On-Demand Multicast Routing Protocol (ODMRP) for Ad Hoc Networks

<draft-ietf-manet-odmrp-01.txt>

Status of This Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026. This document is a submission to the Mobile Ad-hoc Networks (manet) Working Group of the Internet Engineering Task Force (IETF). Comments should be submitted to the Working Group mailing list at "manet@itd.nrl.navy.mil". Distribution of this memo is unlimited.

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

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

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

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

Abstract

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. 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.
Contents

Status of This Memo 1

Abstract 1

1. Introduction 3

2. Terminology 4
   2.1. General Terms ..................................... 4
   2.2. Specification Language ............................... 4

3. Protocol Overview 5
   3.1. Group Establishment and Route Construction .......... 5
       3.1.1. Mesh Creation .................................. 5
       3.1.2. Adapting the Refresh Interval via Mobility Prediction .......... 7
       3.1.3. Soft State ...................................... 7
   3.2. Contents of Tables ................................ 8
       3.2.1. Routing Table .................................. 8
       3.2.2. Forwarding Group Table .......................... 8
       3.2.3. Message Cache .................................. 8
   3.3. Unicast Routing Capability ............................ 8

4. Packet and Table Formats 9
   4.1. Join Data Packet Header ............................... 9
   4.2. Join Table Packet ................................. 11

5. Operation 13
   5.1. Forwarding Group Setup .............................. 13
       5.1.1. Originating a Join Data .......................... 13
       5.1.2. Processing a Join Data .......................... 13
       5.1.3. Processing a Join Data When GPS is Used ......... 14
       5.1.4. Originating a Join Table ........................ 15
       5.1.5. Processing a Join Table ........................ 15
       5.1.6. Processing a Join Table When GPS is Used ....... 16
       5.1.7. Passive Acknowledgments ........................ 17
   5.2. Handling a Multicast Data Packet .................... 17

6. Protocol Applicability 18
   6.1. Networking Context ................................. 18
   6.2. Protocol Characteristics and Mechanisms ............. 18

Acknowledgments 20

References 20

Chair’s Address 21

Authors’ Addresses 22

Lee, Su, and Gerla
1. Introduction

This document describes the On-Demand Multicast Routing Protocol (ODMRP) developed by the Wireless Adaptive Mobility (WAM) Lab [12] at UCLA. ODMRP applies "on-demand" routing techniques to avoid channel overhead and improve scalability. It uses the concept of "forwarding group,"[3] 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. No explicit control message is required to leave the group. We believe the reduction of channel/storage overhead and the relaxed connectivity make ODMRP more scalable for large networks and more stable for mobile wireless networks.

The following properties of ODMRP highlight its advantages.

* Low channel and storage overhead
* Usage of stable routes
* Robustness to host mobility
* Maintenance and exploitation of multiple redundant paths
* Scalability to a large number of nodes
* Exploitation of the broadcast nature of wireless environments
* Adaptivity to node movement patterns
* Reconstruction of routes in anticipation of topology changes
2. Terminology

2.1. General Terms

This section defines terminology used in ODMRP.

node

A device that implements IP.

neighbor

Nodes that are within the radio transmission range.

forwarding group

A group of nodes participating in multicast packet forwarding.

multicast mesh

The topology defined by the link connection between forwarding group members.

join data

The special data packet sent by multicast sources to establish and update group memberships and routes.

join table

The table broadcasted by each multicast receiver and forwarding node to establish and update group membership and routes.

2.2. Specification Language

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [1].
3. Protocol Overview

3.1. Group Establishment and Route Construction

3.1.1. Mesh Creation

In ODMRP, group membership and multicast routes are established and updated by the source on demand. Similar to on-demand unicast routing protocols, a request phase and a reply phase comprise the protocol. When a multicast source has packets to send but no route and group membership is known, it floods a control packet with data payload attached. This packet, called "Join Data" (format shown in Section 4.1) is periodically broadcasted to the entire network to refresh the membership information and update the routes. When a node receives a Join Data packet, it stores the source ID and the sequence number to its "Message Cache" to detect duplicates. The upstream node ID is inserted or updated as the next node for the source node in its "Routing Table." If the Join Data packet is not a duplicate and the Time-To-Live value is greater than zero, appropriate fields are updated and it is rebroadcasted (operation details are explained in Section 5.1.2).

