

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: August 29, 2015

Y. Yi
S. Lee
University of California, Los
Angeles
W. Su
The Boeing Company
M. Gerla
A. Colin de Verdiere
University of California, Los
Angeles
February 25, 2015

**On-Demand Multicast Routing Protocol (ODMRP) for Ad Hoc Networks
draft-gerla-manet-odmrp-05**

Abstract

The On-Demand Multicast Routing Protocol (ODMRP) is a multicast routing protocol designed for ad hoc networks with mobile hosts. ODMRP is a mesh-based, rather than a conventional tree-based, multicast scheme and uses a forwarding group concept (only a subset of nodes forwards the multicast packets via scoped flooding). It applies on-demand procedures to dynamically build routes and maintain multicast group membership, without relying on pre-existing unicast routing protocols. ODMRP is well suited for ad hoc wireless networks with mobile hosts where bandwidth is limited, topology changes frequently and rapidly, and power is constrained.

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 August 29, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](http://trustee.ietf.org/license-info) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	4
1.1.	Motivation and Experiments	4
2.	Terminology and Notation	5
2.1.	Notation	5
2.2.	Terminology	5
3.	Applicability Statement	7
4.	Protocol Overview and Functioning	7
4.1.	Routers and Interfaces	8
4.2.	Information Base Overview	8
4.3.	Signaling Overview	9
4.4.	Overview	9
5.	Parameters and Constants	11
5.1.	Router Parameters	11
5.2.	Interface Parameters	11
6.	Sequence Numbers	12
7.	Packets and Messages	12
7.1.	Join Query Format	12
7.2.	Join Reply Format	13
8.	RFC5444 Encoding	13
8.1.	Join Query Encoding	14
8.2.	Join Reply Encoding	14
9.	Information Bases	15
9.1.	Local Interface Set	15
9.2.	Neighbor Interface Set	15
9.3.	Multicast Routing Set	16
9.4.	Forwarding Table	16
9.5.	Pending Acknowledgements	17
9.6.	Pre-acknowledgements	18
9.7.	Blacklist	18
9.8.	Sent JQ set	19
10.	Protocol Details	19
10.1.	Join Query	20

10.1.1.	Invalid Join Queries	20
10.1.2.	Join Query Generation	20
10.1.3.	Join Query Processing	21
10.1.4.	Join Query Forwarding	22
10.2.	Join Reply	22
10.2.1.	Invalid Join Replies	23
10.2.2.	Join Reply Generation	23
10.2.3.	Join Reply Processing	23
10.2.4.	Join Reply Forwarding	25
10.2.5.	Join Reply Transmission	26
10.3.	Forwarding Group Maintenance	27
10.4.	Message Transmission	27
11.	Unidirectional Links Handling	27
12.	SMF considerations	29
13.	IGMP and MLD considerations	29
14.	Multicast Packet Forwarding	29
15.	Security Considerations	30
15.1.	Confidentiality	30
15.2.	Integrity	30
15.3.	Channel Overload	31
16.	IANA Considerations	31
16.1.	Join Query Registries	31
16.2.	Join Reply Registries	32
17.	Acknowledgements	33
18.	References	33
18.1.	Normative References	33
18.2.	Informative References	34
Appendix A.	Illustrations	35
A.1.	Join Query Message	35
A.2.	Join Reply Message	36
Authors' Addresses	37

1. Introduction

This document describes the On-Demand Multicast Routing Protocol (ODMRP) [[ODMRP-Journal](#)]. ODMRP applies "on-demand" routing techniques to avoid channel overhead and improve scalability. It uses the concept of "forwarding group" [[FGMP](#)], a set of nodes responsible for forwarding multicast data, to build a forwarding mesh for each multicast group. By maintaining and using a mesh instead of a tree, the drawbacks of multicast trees in mobile wireless networks (e.g., intermittent connectivity, traffic concentration, frequent tree reconfiguration, non-shortest path in a shared tree, etc.) are avoided. A soft-state approach is taken to maintain multicast group members, meaning that no explicit control message is required to leave the group. ODMRP does not rely on any unicast routing protocol: in particular, it can operate in conjunction with both reactive and proactive unicast routing protocols.

1.1. Motivation and Experiments

The main rationale for ODMRP is its potential to reduce control and traffic overhead in certain MANET deployments, typically where multicast traffic is relatively sparse. While this protocol has been extensively studied in simulations, it does not yet benefit from sufficient operational experience in order to be considered for Standards Track. In addition to general operational experience such as interoperability testing, this specification is intended to collect data on the following points:

- o As a multicast routing protocol for MANET, ODMRP can be compared with [[RFC6621](#)], but can also be used in conjunction, taking advantage of its Duplicate Packet Detection and optimized flooding mechanisms. The rationale behind ODMRP is that, with sparser traffic, and in particular less sources, ODMRP should reduce the control overhead and number of data packets transmitted by making use of Forwarding Groups. This hypothesis should be validated, and experiments and operational deployments demonstrating the scenarios in which ODMRP performs better, or worse, than [[RFC6621](#)] should be performed.
- o The potential scope of deployment of ODMRP should be assessed, particularly in comparison to other MANET protocols.
- o Default values and guidelines for the parameters described in [Section 5](#) should be provided, based on operational experience gathered from implementing and deploying this specification.
- o The feasibility of implementing ODMRP in common MANET situations should be examined. In particular, it should be determined if a

linux user space implementation is possible.

2. Terminology and Notation

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 also makes use of the terminology defined in [\[RFC5444\]](#). Additionally, it uses the notation defined in [Section 2.1](#), and the terminology defined in [Section 2.2](#).

2.1. Notation

ODMRP Routers generate and process messages, each of which has a number of distinct fields. For describing the protocol operations, specifically the generation and processing of such messages, the following notation is employed:

MsgType.field

where:

MsgType - is the type of message (e.g., JQ or JR);

field - is the field in the message (e.g., SourceAddress).

Furthermore, the following notational conventions are used:

a := b an assignment operator, whereby the left side (a) is assigned the value of the right side (b)

c = d a comparison operator, returning TRUE if and only if the value of the left side (c) is equal to the value of the right side (d)

[x] a list containing x as its only element

The different messages, their fields and their meaning are described in [Section 7](#).

2.2. Terminology

ODMRP Router - A router that implements this protocol. An ODMRP Router is equipped with at least one, and possibly more, ODMRP Interfaces.

ODMRP Interface - An ODMRP Router's attachment to a communication medium, over which it receives and generates control messages, according to this specification. An ODMRP Interface is assigned one or more addresses.

Neighbor (ODMRP) Router - An ODMRP Router A is a neighbor of another ODMRP Router B if B can receive control messages from A according to this specification. This relationship is not necessarily symmetrical.

