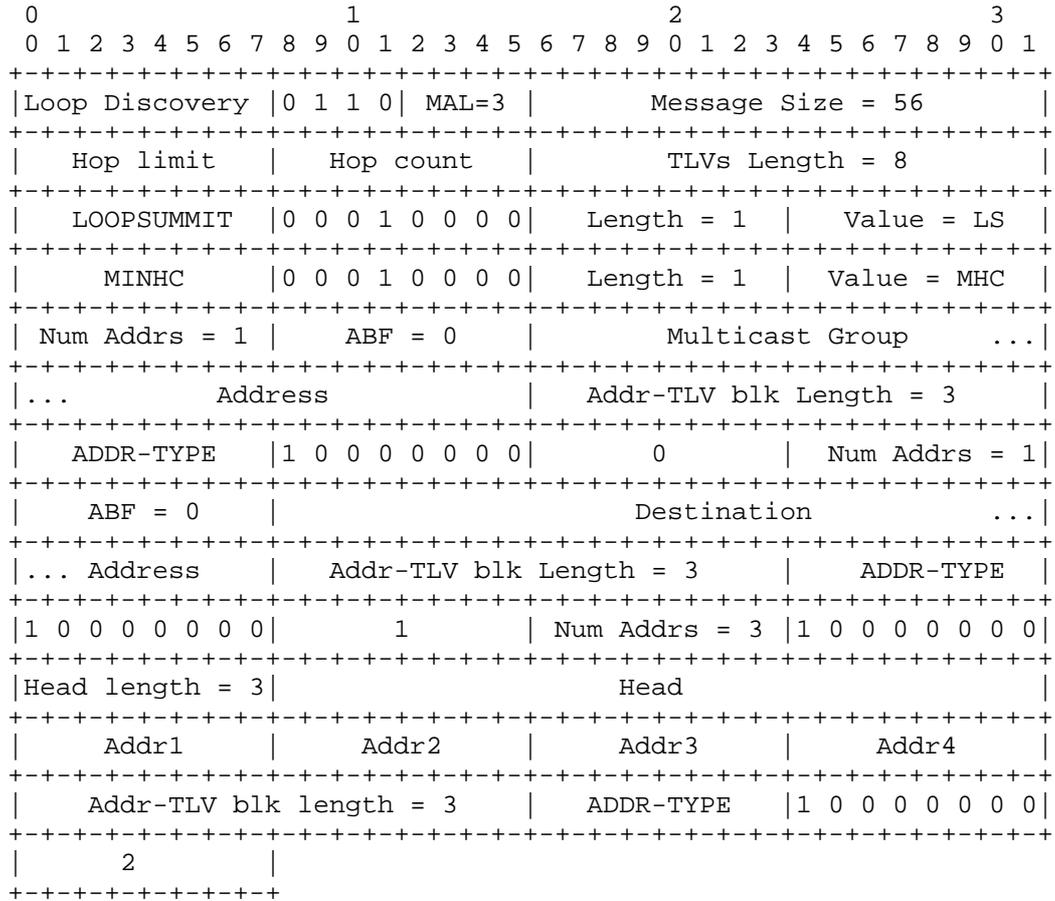respectively.

```
     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       |
    +-+-+-+-+-+-+-+-+
```

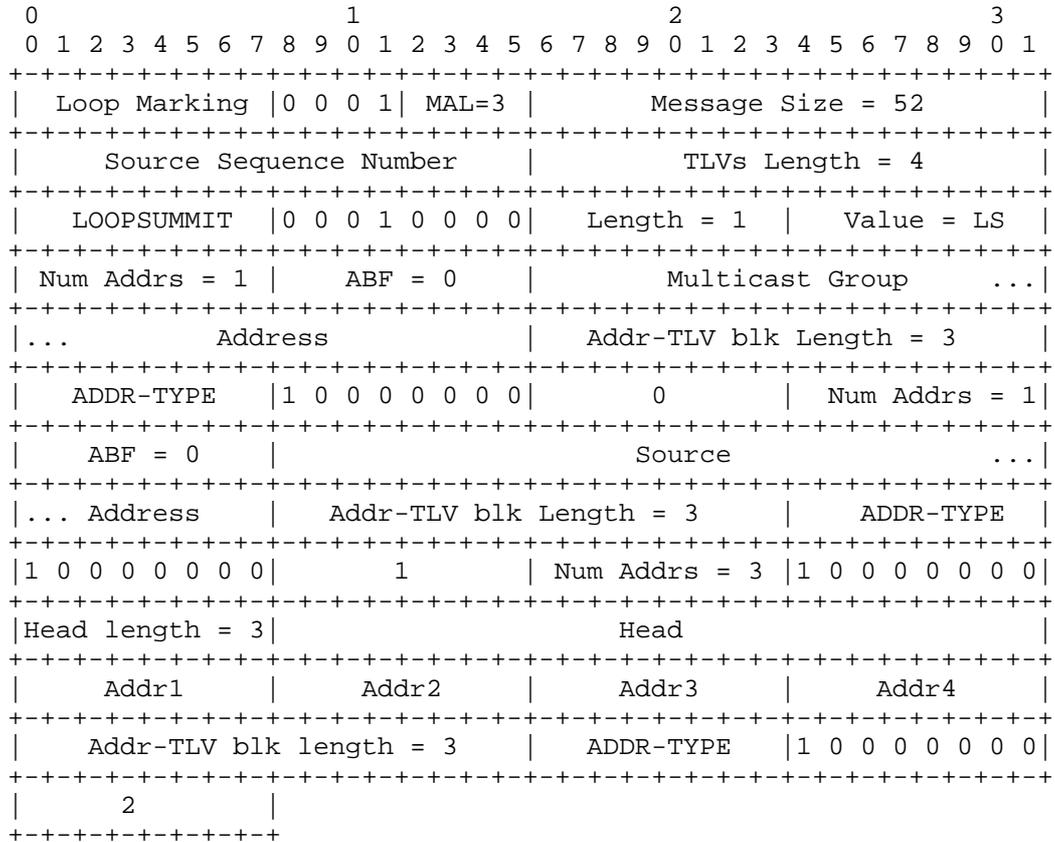   Figure 2

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