When a Join Data packet reaches the multicast receiver, it creates and broadcasts a "Join Table" to its neighbors. When a node receives a Join Table, it checks if the next node ID of one of the entries matches its own ID. If it does, the node realizes that it is on the path to the source and thus is part of the forwarding group; it sets the FG_FLAG. It then broadcasts its own Join Table built upon matched entries. The next node ID field is filled in by extracting the information from its routing table. This way, the Join Table is propagated by each forward group member until it reaches the multicast source via the selected path. This process constructs (or updates) the routes from sources to receivers and builds a mesh of nodes, the forwarding group.
Let us consider the above figure as an example of Join Table forwarding process. Nodes S1 and S2 are multicast sources, and nodes R1 and R2 are multicast receivers. Node R2 sends its Join Table to both S1 and S2 via I2, and R1 sends its packet to S1 via I1 and to S2 via I2. When receivers send their Join Tables to next hop nodes, an intermediate node I1 sets the FG_FLAG and builds its own Join Table since there is a next node ID entry in the Join Table received from R1 that matches its ID. Note that the Join Table build by I1 has an entry for sender S1 but not for S2 because the next node ID for S2 in the received Join table is not I1. In the meanwhile, node I2 sets the FG_FLAG, constructs its own Join Table and sends it to its neighbors. Note that I2 broadcasts the join table only once even though it receives two Join Tables from the receivers because the second table arrival carries no new source information. Channel overhead is thus reduced dramatically in cases where numerous multicast receivers share the same links to the source.

After this group establishment and route construction process, a source can multicast packets to receivers via selected routes and forwarding groups. While outgoing data packets exist, the source sends Join Data every REFRESH_INTERVAL. This Join Data and Join Table propagation process refreshes forwarding group and routes. When receiving the multicast data packet, a node forwards it only when it is not a duplicate and the setting of the FG_FLAG for the multicast group has not expired. This procedure minimizes the traffic overhead and prevents sending packets through stale routes.
3.1.2 Adapting the Refresh Interval via Mobility Prediction

ODMRP requires periodic flooding of Join Data to build and refresh routes. Excessive flooding, however, is not desirable in ad hoc networks because of bandwidth constraints. Furthermore, flooding often causes congestion, contention, and collisions. Finding the optimal flooding interval is critical in ODMRP performance. In highly mobile networks where nodes are equipped with GPS [9] (e.g., tactical networks with tanks, ships, aircrafts, etc.), we can efficiently adapt the REFRESH_INTERVAL to mobility patterns and speeds by utilizing the location and movement information. Note that ODMRP can still operate efficiently in networks where no such information is available, but the protocol can be further improved if those information can be utilized.

We use the location and movement information to predict the duration of time routes will remain valid (the detail of the process is explained in 5.1.3). With the predicted time of route disconnection, Join Data are only flooded when route breaks of ongoing data sessions are imminent.

A different route selection method is applied when we use the mobility prediction. The idea is inspired by the Associativity-Based Routing (ABR) protocol [11] which chooses associatively stable routes. In our algorithm, instead of using the minimum delay path, we can choose a route that is the most stable (i.e., the one that will remain connected for the longest duration of time). To select a route, a multicast receiver must wait for an appropriate amount of time after receiving the first Join Data so that all possible routes and their route qualities will be known. The receiver then chooses the most stable route and broadcasts a Join Table. Route breaks will occur less often and the number of Join Data propagation will reduce because stable routes are used.

3.1.3. Soft State

In ODMRP, no explicit control packets need to be sent to leave the group. If a multicast source wants to leave the group, it simply stops sending any Join Data packets since it does not have any multicast data to send to the group. If a receiver no longer wants to receive from a particular multicast group, it does not send the Join Table for that group. Nodes in the forwarding group are demoted to non-forwarding nodes if not refreshed (no Join Tables received) before they timeout.
3.2. Contents of Tables

Nodes running ODMRP are required to maintain the following tables. These tables MAY be implemented in any format, but MUST include the fields specified in this document.