Neighbor (ODMRP) Interface - An interface X of an ODMRP Router A is a neighbor relative to interface Y of ODMRP Router B if B can receive control messages sent by A over the link X - Y. The link, and this relationship, are not necessarily symmetrical.

Multicast session - The entity defined by a (multicast group, source) pair, representing the group to which a source sends multicast packets.

Forwarding group - A group of ODMRP Routers participating in multicast packet forwarding for a given Multicast Session. In particular, the Forwarding Group is constituted of the Multicast Source, the Multicast Receivers and the Intermediate Routers.

Multicast Receiver - An ODMRP Router is a Multicast Receiver, relative to a given Multicast Session, if it subscribes to the Multicast Session in order to receive data packets sent by the Multicast Source.

Intermediate Router - An Intermediate Router is an ODMRP Router that is a member of a Forwarding Group without being a Multicast Receiver. In other words, it joined the Forwarding Group to transmit control and data traffic between the Multicast Source and the Multicast Receivers.

Join Query - The control message sent by Multicast Sources to establish and update group memberships and routes.

Join Reply - The control message sent by Multicast Receivers and forwarded by Intermediate Routers to build the Forwarding Group according to group membership information.

Upstream - An ODMRP Router (A) is said to be "upstream" compared to another ODMRP Router (B), relatively to a given Multicast Session, if A is on the path, which is discovered by a Join Query-Join Reply exchange and used by data packets, between B and the Multicast Source. In other words, any data packet sent within the Multicast Session has to transit through A before reaching B.

Downstream - An ODMRP Router (A) is said to be "downstream" compared to another ODMRP Router (B), relatively to a given Multicast Session, if B is on the path which is discovered by a Join Query-Join Reply exchange and used by data packets, between A and the Multicast Source. In other words, A is "downstream" from B if B is "upstream" from A.

3. Applicability Statement

This protocol is a multicast routing protocol, intended for use in Mobile Ad Hoc Networks (MANETs). MANETs generally have constrained resources (processing power, battery, etc.) and very dynamic topologies. With ODMRP, routing state is installed and maintained in an on-demand fashion, which avoids the issue of frequent tree reconfiguration seen with more classic multicast routing protocols.

ODMRP does not rely on the use of any unicast routing protocol, whether reactive or proactive, but MAY be used coinjointly with such protocols, such as [\[RFC7181\]](#) or [\[RFC3561\]](#). Additionally, ODMRP can run in conjunction with [\[RFC6621\]](#), and take advantage of any optimized flooding mechanism used in the network, such as those offered by SMF, to disseminate Join Query messages as described in [Section 12](#).

4. Protocol Overview and Functioning

The objective of this protocol is to allow each ODMRP Router to:

- o Build a Forwarding Group only when it has data traffic to send to a Multicast group.
- o Maintain the Forwarding Group for as long as necessary, until there is no more data to be sent to the Multicast group.
- o Join any Forwarding Group, in order to receive multicast data packets from the corresponding multicast source. The decision to join a given Forwarding Group is triggered by Multicast membership information relative to the corresponding Multicast session. Such information can be received from other protocols, such as IGMP

[RFC3376] and MLD [[RFC3810](#)].

4.1. Routers and Interfaces

Each ODMRP Router MUST be provisioned with at least one ODMRP Interface, and keep a list of all these interfaces, as described in [Section 9](#). The management of these interfaces (addition, deletion, re-addressing of any interface) is out of scope for this document.

4.2. Information Base Overview

Protocol state is recorded in eight distinct information sets: the Local Interface Set, the Neighbor Interface Set, the Multicast Routing Set, the Forwarding Table, the Pending Acknowledgement Set, the Pre-acknowledgement Set, the Blacklist and the Sent JQ Set. With the exception of the Local Interface Set, all these information sets are both used and updated by this protocol.

The Local Interface Set records a list matching each ODMRP interface of this router to the addresses in use for this interface. This set is used, but not updated by this protocol.

The Neighbor Interface Set records all the known addresses of neighbor ODMRP interfaces, by way of recording data from received JQ messages. This set can also be updated by other protocols with knowledge of neighbor interfaces, such as [[RFC6130](#)].

The Multicast Routing Set contains tuples, each representing the address of a multicast group, the address of a source sending data to this multicast group, and the next hop towards the multicast source.

The Forwarding Table contains tuples, each representing a given Multicast session for which the ODMRP Router forwards packets.

The Pending Acknowledgement Set contains tuples, each corresponding to a Join Reply message which has been sent by this Router and is waiting for an acknowledgement from a chosen upstream Router.

The Pre-acknowledgement Set contains tuples, representing overheard Join Reply messages, that are not destined to this Router but may pre-acknowledge a future Join Reply from this Router.

The Blacklist contains tuples, corresponding to neighbor ODMRP Routers, with which connectivity has been detected to be unidirectional.

The Sent JQ Set matches interfaces of this Router with the address carried by the last JQ message to have transited through that

interface.

4.3. Signaling Overview

This protocol generates and processes the following routing messages:

Join Query - Generated by an ODMRP Router while it has data packets to send to a multicast group, and flooded periodically to maintain the Forwarding Group necessary to deliver these data packets. A Join Query message hence advertises a Multicast Session, and contains:

- * The multicast group address
- * The source address
- * A sequence number
- * The last address used by this interface to send a JQ message

Join Reply - Generated by an ODMRP Router belonging to a multicast session, in reply to a Join Query message advertising this multicast session (corresponding Join Query), then forwarded by ODMRP Routers belonging to the same multicast session along the reverse path to the multicast source. A Join Reply message contains:

- * The multicast group address and source address, identifying the multicast session
- * The sequence number carried by the corresponding Join Query
- * The address of the next hop on the path towards the multicast session source

4.4. Overview

The objectives of this protocol are achieved, for each ODMRP Router, by the following operations:

- o When having data to send to a multicast group, for which no Forwarding Group is already established, an ODMRP Router generates a Join Query and transmits it over all of its ODMRP Interfaces. It then periodically repeat this process, until it has no more data to send to the multicast group.
- o Upon receiving a Join Query, an ODMRP Router installs or refreshes a tuple in the Multicast Routing Set indicating the reverse path

towards the source of the Join Query, then considers it for forwarding, according to the forwarding mechanism specified for the network.

