Mobile Ad Hoc Networking Working Group — Charles E. Perkins
INTERNET DRAFT — Nokia Research Center
22 October 1999 — Elizabeth M. Royer
University of California, Santa Barbara
Samir R. Das
University of Texas, San Antonio

Ad Hoc On-Demand Distance Vector (AODV) Routing
draft-ietf-manet-aodv-04.txt

Status of This Memo

This document is a submission by the Mobile Ad Hoc Networking Working Group of the Internet Engineering Task Force (IETF). Comments should be submitted to the manet@itd.nrl.navy.mil mailing list.

Distribution of this memo is unlimited.

Abstract

The Ad Hoc On-Demand Distance Vector (AODV) routing protocol is intended for use by mobile nodes in an ad hoc network. It offers quick adaptation to dynamic link conditions, low processing and memory overhead, low network utilization, and determines both unicast

and multicast routes between sources and destinations. It uses destination sequence numbers to ensure loop freedom at all times (even in the face of anomalous delivery of routing control messages), solving problems (such as ''counting to infinity'') associated with classical distance vector protocols.

Contents

1. Introduction

The Ad Hoc On-Demand Distance Vector (AODV) algorithm enables dynamic, self-starting, multihop routing between participating mobile nodes wishing to establish and maintain an ad hoc network. AODV allows mobile nodes to obtain routes quickly for new destinations, and does not require nodes to maintain routes to destinations that are not in active communication. Additionally, AODV allows for the formation of multicast groups whose membership is free to change during the lifetime of the network. AODV allows mobile nodes to respond quickly to link breakages and changes in network topology. The operation of AODV is loop-free, and by avoiding the Bellman-Ford ''counting to infinity'' problem offers quick convergence when the ad hoc network topology changes (typically, when a node moves in the network).

One distinguishing feature of AODV is its use of a destination sequence number for each route entry. The destination sequence number is created by the destination or the multicast group leader for any route information it sends to requesting nodes. Using destination sequence numbers ensures loop freedom and is simple to program. Given the choice between two routes to a destination, a requesting node always selects the one with the greatest sequence number.

Another feature of AODV is that link breakages cause immediate notifications to be sent to the affected set of nodes.

2. Overview

Route Requests (RREQs), Route Replies (RREPs), Route Errors (RERRs), Multicast Activations (MACTs), and Group Hellos (GRPHs) are the message types defined by AODV. These message types are handled by UDP, and normal IP header processing applies. So, for instance, the requesting node is expected to use its IP address as the source IP address for the messages. The range of dissemination of broadcast RREQs can be indicated by the TTL in the IP header. Fragmentation is typically not required.

As long as the endpoints of a communication connection have valid routes to each other, AODV does not play any role. When a route to a new destination (either a single node or a multicast group) is needed, the node uses a broadcast RREQ to find a route to the destination. A route can be determined when the RREQ reaches either the destination itself, or an intermediate node with a 'fresh enough' route to the destination. A 'fresh enough' route is an unexpired route entry for the destination whose associated sequence number is at least as great as that contained in the RREQ. The route is made available by unicasting a RREP back to the source of the RREQ. Since each node receiving the request caches a route back to the source of the request, the RREP can be unicast back from the destination to the source, or from any intermediate node that is able to satisfy the request back to the source. A RREQ can be conditioned by requirements on the path to the destination, namely bandwidth or delay bounds.

Nodes monitor the link status of next hops in active routes. When a link break in an active route is detected, a RERR message is used to notify other nodes that the loss of that link has occured. The RERR message indicates which destinations are now unreachable due to the loss of the link.

RREQs are also used when a node wishes to join a multicast group. A join flag in the RREQ informs nodes that when receiving the RREP, they are not just setting route pointers but are also setting multicast route pointers, which will be used if the route is selected to be added onto the tree. If the route is chosen for addition to the multicast tree, it will be activated by a MACT message.

For multicast groups, a Group Hello message is periodically broadcast across the network by the multicast group leader. The message carries multicast group and corresponding group leader IP addresses. This information is used for repairing multicast trees after a previously disconnected portion of the network containing part of the multicast tree becomes reachable once again.

Since AODV is a routing protocol, it deals with route table management. Route table information must be kept even for ephemeral routes, such as are created to temporarily store reverse paths towards nodes originating RREQs. AODV uses the following fields with each route table entry:

- Destination IP Address
- Destination Sequence Number
- Hop Count (number of hops needed to reach destination)
- Last Hop Count (described in subsection 9.2.1)
- Next Hop
- List of Precursors (described in Section 9.1)
- Lifetime (expiration time of the route)

- Routing Flags

The following information is stored in each entry of the multicast route table for multicast tree routes:

- Multicast Group IP Address
- Multicast Group Leader IP Address
- Multicast Group Sequence Number
- Next Hops
- Hop Count to next Multicast Group Member
- Hop Count to Multicast Group Leader

The Next Hops field is a linked list of structures, each of which contains the following fields:

- Next Hop IP Address
- Link Direction
- Activated Flag

The direction of the link is relative to the location of the group leader, i.e. UPSTREAM is a next hop towards the group leader, and DOWNSTREAM is a next hop away from the group leader. A node on the multicast tree must necessarily have only one UPSTREAM link. The IP Address of a Next Hop MUST NOT be used to forward multicast messages until after a MACT message has activated the route (see Section 10.6).