3.2.1. Routing Table

A routing table is created on demand and is maintained by each node. An entry is inserted or updated when a non-duplicate Join Data is received. The node stores the destination (i.e., the source of the Join Data) and the next hop to the destination (i.e., the last node that propagated the Join Data). The routing table provides the next hop information when transmitting Join Tables.

3.2.2. Forwarding Group Table

When a node is a forwarding group node of the multicast group, it maintains the group information in the forwarding group table. The multicast group ID and the time when the node was last refreshed are recorded.

3.2.3. Message Cache

The message cache is maintained by each node to detect duplicates. When a node receives a new Join Data or data, it stores the source address and the sequence number of the packet. Note that entries in the message cache need not be maintained permanently. Schemes such as LRU (Least Recently Used) or FIFO (First In First Out) can be employed to expire and remove old entries and prevent the size of the message cache to be extensive.

3.3. Unicast Routing Capability

One of the major strengths of ODMRP is its unicast routing capability. Not only ODMRP can work with any unicast routing protocol, it can function as both multicast and unicast. Thus, ODMRP can run without any underlying unicast protocol. Other ad hoc multicast routing protocols such as AMRoute [2], CAMP [5], R&B [4], and LAM [7] must be run on top of a unicast routing protocol. CAMP, R&B, and LAM in particular, only work on top of certain underlying unicast protocols.
4. Packet and Table Formats

4.1. Join Data Packet Header

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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Type     |   Reserved    |  Time To Live |   Hop Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Multicast Group IP Address                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Sequence Number                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Source IP Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Previous Hop IP Address                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Previous Hop X Coordinate               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Previous Hop Y Coordinate               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Previous Hop Moving Speed   | Previous Hop Moving Direction |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Minimum Link Expiration Time           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

Type

01; ODMRP Join Data.

Reserved

Sent as 0; ignored on reception.

Time To Live

Number of hops this packet can traverse.

Hop Count

The number of hops traveled so far by this packet.

Multicast Group IP Address

The IP address of the multicast group.
Sequence Number

The sequence number assigned by the source to uniquely identify the packet.

Source IP Address

The IP address of the node originating the packet.

Previous Hop IP Address

The IP address of the last node that has processed this packet.

Previous Hop X Coordinate (Optional)

The x-coordinate of the last node that has processed this packet. The information can be obtained from the GPS. This field is required only when network hosts are GPS equipped.

Previous Hop Y Coordinate (Optional)

The y-coordinate of the last node that has processed this packet. The information can be obtained from the GPS. This field is required only when network hosts are GPS equipped.

Previous Hop Moving Speed (Optional)

The mobility speed of the last node that has processed this packet. The information can be obtained from the GPS or the node’s own instruments and sensors (e.g., campus, odometer, speed sensors, etc.). This field is required only when network hosts are GPS equipped.

Previous Hop Moving Direction (Optional)

The moving direction of the last node that has processed this packet. The information can be obtained from the GPS or the node’s own instruments and sensors (e.g., campus, odometer, speed sensors, etc.). This field is required only when network hosts are GPS equipped.

Minimum Link Expiration Time (Optional)

The minimum expiration time among the links taken by this packet so far. This field is required only when network hosts are GPS equipped.
### 4.2. Join Table Packet

<table>
<thead>
<tr>
<th>Type</th>
<th>Count</th>
<th>R</th>
<th>F</th>
<th>Reserved</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Multicast Group IP Address</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Previous Hop IP Address</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Sequence Number</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Sender IP Address [1]</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Next Hop IP Address [1]</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Route Expiration Time [1]</th>
</tr>
</thead>
</table>

| : | : | : | : | : | : | : |

<table>
<thead>
<tr>
<th>Source IP Address [n]</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Next Hop IP Address [n]</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Route Expiration Time [n]</th>
</tr>
</thead>
</table>

Reserved

    Sent as 0; ignored on reception.

Multicast Group IP address

    The IP address of the multicast group.

Previous Hop IP Address

    The IP address of the last node that has processed this packet.

Sequence Number

    The sequence number assigned by the previous hop node to
    uniquely identify the packet.

Sender IP Address [1..n]

    The IP addresses of the sources of this multicast group.