- o If this Router belongs to, i.e., has attached hosts which have subscribed to, the multicast session that the Join Query advertises, it originates a Join Reply and transmits it over all of its ODMRP Interfaces.
- o Upon receiving a Join Reply, an ODMRP Router inspects the next hop address carried by this packet:
 - * If it corresponds to the address of an interface of this Router, and if this Router has a tuple in its Multicast Routing Set, corresponding to the advertised source, the ODMRP Router belongs to the Forwarding Group for the Multicast Session. Consequently, it installs or refreshes the corresponding entry in its Forwarding Table. It then considers the Join Reply for forwarding, according to the forwarding mechanism specified for the network.
 - * Otherwise, it verifies if any Pending Acknowledgement tuple corresponds to this Join Reply and marks each such tuple as acknowledged. It silently discards the Join Reply.
- o After sending a Join Reply, addressed to an upstream router A, an ODMRP Router looks in its Pre-acknowledgement Set for a corresponding Overheard tuple.
 - * If such a tuple exists, the Overheard tuple is discarded and no further action is taken.
 - * Otherwise, i.e., if the Pre-acknowledgement Set does not contain any corresponding Overheard tuple, it creates a Pending Acknowledgement tuple in the Pending Acknowledgement Set. If this tuple expires without being acknowledged, the link with router A is considered unidirectional: it is blacklisted, and the current router MAY try other means of joining the Forwarding Group.
- o While it has data to send to the multicast group, an ODMRP Router periodically originates a Join Query and transmits it to all of its neighbors, in order to maintain the Forwarding Group.

5. Parameters and Constants

This specification uses both Router parameters, described in [Section 5.1](#), and per-interface parameters, described in [Section 5.2](#).

5.1. Router Parameters

This specification uses the following Router parameters:

ROUTE_REFRESH_INTERVAL - is the interval between two periodic Join Queries sent by a Multicast Source

FG_TIMEOUT - is the minimum time a Forwarding Tuple SHOULD be kept in the Forwarding Table after it was last refreshed

5.2. Interface Parameters

This specification uses the following interface parameters:

ROUTE_TIMEOUT - is the minimum time a Routing Tuple SHOULD be kept in the Routing Set after it was last refreshed

JR_RETRIES - is the number of times an ODMRP Router SHOULD attempt to retransmit a given Join Reply before declaring the link with the upstream neighbor interface unidirectional

ACK_TIMEOUT - is the time after which a Pending Tuple expires and MUST be considered invalid, as well as trigger the appropriate action according to [Section 11](#)

PRE_ACK_TIMEOUT - is the time after which an Overheard Tuple expires and MUST be considered invalid

LOCAL_ADDRESS_TIMEOUT - is the time after which a sent JQ tuple expires and MUST be considered invalid. This parameter SHOULD be less than the time an interface address is expected to be in use for the corresponding communication medium

NEIGHBOR_ADDRESS_TIMEOUT - is the time after which an address tuple of a neighbor interface tuple expires and MUST be considered invalid. This parameter SHOULD be less than the time an interface address is expected to be in use for the corresponding communication medium

6. Sequence Numbers

Each ODMRP Router maintains a single sequence number, which must be included in each Join Query message it generates. Each ODMRP Router MUST make sure that no two Join Query messages are generated with the same sequence number, and MUST generate sequence numbers such that these are monotonically increasing. This sequence number is used as freshness information for when comparing routes to the ODMRP Router having generated the message.

However, with a limited number of bits for representing sequence numbers, wrap-around (that the sequence number is incremented from the maximum possible value to zero) will occur. To prevent this from interfering with the operation of the protocol, the following MUST be observed. The term MAX_SEQ_NUM designates in the following the largest possible value for a sequence number. The sequence number S1 is said to be "greater than" (denoted '>') the sequence number S2 if:

$$S2 < S1 \text{ AND } S1 - S2 \leq \text{MAX_SEQ_NUM}/2 \text{ OR}$$
$$S1 < S2 \text{ AND } S2 - S1 > \text{MAX_SEQ_NUM}/2$$

7. Packets and Messages

This section describes the protocol messages generated and processed by ODMRP, according to the notations defined in [Section 2](#). The objective of this section is to specify the content and meaning of each message. The specifics of the encoding of these messages, including the exact type and length of each field, in accordance with [\[RFC5444\]](#), are described in [Section 8](#).

7.1. Join Query Format

A Join Query (JQ) message has the following fields:

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

$$\text{JQ.AddressLength} := \text{the length of an address in octets} - 1$$

JQ.MulticastGroupAddress encodes the address of the Multicast Group, to which this Join Query is addressed

JQ.SourceAddress encodes an address of the source of this Join Query

JQ.SequenceNumber encodes the sequence number (see [Section 6](#)) of the ODMRP Router, generating the Join Query message

JQ.LastAddress encodes the address set as the source address of the last IP datagram sent through the same interface and containing a JQ message, or of the IP datagram carrying this JQ message if no such datagram is known

7.2. Join Reply Format

A Join Reply (JR) message has the following fields:

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

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

JR.MulticastGroupAddress encodes the address of the Multicast Group, to which this Join Reply is addressed

JR.AckRequired is a boolean flag. When set ('1'), it specifies that the recipient of the Join Reply MUST acknowledge its reception by a sending Join Reply message. If cleared ('0'), the recipient of this message MAY suppress it's Join Reply transmission, according to [Section 10](#)

JR.SourceAddress encodes the address of the Source of the Multicast Session

JR.SequenceNumber encodes the sequence number (see [Section 6](#)) of the corresponding Join Query message

JR.NextHopAddress encodes the the address of the next hop on the path towards the source of the multicast session

8. [RFC5444](#) Encoding

This section describes the encoding of ODMRP messages using [\[RFC5444\]](#).

8.1. Join Query Encoding

This protocol defines the Join Query message type. Hence, according to [RFC5444], all Join Query messages are generated, processed and transmitted following this specification. Table 1 shows the mapping between the Join Query elements described in [Section 7.1](#) and their encoding. All elements described in Table 1 MUST be included in every Join Query message, with the exception of the JQ.LastAddress element. JQ.LastAddress MAY be omitted in a given JQ message if it corresponds to the source address of the IP datagram containing this message.

JQ Element	RFC5444 Element
JQ.AddressLength	<msg-addr-length>
JQ.SourceAddress	<msg-orig-addr>
JQ.MulticastGroupAddress	Address in address block + TLV
JQ.SequenceNumber	<msg-seq-num>
JQ.LastAddress	Address in address block + TLV

Table 1: Join Query Message Elements

8.2. Join Reply Encoding

This protocol defines the Join Reply message type. Hence, according to [RFC5444], all Join Reply messages are generated, processed and transmitted following this specification. Table 2 shows the mapping between the Join Reply elements described in [Section 7.2](#) and their encoding. With the exception of the ACKREQUIRED TLV, all elements described in Table 2 MUST be included in every Join Reply message.