3. AODV Terminology

This protocol specification uses conventional meanings [1] for capitalized words such as MUST, SHOULD, etc., to indicate requirement levels for various protocol features. This section defines other terminology used with AODV that is not already defined in [2].

active route

A routing table entry with an unexpired Lifetime and a finite metric in the Hop Count field. A routing table may contain entries that are not active. Only active entries can be used to forward data packets.

forwarding node

A node which agrees to forward packets destined for another destination node, by retransmitting them to a next hop which is closer to the unicast destination along a path which has been set up using routing control messages.

group leader

A node which is a member of the given multicast group and which is typically the first such group member in the connected portion of the network. This node is responsible for initializing and maintaining the multicast group destination sequence number.

group leader table

The table where ad hoc nodes keep information concerning each multicast group and its corresponding group leader. There is one entry in the table for each multicast group for which the node has received a Group Hello (see Section 10.2).

multicast tree

The tree containing all nodes which are members of the multicast group and all nodes which are needed to connect the multicast group members.

multicast route table

The table where ad hoc nodes keep routing (including next hops) information for various multicast groups.

subnet leader

A node which is a member of the subnet defined by a specific routing prefix, and which offers reachability to every other node with the same routing prefix. The subnet leader is responsible for initializing and maintaining the destination sequence number for every node on the subnet.

4. Route Request (RREQ) Message Format

```
 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      |J|R|         Reserved            |   Hop Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Broadcast ID                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination IP address                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Destination Sequence Number                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source IP address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Source Sequence Number                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format of the Route Request message is illustrated above, and contains the following fields:

Type 1

J Join flag; set when source node wants to join a multicast group.

R Repair flag; set when a node wants to initiate a repair to connect two previously disconnected portions of the multicast tree.

Reserved Sent as 0; ignored on reception.

Hop Count
: The number of hops from the Source IP Address to the node handling the request.

Broadcast ID
: A sequence number uniquely identifying the particular RREQ when taken in conjunction with the source node's IP address.

Destination IP Address
: The IP address of destination for which a route is desired.

Destination Sequence Number
: The last sequence number received in the past by the source for any route towards the destination.

Source IP Address
: The IP address of the node which originated the Route Request.

Source Sequence Number
: The current sequence number to be used for route entries pointing to (and generated by) the source of the route request.

When a node wishes to repair a multicast tree, it appends the Multicast Group Rebuild extension (see Section 15.3). When a node wishes to unicast the RREQ for a multicast group to the group leader, it includes the Multicast Group Leader extension (see Section 15.2).

5. Route Reply (RREP) Message Format

```
 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      |R|       Reserved      | Prefix  |  Hop Count  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Destination IP address                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Destination Sequence Number                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source IP address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Lifetime                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format of the Route Reply message is illustrated above, and contains the following fields:

Type 2

R Repair flag; set when a node is responding to a repair request to connect two previously disconnected portions of the multicast tree.

Reserved Sent as 0; ignored on reception.

Prefix If nonzero, the Prefix Size specifies that the indicated next hop may be used for any nodes with the same routing prefix (as defined by the Prefix Size) as the requested destination.

Hop Count The number of hops from the Source IP Address to the Destination IP Address. For multicast route requests this indicates the number of hops to the multicast tree member sending the RREP.

Destination IP Address
The IP address of the destination for which a route is supplied.

Destination Sequence Number
The destination sequence number associated to the route.

Source IP Address
The IP address of the source node which issued the RREQ for which the route is supplied.

Lifetime
The time for which nodes receiving the RREP consider the route to be valid.

When the RREP is sent for a multicast destination, the Multicast Group Information extension is appended (see Section 15.4).

Note that the Prefix Size allows a Subnet Leader to supply a route for every host in the subnet defined by the routing prefix, which is determined by the IP address of the Subnet Leader and the Prefix Size. In order to make use of this feature, the Subnet Leader has to guarantee reachability to all the hosts sharing the indicated subnet prefix. The Subnet Leader is also responsible for maintaining the Destination Sequence Number for the whole subnet.

6. Route Error (RERR) Message Format

```
 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           |    Length     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Unreachable Destination IP Address (1)            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Additional Unreachable Destination IP Addresses (if needed...) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format of the Route Error message is illustrated above, and contains the following fields:

Type   3

Reserved Sent as 0; ignored on reception.

Length The number of unreachable destinations included in the message. Must be at least 1.

Unreachable Destination IP Address
The IP address of the destination which has become unreachable due to a link break.

The RERR message is sent whenever a link break causes one or more destinations to become unreachable. The unreachable destination addresses included are those of all lost destinations which are now unreachable due to the loss of that link.

7. Multicast Activation (MACT) Message Format

```
 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      |P|G|U|       Reserved          |   Hop Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Multicast Group IP address                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Source IP address                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Source Sequence Number                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format of the Multicast Activation message is illustrated above, and contains the following fields:

Type 4

P Prune flag; set when a node wishes to prune itself from the tree, unset when the node is activating a tree link.