Next Hop IP Address [1..n]

    The IP addresses of next nodes that this packet is target to.

Route Expiration Time [1..n] (Optional)

    The minimum route expiration times of this multicast group.
    This field is required only when network hosts are GPS equipped.
5. Operation

5.1. Forwarding Group Setup

5.1.1. Originating a Join Data

When a multicast source has data packets to send but no route is known, it originates a "Join Data" packet. The Type field MUST be set to 01. TTL MAY be set to TIME_TO_LIVE_VALUE, but SHOULD be adjusted based on network size and network diameter. The Sequence number MUST be large enough to prevent wraparound ambiguity, and the Hop Count is initially set to zero. The source puts its IP address in the Source IP Address and Last Hop IP Address field. It appends its location, speed, and direction into JOIN DATA.

When location and movement information is utilized, it sets the MIN_LET (Link Expiration Time) field to the MAX_LET_VALUE since the source does not have any previous hop node. When the source receives Join Tables from multicast receivers, it selects the minimum RET (Route Expiration Time) among all the Join Tables received. Then the source can build new routes by originating a Join Data before the minimum RET approaches (i.e., route breaks of ongoing data sessions are imminent).

5.1.2. Processing a Join Data

When a node receives a Join Data packet:

1. Check if it is a duplicate by comparing the (Source IP Address, Sequence Number) combination with the entries in message cache. If duplicate, then discard the packet. DONE.

2. If it is not a duplicate, insert an entry into message cache with the information of the received packet (i.e., sequence number and source IP address) and insert/update the entry for routing table (i.e., backward learning).

3. If the node is a member of the multicast group, it originates a Join Table packet with the RET value enclosed (see Section 5.1.4).

4. Increase the Hop Count field by 1 and decrease the TTL field by 1.

5. If the TTL field value is less than or equal to 0, then discard the packet. DONE.

6. If the TTL field value is greater than 0, then set the node’s IP Address into Last Hop IP Address field and broadcast. DONE.
5.1.3. Processing a Join Data When GPS is Used

When a node receives a Join Data packet:

1. Check if it is a duplicate by comparing the (Source IP Address, Sequence Number) combination with the entries in message cache. If duplicate, then discard the packet. DONE.

2. If it is not a duplicate, insert an entry into message cache with the information of the received packet (i.e., sequence number and source IP address) and insert/update the entry for routing table (i.e., backward learning).

3. Predict the duration of time the link between the node and the upstream node will remain connected using the following equation.

Assume node i is the upstream node and node j is the current node. Let \((x_{i}, y_{i})\) be the coordinate of node i and \((x_{j}, y_{j})\) be that of node j. Also let \(v_{i}\) and \(v_{j}\) be the speeds, and \(\theta_{i}\) and \(\theta_{j}\) be the moving directions of nodes i and j, respectively. The information of node i (the previous hop node) can be obtained from the Join Data and the current node’s location and mobility information can be provided by the GPS. The duration of time that the link between two nodes will stay connected, \(D_{t}\), is given by:

\[
D_{t} = \frac{-(a*b + c*d) + \sqrt{(a^{2} + c^{2})*r^{2} - (a*d - b*c)^{2}}}{a^{2} + c^{2}}
\]

where

\[
\begin{align*}
    a &= v_{i} \cdot \cos(\theta_{i}) - v_{j} \cdot \cos(\theta_{j}), \\
    b &= x_{i} - x_{j}, \\
    c &= v_{i} \cdot \sin(\theta_{i}) - v_{j} \cdot \sin(\theta_{j}), \\
    d &= y_{i} - y_{j}.
\end{align*}
\]

Note that when \(v_{i} = v_{j}\) and \(\theta_{i} = \theta_{j}\), \(D_{t}\) is set to infinity without applying the above equation.

The minimum between this \(D_{t}\) value and the indicated value in MIN_LET field of the Join Data is included in the packet. The rationale is that as soon as a single link on the path is disconnected, the entire path is invalidated. The node also overwrites the location and mobility information field written by the previous node with its own information.
4. If the node is a member of the multicast group, it calculates the predicted LET of the last link of the path. The minimum between the last link expiration time and the MIN_LET value specified in the Join Data is the RET (Route Expiration Time).