JR Element	RFC5444 Element
JR.AddressLength	<msg-addr-length>
JR.SourceAddress	<msg-orig-addr>
JR.MulticastGroupAddress	Address in address block + TLV
JR.SequenceNumber	<msg-seq-num>
JR.NextHopAddress	Address in address block + TLV
JR.AckRequired	ACKREQUIRED TLV

Table 2: Join Reply Message Elements

9. Information Bases

Each router maintains an Information Base, containing a Local Interface Set, a Neighbor Interface Set, a Multicast Routing Set, a Forwarding Table, a Pending Acknowledgement Set, a Pre-Acknowledgement Set, a Blacklist and a Sent JQ Set, as described in the following sections. These information sets are given so as to facilitate description of message generation, forwarding and processing rules. In particular, an implementation may choose any representation or structure for when maintaining this information.

9.1. Local Interface Set

The Local Interface Set records a list of all the interfaces of the ODMRP Router, which participate in the operations of this protocol; that is, over which ODMRP control messages are exchanged, according to this specification. Each tuple of the Interface Set, or Interface Tuple, is as follows:

(I_interface, I_interface_address_list)

Where:

I_interface - The local ODMRP Interface

I_interface_address_list - The list of addresses used by the local interface in the operations of this protocol.

9.2. Neighbor Interface Set

The Neighbor Interface Set records the known addresses of Neighbor ODMRP Interfaces. It is used by this protocol, and can be updated by this protocol or by any other suitable protocol in operation that provides the necessary information, such as NHDP [[RFC6130](#)]. Each neighbor interface tuple is as follows:

(N_interface_address_list)

Where:

N_interface_address_list - Is an unordered list of at least one address tuple (i_addr, i_addr_exp_time), where:

i_addr - is a known address of the Neighbor Interface, i.e., an address that was set as the sender of an IP datagram sent through this interface

i_addr_exp_time - is the time at which the address tuple MUST be considered expired and thus MUST NOT be taken in considerations for the operations of this protocol

A neighbor interface tuple that contains no valid (i.e., non-expired) address tuple MUST considered expired and MUST NOT be taken in considerations for the operations of this protocol

9.3. Multicast Routing Set

The Multicast Routing Set contains Routing Tuples, indicating the path towards Multicast Sources, and containing the following fields:

(R_source, R_next_hop, R_local_interface,
R_seq_num, R_exp_time)

Where:

R_source - is the address of the Multicast Source.

R_next_hop - is an address of the next hop along the path to the Multicast Source, i.e., an address of one of the interfaces of the neighbor ODMRP Router, from which the last valid Join Query message from this source was received, as recorded by the packet containing this Join Query.

R_local_interface - is the local interface, through which the next hop can be reached.

R_seq_num - corresponds to the JQ.SequenceNumber of the last valid Join Query originated by the Multicast Source and received by this ODMRP Router.

R_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.

9.4. Forwarding Table

The Forwarding Table contains Forwarding Tuples, representing Multicast Sessions for which the ODMRP Router forwards messages, i.e., the ODMRP Router is part of these Multicast Sessions' Forwarding Groups. These tuples are as follows:

(F_multicast_group, F_multicast_source,
F_seq_num, F_exp_time)

Where:

F_multicast_group - is the address of the Multicast Group of the Multicast Session, for which the ODMRP Router forwards messages.

F_source - is the address of the Multicast Source of the Multicast Session, for which the ODMRP Router forwards messages.

F_seq_num - is the sequence number, corresponding to the last Join Query sent by the multicast source for the multicast session.

F_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.

9.5. Pending Acknowledgements

The Pending Acknowledgements Set contains Pending Acknowledgement tuples, representing Join Reply messages that are waiting to be acknowledged by the selected upstream Forwarding Group member. These tuples are as follows:

(P_multicast_group, P_multicast_source, P_seq_num,
P_local_interface, P_next_hop, P_nth_time, P_exp_time)

Where:

P_multicast_group - is the JR.MulticastGroupAddress carried in the Join Reply awaiting acknowledgement (henceforth corresponding Join Reply).

P_multicast_source - is the JR.SourceAddress field carried in the corresponding Join Reply.

P_seq_num - is the JR.SequenceNumber field of the corresponding Join Reply.

P_next_hop - is the JR.NextHopAddress field of the corresponding Join Reply.

P_local_interface - is the local interface, through which the Join Reply was sent.

P_nth_time - corresponds to the number of times this Join Reply has been previously sent without being acknowledged.

P_exp_time - is the time at which this tuple MUST be considered expired.

P_acknowledged - is a boolean indicating whether the corresponding Join Reply has been acknowledged.

9.6. Pre-acknowledgements

The Pre-acknowledgements Set contains Overheard Tuples, corresponding to Join Reply messages, which have been sent by neighbors of this ODMRP Router but do not contain an address of this Router and do not acknowledge any tuple in the Pending Acknowledgement Set. The Overheard Tuples are as follows:

(O_multicast_group, O_multicast_source, O_seq_num,
O_originator, O_exp_time)

Where:

O_multicast_group - is the JR.MulticastGroupAddress carried in the overheard Join Reply.

O_multicast_source - is the JR.SourceAddress field carried in the corresponding Join Reply.

O_seq_num - is the JR.SequenceNumber field of the corresponding Join Reply.

O_originator - is the address of the ODMRP Router's interface which has sent the Join Reply.

O_exp_time - is the time at which this tuple expires MUST be considered invalid.

9.7. Blacklist

The Blacklist contains Blacklisted Tuples, corresponding to neighbor ODMRP Router interfaces, with which connectivity has been detected to be unidirectional, e.g., which have not acknowledged Join Replies from this Router, as specified in [Section 10](#). In other words, a Blacklisted Tuple corresponds to a link between one local interface and one neighbor interface which has been detected to be unidirectional or broken. The Blacklist Tuples are as follows:

(B_neighbor_interface, B_local_interface, B_exp_time)

Where:

B_neighbor_interface_address_list - is a list of addresses belonging to the blacklisted interface.

B_local_interface - is the interface of this ODMRP router over which packets from the blacklisted interface were received.

B_exp_time - is the time at which this tuple expires and MUST be considered invalid.

9.8. Sent JQ set

The Sent JQ Set contains tuples matching transmitted (generated or relayed) Join Queries with interfaces addresses. Each of its tuples contains the source address, multicast group address and sequence number uniquely identifying a JQ message, as well as an interface and the address of that interface that was advertised when transmitting the packet containing the Join Query. More precisely, given a transmitted Join Query and an interface over which it was transmitted, a tuple of this set, or Sent JQ Tuple, is as follows:

Where:

S_interface - is the local interface, through which the JQ message was sent

S_interface_address - is the address of the ODMRP interface that was set as the packet source

S_exp_time - is the time at which this tuple expires and MUST be considered invalid.

10. Protocol Details

This protocol generates and processes Join Query and Join Reply messages, according to the operations described in the following sections. This section uses the additional notation and variables:

previous-hop-address - refers to the address of the neighbor ODMRP interface recorded by the source address field of the IP datagram carrying the message currently being processed (Join Query or Join Reply)

this Router - refers to the ODMRP Router generating, processing or forwarding the message (Join Query or Join Reply)

Receiving Interface (receiving-interface) - refers to the local ODMRP Interface, over which the message currently being processed was received

10.1. Join Query

A Join Query is generated by an ODMRP Router, which has data to send to a multicast group, for which no multicast session has been initialized. Join Queries are then periodically originated by the ODMRP Router while it has data to send to the multicast group.

10.1.1. Invalid Join Queries

A Join Query, received by an ODMRP Router, is invalid and MUST be discarded without processing (and in particular, MUST NOT be considered for forwarding) if any of these conditions applies:

- o The address length carried by the Join Query (see [Section 7](#)) differs from the length of the addresses of this Router
- o The Multicast Routing Set of this Router contains a Multicast Routing tuple, for which:
 - * $R_multicast_source = JQ.SourceAddress$, and
 - * $R_seqnum > JQ.SequenceNumber$ or $R_seqnum = JQ.SequenceNumber$
- o $JQ.SourceAddress$ is an address of an interface of this Router
- o The Blacklist contains a Blacklisted Tuple, for which
 - * $previous-hop-address$ is contained in $B_neighbor_interface_address_list$
 - * $B_local_interface = receiving-interface$

10.1.2. Join Query Generation

A Join Query is generated according to [Section 7](#) with the following content:

- o $JQ.AddressLength$ set to the length of the addresses of this Router minus 1, as specified in [Section 7](#)

- o JQ.MulticastGroupAddress set to the address of the multicast group, to which this Router is sending data
- o JQ.SourceAddress set to an address of this ODMRP Router
- o JQ.SequenceNumber set to the current sequence number of this Router, as specified in [Section 6](#)

10.1.3. Join Query Processing

Upon receiving a valid Join Query message, an ODMRP Router proceeds as follows:

1. Find the neighbor interface tuple such as
N_interface_address_list contains an address tuple with i_addr = previous-hop-address, and update the address tuple such as
i_addr_exp_time := current-time + NEIGHBOR_ADDRESS_TIMEOUT
2. If no such tuple exists, create one with:
 - * N_interface_address_list := [(previous-hop-address, current-time + NEIGHBOR_ADDRESS_TIMEOUT)]
3. Find the Routing Tuple which satisfies: R_source = JQ.SourceAddress
4. If no such tuple exists, create a Routing Tuple with the following fields:
 - * R_source := JQ.SourceAddress
 - * R_next_hop := previous-hop
 - * R_local_interface := receiving-interface
 - * R_seq_num := JQ.SequenceNumber
 - * R_exp_time := current-time + ROUTE_TIMEOUTand insert this tuple in the Routing Set
5. Else, i.e., if such a tuple exist, update it as follows:
 - * R_next_hop := previous-hop
 - * R_seq_num := JQ.SequenceNumber

- * `R_local_interface` := receiving-interface
- * `R_exp_time` := current-time + `ROUTE_TIMEOUT`
- 6. Consider the Join Query for forwarding, according to [Section 10.1.4](#)
- 7. If this Router is a member of the Multicast Group, addressed by `JQ.MulticastGroupAddress`, create a new Join Reply according to [Section 10.2](#) and transmit it to all of this Router's neighbors

[10.1.4.](#) Join Query Forwarding

This section defines the following additional variables:

`this-interface` - is the ODMRP interface being considered

`packet-source-address` - is the source address of the outbound IP datagram carrying the JQ message being transmitted

For each ODMRP interface over which a JQ message is to be transmitted, a router MUST proceed as follows:

- o Find the corresponding sent JQ tuple in the sent JQ set, such as `S_interface = this-interface`
- o If no such tuple exists, create one with:
 - * `S_interface` := `this-interface`
 - * `S_interface_address` := `packet-source-address`
 - * `S_exp_time` := current-time + `LOCAL_ADDRESS_TIMEOUT`
- o Set `JQ.LastAddress` to `S_interface_address`
- o Update the corresponding sent JQ tuple such as:
 - * `S_interface_address` = `packet-source-address`
 - * `S_exp_time` = current-time + `LOCAL_ADDRESS_TIMEOUT`

[10.2.](#) Join Reply

A Join Reply is generated by an ODMRP Router when it receives a Join Query such that at least one host attached to the ODMRP Router is a member of the Multicast Session advertised by the Join Query. This section makes use of the variable "new-jr", which is a boolean flag

set to TRUE if the Join Reply being processed contains more recent data than in the current information base. It has an initial value of FALSE.

10.2.1. Invalid Join Replies

A Join Reply, received by an ODMRP Router, is invalid and MUST be discarded without processing (and in particular, MUST NOT be considered for forwarding) if:

- o The address length carried by the Join Reply (see [Section 7](#)) differs from the length of the address of the ODMRP Router
- o There exists a Forwarding Tuple in this Router's Forwarding Group table, such as:
 - * F_source = JR.MulticastSourceAddress
 - * F_seq_num > JR.SequenceNumber

10.2.2. Join Reply Generation

An ODMRP Router MUST generate a Join Reply in response to a received Join Query (henceforth "corresponding Join Query"), if at least one host attached to this Router is a member of the Multicast Session, advertised by the Join Query. A Join Reply is generated according to [Section 7](#) with the following content:

- o JR.AddressLength is set to the length of the address of this router minus 1, as specified in [Section 7](#)
- o JR.MulticastGroupAddress is set to JQ.MulticastGroupAddress for the corresponding Join Query
- o JR.SourceAddress is set to JQ.SourceAddress for the corresponding Join Query
- o JR.SequenceNumber is set to JQ.SequenceNumber for the corresponding Join Query
- o JR.NextHopAddress is set to the source address of the IP datagram containing the Join Query message

10.2.3. Join Reply Processing

Upon receiving a valid Join Reply, an ODMRP Router proceeds as follows:

1. If JR.NextHopAddress corresponds to an address recorded in the Local Interface Set of this ODMRP Router:
 1. Find the Forwarding Tuple (henceforth Matching Forwarding Tuple) such that:
 - + F_multicast_group = JR.MulticastGroupAddress
 - + F_multicast_source = JR.MulticastSourceAddress
 2. If no such tuple exists, insert in the Forwarding Table a new Forwarding Tuple such that:
 - + F_multicast_group = JR.MulticastGroupAddress
 - + F_multicast_source = JR.MulticastSourceAddress
 - + F_seq_num = JR.SequenceNumber
 - + F_exp_time = current-time + FG_TIMEOUTAnd set new-jr to TRUE
 3. Otherwise, the variable "new-jr" is set to TRUE if JR.SequenceNumber > F_seq_num, and to FALSE otherwise. Then, the pre-existing Matching Forwarding Tuple is updated as follows:
 - + F_seq_num := JR.SequenceNumber
 - + F_exp_time := current-time + FG_TIMEOUT
 4. If new-jr = TRUE or if JR.AckRequired is set the Join Reply is considered for forwarding. Otherwise, it is not processed further; in particular, it MUST NOT be considered for forwarding.
2. Otherwise, find the Multicast Routing Tuple in the Routing Set (henceforth "Matching Multicast Routing Tuple"), such as:
 - * R_source = JR.SourceAddress
 - * R_seq_num <= JR.SequenceNumberIf previous-hop-address = R_next_hop, then:
 3. If the Pending Acknowledgement Set contains a Pending Tuple (henceforth "Matching Pending Tuple") such as:

- + P_multicast_group = JR.MulticastAddress
- + P_multicast_source = JR.SourceAddress
- + P_seq_num = JR.SequenceNumber
- + P_next_hop = previous-hop-address

The Matching Pending Tuple MUST be updated as follows:

- + P_acknowledged = TRUE
- + P_exp_time = EXPIRED

The Join Reply is not processed further, and in particular MUST NOT be considered for forwarding

4. Otherwise, if the Pre-Acknowledgement Set does not contain any Overheard Tuple such as:

- + O_multicast_group = JR.MulticastGroupAddress
- + O_multicast_source = JR.SourceAddress
- + O_seq_num = JR.SequenceNumber
- + O_originator = previous-hop-address

Insert a tuple with these fields, and O_exp_time = current-time + PRE_ACK_TIMEOUT in the Pre-Acknowledgement Set. The Join Reply is not processed further, and in particular MUST NOT be considered for forwarding

3. Otherwise, the Join Reply is silently discarded without further processing

10.2.4. Join Reply Forwarding

A Join Reply, considered for forwarding, MUST be updated as follows:

- o Find the Matching Routing Tuple, such that:

- * R_source = JR.MulticastSourceAddress
- * R_seq_num <= JR.SequenceNumber

- o If no such tuple exists, then the Join Reply is not processed further, and in particular MUST NOT be forwarded
- o Otherwise, set JR.NextHop to R_next_hop

The Join Reply is then transmitted according to [Section 10.2.5](#)

10.2.5. Join Reply Transmission

A Join Reply is transmitted to all of an ODMRP Router's neighbors, in order to achieve two objectives:

- o Set up or refresh the corresponding Forwarding Tuple for the upstream ODMRP neighbor
- o If the Join Reply was not originated by this router, acknowledge its reception to the previous hop

Before transmitting the Join Reply, the Information Base is updated as follows:

1. If the Pre-acknowledgement Set contains a tuple, such that:

- * O_multicast_group = JR.MulticastGroupAddress
- * O_multicast_source = JR.SourceAddress
- * O_seq_num = JR.SequenceNumber
- * O_originator = JR.NextHopAddress

Then clear the JR.AckRequired flag, and set O_exp_time to EXPIRED

2. Otherwise, if the Pending Acknowledgement Set contains a Pending Tuple such as:

- * P_multicast_group = JR.MulticastGroupAddress
- * P_multicast_source = JR.SourceAddress
- * P_seq_num = JR.SequenceNumber
- * P_next_hop = JR.NextHopAddress

Then set JR.AckRequired, and increase P_nth_time by 1

3. Finally, if neither the Pre-acknowledgement Set nor the Pending Acknowledgement Set contain a corresponding tuple:

1. Insert a Pending Tuple in the Pending Acknowledgement Set, such as:
 - + P_multicast_group = JR.MulticastGroupAddress
 - + P_multicast_source = JR.SourceAddress
 - + P_seq_num = JR.SequenceNumber
 - + P_next_hop = JR.NextHopAddress
 - + P_nth_time = 1
 - + P_acknowledged = FALSE
 - + P_expiration_time = current-time + ACK_TIMEOUT
2. Clear the JR.AckRequired flag

10.3. Forwarding Group Maintenance

While an ODMRP Router has data to send to a Multicast Group (on behalf of the Multicast Source), it MUST maintain the Forwarding Group generated by the initial Join Query. To this end, it MUST periodically generate JQ messages, according to [Section 10.1.2](#). The interval between two Join Queries SHOULD be no less than ROUTE_REFRESH_INTERVAL. Note should be taken that, if the Multicast Session has no member other than the source, the Forwarding Group may contain only the designated ODMRP Router for the Multicast Source. That Router still needs to periodically flood Join Queries in order to rebuild a Forwarding Group if necessary.

10.4. Message Transmission

When using physical media subject to collisions and packet loss, both Join Query and Join Reply messages SHOULD be jittered to minimize the effect of collisions, as described in [[RFC5148](#)]

11. Unidirectional Links Handling

After sending a Join Reply, an ODMRP Router MUST verify that the upstream neighbor has joined the Forwarding Group. To this end, the following three mechanisms are used after transmitting a given Join Reply:

- o If the ODMRP Router overhears a corresponding Join Reply from the upstream neighbor (see [Section 10.2.3](#)), this verifies that the

link is bidirectional and that the upstream neighbor has joined the Forwarding Group (passive acknowledgement)

- o If the ODMRP Router has already overheard a corresponding Join Reply from the upstream neighbor prior to transmitting its own Join Reply, this means that the upstream neighbor has already joined the Forwarding Group (see [Section 10.2.3](#)) (pre-acknowledgement)
- o Otherwise, i.e., if neither the pre-acknowledgement nor the passive acknowledgement have verified that the upstream neighbor joined the Forwarding Group (i.e., if the corresponding Pending Tuple expires with P_acknowledged set to False), then the ODMRP Router MUST proceed as follows:
 1. If the corresponding Pending tuple verifies $P_nth_time < JR_RETRIES$, then the ODMRP Router MUST retransmit the Join Reply with the JR.AckRequired flag set
 2. Otherwise, the link between the local interface and the interface of the upstream ODMRP Router identified by JR.NextHopAddress is considered unidirectional. In that case, the ODMRP Router SHOULD proceed as follows:
 - + Find the neighbor interface tuple such that N_address_list contains an address tuple with i_addr = JR.NextHopAddress, and set the variable blacklisted-addresses to the list of addresses contained in N_address_list
 - + Otherwise, if no such tuple exists, set the variable blacklisted-addresses to [JR.NextHopAddress]
 - + Add a tuple in the Blacklist such as:
 - B_neighbor_interface_address_list := blacklisted-addresses
 - B_local_interface := P_local_interface
 - B_exp_time = current-time + BLACKLIST_TIMEOUT

An ODMRP Router MAY attempt to use other mechanisms, such as [\[I-D.gerla-manet-odmrp-asym\]](#), to resume the Forwarding Group building process, instead of or in addition to using the Blacklist

12. SMF considerations

This protocol MAY be run in conjunction with SMF [[RFC6621](#)], and benefit from some of its features. In particular, if SMF is in use, it is RECOMMENDED that its duplicate packet detection feature described in [Section 6](#) be used for multicast packet forwarding. Additionally, optimized flooding mechanisms, such as E-CDS or S-MPR, as specified in Appendices A through C of [[RFC6621](#)], MAY be used to flood Join Query messages throughout the network.