To select a route, a multicast receiver must wait for an appropriate amount of time after receiving the first Join Data so that all possible routes and their RET will be known. The receiver then chooses the most stable route (i.e., the route with the largest RET) and originates a Join Table packet with the RET value enclosed (see Section 5.1.3).

5. Increase the Hop Count field by 1 and decrease the TTL field by 1.

6. If the TTL field value is less than or equal to 0, then discard the packet. DONE.

7. If the TTL field value is greater than 0, then set the node’s IP Address into Last Hop IP Address field and broadcast. DONE.

5.1.4. Originating a Join Table

A multicast receiver transmits a "Join Table" packet after selecting the multicast route. Each sender IP address and next hop IP address of a multicast group are contained in the Join Table packet. The route expiration time is also included if the network hosts operate with GPS.

5.1.5. Processing a Join Table

When a Join Table is received:

1. The node looks up the Next Hop IP Address field of the received Join Table entries. If no entries match the node’s IP Address, do nothing. DONE.

2. If one or more entries coincide with the node’s IP Address, set the FG_FLAG and build its own Join Table. The next hop IP address can be obtained from the routing table.

3. Broadcast the Join Table packet to the neighbor nodes. DONE.
5.1.6. Processing a Join Table When GPS is Used

When a Join Table is received:

1. The node looks up the Next Hop IP Address field of the received Join Table entries. If no entries match the node’s IP Address, do nothing. DONE.

2. If one or more entries coincide with the node’s IP Address, set the FG_FLAG and build its own Join Table. If multiple Join Tables with different RET values are received (i.e., the node lies in paths from the same source to multiple receivers), it selects the minimum RET among them and attaches the chosen RET value. Next hop IP address can be obtained from the routing table.

3. Broadcast the Join Table packet to the neighbor nodes.

4. If the node is a source, it selects the minimum RET among all the Join Tables received. Then the source can build new routes by flooding a Join Data before the minimum RET approaches (i.e., route breaks of ongoing data sessions are imminent).

In addition to the estimated RET value, other factors need to be considered when choosing the refresh interval of Join Data. If the node mobility rate is high and the topology changes frequently, routes will expire quickly and often. The source may propagate Join Requests excessively and this excessive flooding can cause collisions, congestion, and clogs the network with control packets. Thus, the MIN_REFRESH_INTERVAL should be enforced to avoid control message overflow. On the other hand, if nodes are stationary or move slowly and link connectivity remains unchanged for a long duration of time, routes will hardly expire and the source will rarely send Join Data. A few problems arise in this situation. First, if a node in the route suddenly changes its movement direction or speed, the predicted RET value becomes obsolete and routes will not be reconstructed. Second, when a non-member node which is located remotely to multicast members wants to join the group, it cannot inform the new membership or receive data until a Join Data is received. Hence, the MAX_REFRESH_INTERVAL should be set. The selection of the MIN_REFRESH_INTERVAL and the MAX_REFRESH_INTERVAL should be adaptive to network situations (e.g., traffic type, traffic load, mobility pattern, channel capacity, etc.).
5.1.7. Passive Acknowledgments

The reliable transmission of Join Tables plays an important role in establishing and refreshing multicast routes and forwarding groups. Hence, if Join Tables are not properly delivered, effective multicast routing cannot be achieved by ODMRP. The IEEE 802.11 MAC protocol [6], which is the standard in wireless networks, performs reliable transmission by retransmitting the packet if no acknowledgment is received. However, if the packet is broadcasted, the acknowledgments and retransmissions are not sent. In ODMRP, the transmission of Join Tables are mostly broadcasted. Thus, the verification of Join Table delivery and the retransmissions must be done by the ODMRP layer.