13. IGMP and MLD considerations

In order to determine whether or not it needs to reply to a Join Query message with a Join Reply message (as specified in [Section 10.1.3](#)), an ODMRP Router needs Multicast Group membership information. Such information can be provided by protocols such as IGMP [[RFC3376](#)] and/or MLD [[RFC3810](#)]. In particular, an ODMRP Router MUST reply with a Join Reply message to a valid Join Query messages advertising a Multicast Session if any of those conditions apply:

- o This Router is subscribed to the corresponding Multicast Group.
- o A host attached to this Router has signaled, for example using IGMP, that it has subscribed to the corresponding Multicast Group.

14. Multicast Packet Forwarding

ODMRP Routers originating and forwarding multicast packets MUST implement a duplicate packet detection (DPD) mechanism. If using IPv4 or IPV6 addresses, the use of SMF [[RFC6621](#)] is RECOMMENDED, as described in [Section 12](#).

An ODMRP Router, receiving a non-duplicate multicast data packet, transmits it over all of its interfaces if it is a member of the forwarding group for this data packet, i.e., there exists a tuple in the Forwarding Group Table such as:

F_multicast_group correspond to the multicast address of this packet

F_multicast_source corresponds to the source of this packet

15. Security Considerations

As a multicast routing protocol, this protocol is potentially vulnerable to a number of attacks. This section attempts to describe the envisioned threats to the protocol, as well as some guidelines as to how to ensure confidentiality and integrity of the operations of ODMRP, and to mitigate threats of network overload.

This protocol relies on the use of a Duplicate Packet Detection (DPD) mechanism, such as one described in [[RFC6621](#)] (SMF), and suggests the use of optimized flooding to disseminate JQ messages. Some deployments of ODMRP are thus expected to function on top of [[RFC6621](#)] by taking advantage of the DPD and optimized flooding mechanisms provided by SMF. Such deployments are thus subject to the same security threats as SMF, such as those described in [[I-D.ietf-manet-smf-threats](#)].

15.1. Confidentiality

ODMRP routers which forward packets for multicast data source have to periodically transmit JQ messages throughout the network. In an unsecured network, an attacker could then eavesdrop on those messages and learn part or all of the network topology, depending on the traffic pattern.

15.2. Integrity

ODMRP relies on routers, in particular intermediate routers, to correctly transmit JQ and JR messages. An ODMRP router could, by malice or malfunction, originate JQ messages on behalf of a target multicast source with high enough sequence numbers to replace routing information in other routers. Such behavior would prevent the interested multicast receivers from receiving data packets sent by the target multicast source. An ODMRP router could also forward JQ messages with altered sequence numbers, thus preventing future routing updates. Both behaviors can be mitigated by end-to-end authentication of routing messages.

If NHDP [[RFC6130](#)] is not in use to update the Neighbor Interface Set, ODMRP relies on routers correctly informing their neighbors of the addresses they use via the JQ.LastAddress field. Upon transmission of a JQ message, an ODMRP router could, by malice or malfunction, set JQ.LastAddress to a network address that does not belong to this router (address spoofing). This could force neighbor ODMRP routers to blacklist this address in case the malicious router simulate unidirectional links by withholding JR messages. This behavior would break or slow down protocol convergence, potentially triggering data packet loss for multicast receivers. If NHDP is in use, the

deployment is subject to its own security vulnerabilities, such as those described in [\[RFC7186\]](#).

15.3. Channel Overload

ODMRP's main construct, the forwarding group, is built and maintained by having the source ODMRP router periodically flood JQ messages, which can be a costly operation in terms of bandwidth, processing and battery life, if applicable. A malicious router could flood JQ messages at a very high rate to overload the network. It is thus RECOMMENDED that ODMRP routers in a given deployment implement a rate-limit mechanism to prevent such behavior.

The efficiency (in terms of number of multicast data packets transmitted) of forwarding groups depends on routers actually sending JR messages only when necessary, in order to build a graph as sparse as possible and avoid redundant transmissions. Thus an ODMRP router which replies to JQ messages by transmitting one JR message for each of its known neighbors and with JR.NextHopAddress set to an address of this neighbor, would severely harm the efficiency of ODMRP by forcing the routers to build a forwarding group with unnecessary redundancy. Such behavior could also result in routing loops.

16. IANA Considerations

This specification defines two new Message Types, Join Query and Join Reply, which must be allocated from the "Message Type" repository of [\[RFC5444\]](#).

16.1. Join Query Registries

IANA is requested to create a registry for Message-Type-specific Message TLV Types for Join Query messages, with initial assignments according to Table 3.

+-----+-----+-----+		
Type	Description	Allocation Policy
+-----+-----+-----+		
128-223	Unassigned	Expert Review
+-----+-----+-----+		

Table 3: Join Query Message-Type-specific Message TLV Types

IANA is requested to create a registry for Message-Type-specific Address Block TLV Types for Join Query messages, with initial assignments according to Table 4.

Type	Description	Allocation Policy
128	ADDR-TYPE	
129-223	Unassigned	Expert Review

Table 4: Join Query Message-Type-specific Address Block TLV Types

Allocation of the ADDR-TYPE TLV from the Join Query specific Address Block TLV Types will create a new Type Extension Registry with initial assignments as specified in Table 5.

Name	Type	Type Ext.	Description	Al. Policy
ADDR-TYPE	128	0	MULTICAST-GROUP-ADDRESS	
ADDR-TYPE	128	1	LAST-ADDRESS	
ADDR-TYPE	128	2-255	Unassigned	Expert Review

Table 5: Address Block TLV Type assignment for ADDR-TYPE

16.2. Join Reply Registries

IANA is requested to create a registry for Message-Type-Specific Message TLV Types for Join Reply messages, with initial assignments according to Table 6.