We adopt a scheme that was used in [8]. When a node transmits a Join Table packet to the immediate upstream node of the route, the immediate downstream node can hear the transmission if it is within the transmitter’s radio range. Hence, the packet is used as an "passive acknowledgment." We can utilize this passive acknowledgments to verify the delivery of Join Tables. Multicast sources must send active acknowledgments to the previous hops since they do not have any next hops to send Join Tables to unless they are forwarding group nodes. When no acknowledgment is received within the timeout interval, the node retransmits the message. If packet delivery cannot be verified after an appropriate number of retransmissions, the node considers the route to be invalidated. The node then broadcasts a message to its neighbors specifying that the next hop to the source cannot be reached. Upon receiving this packet, the neighboring node builds and unicasts the Join Table to its next hop if it has a route to the multicast source. If no route is known, it simply rebroadcasts the packet specifying the next hop is not available. In both cases, the node sets its FG_FLAG. The FG_FLAG setting of every neighbors may create excessive redundancy, but most of these settings will expire because only necessary forwarding group nodes will be refreshed in the next Join Table propagation phase.

5.2. Handling a Multicast Data Packet

Multicast sources send the Data whenever they have packets to send. Nodes relay data packets only if the packet is not a duplicate and the setting of FG_FLAG for the multicast group has not expired.
6. Protocol Applicability

6.1. Networking Context

ODMRP is best suited for mobile ad hoc wireless networks.

6.2. Protocol Characteristics and Mechanisms

* Does the protocol provide support for unidirectional links? (if so, how?)
  - No. We assume bidirectional links.

* Does the protocol require the use of tunneling? (if so, how?)
  - No.

* Does the protocol require using some form of source routing? (if so, how?)
  - No.

* Does the protocol require the use of periodic messaging? (if so, how?)
  - No.

* Does the protocol require the use of reliable or sequenced packet delivery? (if so, how?)
  - No.

* Does the protocol provide support for routing through a multi-technology routing fabric? (if so, how?)
  - No.

* Does the protocol provide support for multiple hosts per router? (if so, how?)
  - No. In this document, we assume each mobile host is combined with a router, sharing the same IP address. It is possible, however, to extend the protocol to handle multiple hosts per router.
* Does the protocol support the IP addressing architecture? (if so, how?)
  - Yes. The message contains host IP address as its identification.

* Does the protocol require link or neighbor status sensing (if so, how?)
  - No.

* Does the protocol have dependence on a central entity? (if so, how?)
  - No.

* Does the protocol function reactively? (if so, how?)
  - Yes. For example, the source creates and maintains routes and multicast group membership only when it has data packets to send.

* Does the protocol function proactively? (if so, how?)
  - No.

* Does the protocol provide loop-free routing? (if so, how?)
  - Yes. By using the Message Cache, duplicate packets are detected and packets can only go through the loop-free route.

* Does the protocol provide for sleep period operation? (if so, how?)
  - TBD. The work is in progress.

* Does the protocol provide some form of security? (if so, how?)
  - TBD. The work is in progress.

* Does the protocol provide support for utilizing multi-channel, link-layer technologies? (if so, how?)
  - This document assumed an arbitrary single channel link-layer protocol. The protocol can work with any MAC and link-layer technology. It can also support multi-channel link-layer technology (e.g., separate channels for data, control packets, etc.).
Acknowledgments

Authors thank Ching-Chuan Chiang and Guangyu Pei for their initial contributions.

References


Chair's Address

The Working Group can be contacted via its current chairs:

M. Scott Corson
Institute for Systems Research
University of Maryland
College Park, MD 20742
USA

Phone: +1 301 405-6630
Email: corson@isr.umd.edu

Joseph Macker
Information Technology Division
Naval Research Laboratory
Washington, DC 20375
USA

Phone: +1 202 767-2001
Email: macker@itd.nrl.navy.mil
Authors’ Addresses

Questions about this document can also be directed to the authors:

Sung-Ju Lee
3771 Boelter Hall
Computer Science Department
University of California
Los Angeles, CA  90095-1596
USA
Phone:  +1 310 206-8589
Fax:    +1 310 825-7578
Email:  sjlee@cs.ucla.edu

William Su
3771 Boelter Hall
Computer Science Department
University of California
Los Angeles, CA  90095-1596
USA
Phone:  +1 310 206-8589
Fax:    +1 310 825-7578
Email:  wsu@cs.ucla.edu

Mario Gerla
3732F Boelter Hall
Computer Science Department
University of California
Los Angeles, CA  90095-1596
USA
Phone:  +1 310 825-4367
Fax:    +1 310 825-7578
Email:  gerla@cs.ucla.edu