Type	Description	Allocation Policy
128	ACKREQUIRED	
129-223	Unassigned	Expert Review

Table 6: Join Reply Message-Type-specific Message TLV Types

IANA is requested to create a registry for Message-Type-specific Address Block TLV Types for Join Reply messages, with initial assignments according to Table 7.

Type	Description	Allocation Policy
128	ADDR-TYPE	
129-223	Unassigned	Expert Review

Table 7: Join Reply Message-Type-specific Address Block TLV Types

Allocation of the ADDR-TYPE TLV from the Join Reply specific Address Block TLV Types will create a new Type Extension Registry with initial assignments as specified in Table 8.

Name	Type	Type Ext.	Description	Al. Policy
ADDR-TYPE	128	0	MULTICAST-GROUP-ADDRESS	
ADDR-TYPE	128	1	NEXT-HOP-ADDRESS	Expert Review
ADDR-TYPE	128	2-255	Unassigned	Expert Review

Table 8: Address Block TLV Type assignment for ADDR-TYPE

17. Acknowledgements

The authors would like to thank Thomas Clausen and Justin Dean for their insightful reviews and comments.

18. References

18.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [RFC 2119](#), [BCP 14](#), March 1997.
- [RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", [RFC 3376](#), October 2002.
- [RFC3810] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", [RFC 3810](#), June 2004.
- [RFC5148] Clausen, T., Dearlove, C., and B. Adamson, "Jitter

Considerations in Mobile Ad Hoc Networks (MANETs)",
[RFC 5148](#), February 2008.

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

[RFC6621] Macker, J., "Simplified Multicast Forwarding", [RFC 6621](#),
May 2012.

[18.2.](#) Informative References

[FGMP] Chiang, C., Gerla, M., and L. Zhang, "Forwarding Group
Multicast Protocol (FGMP) for Multihop, Mobile Wireless
Networks", Avril 1998.

[I-D.gerla-manet-odmrp-async]
Gerla, M., Oh, S., and A. Colin de Verdiere, "ODMRP_ASYNC",
[draft-gerla-manet-odmrp-async-00](#) (work in progress).

[I-D.ietf-manet-smf-threats]
Yi, J., Clausen, T., and U. Herberg, "Security Threats for
Simplified Multicast Forwarding (SMF)",
[draft-ietf-manet-smf-threats-00](#) (work in progress),
August 2014.

[ODMRP-Journal]
Lee, S., Su, W., and M. Gerla, "On-Demand Multicast
Routing Protocol in Multihop Wireless Networks",
Journal of Mobile Networks and Applications, Volume 7
Issue 6, Pages 441 - 453, December 2002.

[RFC3561] Perkins, C., Belding-Royer, E., and S. Das, "Ad hoc On-
Demand Distance Vector (AODV) Routing", [RFC 3561](#),
July 2003.

[RFC6130] Clausen, T., Dearlove, C., and J. Dean, "Mobile Ad Hoc
Network (MANET) Neighborhood Discovery Protocol (NHDP)",
[RFC 6130](#), April 2011.

[RFC7181] Clausen, T., Dearlove, C., Jacquet, P., and U. Herberg,
"The Optimized Link State Routing Protocol Version 2",
[RFC 7181](#), April 2014.

[RFC7186] Yi, J., Herberg, U., and T. Clausen, "Security Threats for
the Neighborhood Discovery Protocol (NHDP)", [RFC 7186](#),
April 2014.

Appendix A. Illustrations

This section shows examples of ODMRP 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. This specification does not use or require any packet TLV. Additionally, the minimal packet header required by ODMRP is shown in Figure 1.

```

0
0 1 2 3 4 5 6 7
+---+---+---+---+
| Ver=0 | PF=0 |
+---+---+---+---+

```

Figure 1: Packet Header

A.1. Join Query Message

JQ messages are instances of [RFC5444] messages. This section illustrates an example of one such message.

The JQ message's header's flag octet has a value of 9, meaning that the sequence number and source address fields are present, encoding respectively the sequence number and the address of the multicast source that originated the message. Additionally, the address length field (MAL) is set to 3, corresponding to an address length of 4 octets (i.e., the length of an IPv4 address). The overall message size is 23 octets.

An additional Message-Type specific address block is present, with one address and a flag octet (ABF) having value 0, meaning that the address block has no Head or Tail element. The Mid element encodes the Multicast group address. The associated TLV is of type ADDR-TYPE and value 0, i.e. MULTICAST-GROUP-ADDRESS.

The LastAddress element is omitted, meaning that the last JQ message from this interface was transmitted using the same address as this one.


```

      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
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Join Query   | 1 0 0 1 | MAL=3 |      Message Length = 23      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                     Multicast Source Address      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   Message Sequence Number   |      TLVs length = 0      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Num Addrs = 1 |   ABF = 0   |      Multicast Group      ... |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| ...   Address               | Address TLV Block Length = 3   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| ADDR-TYPE   | 1 0 0 0 0 0 0 0 |      0      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

A.2. Join Reply Message

JR messages are instances of [\[RFC5444\]](#) messages. This section illustrates an example of one such message.

The JR message's header's flag octet has a value of 9, meaning that the sequence number and source address fields are present, encoding respectively the sequence number and the address of the multicast source that originated the message. Additionally, the address length field (MAL) is set to 3, corresponding to an address length of 4 octets (i.e., the length of an IPv4 address). The overall message size is 34 octets.

Two additional Message-Type specific address blocks are present, both with one address and a flag octet (ABF) having value 0, meaning that the address block has no Head or Tail element. For the first address block, the Mid element encodes the Multicast group address; the associated Message-Type-specific TLV is of type ADDR-TYPE and value 0, i.e. MULTICAST-GROUP-ADDRESS. The second address block's Mid element encodes the Next Hop address; its associated Message-Type-specific TLV is of type ADDR-TYPE and value 1, i.e., NEXT-HOP-ADDRESS.


```

      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
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Join Reply   |1 0 0 1| MAL=3 |      Message Length = 34      |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               Multicast Source Address                       |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   Message Sequence Number   |      TLVs length = 0          |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Num Addrs = 1 |   ABF = 0   |      Multicast Group      ...|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| ...   Address               | Address TLV Block Length = 3  |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| ADDR-TYPE   |1 0 0 0 0 0 0 0|   0          | Num Addrs = 1|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|   ABF = 0   |               Next Hop                  ...|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| ... Address | Address TLV Block Length = 3 | ADDR-TYPE   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|1 0 0 0 0 0 0 0|   1          |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Authors' Addresses

Yunjung Yi
University of California, Los Angeles

Sung-Ju Lee
University of California, Los Angeles

William Su
The Boeing Company

Email: william.su@boeing.com

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

Axel Colin de Verdiere
University of California, Los Angeles

Email: axel@axelcdv.com