G Group Leader flag; set by a multicast tree member that fails to repair a multicast tree link breakage, and indicates to the group member receiving the message that it should become the new multicast group leader.

U — Update flag; set when a multicast tree member has repaired a broken tree link and is now a new distance from the group leader.

Reserved — Sent as 0; ignored on reception.

Hop Count — The distance of the sending node from the multicast group leader. Used only when the 'U' flag is set; otherwise sent as 0.

Multicast Group IP Address
The IP address of the Multicast Group for which a route is supplied.

Source IP Address
The IP address of the sending node.

Source Sequence Number
The current sequence number for route information generated by the source of the route request.

To prune itself from the tree (i.e., inactivate its last link to the multicast tree), a multicast tree member sends a MACT with the 'P' flag = 1 to its next hop on the multicast tree. A multicast tree member that has more than one next hop to the multicast tree SHOULD NOT prune itself from the multicast tree.

8. Group Hello (GRPH) Message Format

```
 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      |U|M|       Reserved              |   Hop Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Group Leader IP address                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Multicast Group IP address                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Multicast Group Sequence Number                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The format of the Group Hello message is illustrated above, and contains the following fields:

Type 5

U Update flag; set when there has been a change in group leader information.

M Off_Mtree flag; set by a node receiving the group hello that is not on the multicast tree.

Reserved Sent as 0; ignored on reception.

Hop Count The number of hops the packet has traveled. Used by multicast tree nodes to update their distance from the group leader when the M flag is not set.

Group Leader IP Address
The IP address of the group leader.

Multicast Group IP Address
The IP address of the Multicast Group for which the sequence number is supplied.

Multicast Group Sequence Number
The current sequence number of the multicast group.

9. Node Operation - Unicast

This section describes the scenarios under which nodes generate RREQs and RREPs for unicast communication, and how the fields in the message are handled.

9.1. Maintaining Route Utilization Records

For each valid route maintained by a node (containing a finite Hop Count metric) as a routing table entry, the node also maintains a list of precursors that may be forwarding packets on this route. These precursors will receive notifications from the node in the event of detection of the loss of the next hop link. The list of precursors in a routing table entry contains those neighboring nodes to which a route reply was generated or forwarded.

Each time a route is used to forward a data packet, its Lifetime field is updated to be current time plus ACTIVE_ROUTE_TIMEOUT.

9.2. Generating Route Requests

A node broadcasts a RREQ when it determines that it needs a route to a destination and does not have one available. This can happen if the destination is previously unknown to the node, or if a previously valid route to the destination expires or is broken (i.e., an infinite metric is associated with the route). When a route table entry is marked with an infinite metric, its Lifetime is also updated to be the current time plus BAD_LINK_LIFETIME milliseconds. After the Lifetime expires, the route MAY be expunged from the node's route table.

After broadcasting a RREQ, a node waits for a RREP. If the RREP is not received within RREP_WAIT_TIME milliseconds, the node MAY rebroadcast the RREQ, up to a maximum of RREQ_RETRIES times. Each rebroadcast MUST increment the Broadcast ID field.

Data packets waiting for a route (i.e., waiting for a RREP after RREQ has been sent) SHOULD be buffered. The buffering SHOULD be FIFO. If a RREQ has been rebroadcast RREQ_RETRIES times without receiving any RREP, all data packets destined for the corresponding destination SHOULD be dropped from the buffer.

9.2.1. Controlling Route Request Broadcasts

To prevent unnecessary network-wide broadcasts of RREQs, the source node SHOULD use an expanding ring search technique as an optimization. In an expanding ring search, the source node initially uses a TTL = TTL_START in the RREQ packet IP header and sets the timeout for receiving a RREP to 2 * TTL * NODE_TRAVERSAL_TIME milliseconds. Upon timeout, the source rebroadcasts the RREQ with the TTL incremented by TTL_INCREMENT. This continues until the TTL set in the RREQ reaches TTL_THRESHOLD, beyond which a TTL = NET_DIAMETER is used for each rebroadcast. Each time, the timeout for receiving a RREP is calculated as before. Each rebroadcast increments the Broadcast ID field in the RREQ packet. The RREQ can be rebroadcast with TTL = NET_DIAMETER up to a maximum of RREQ_RETRIES times.

When a RREP is received, the Hop Count used in the RREP packet is remembered as Last Hop Count in the routing table. When a new route to the same destination is required at a later time (e.g., upon route loss), the TTL in the RREQ IP header is initially set to this Last Hop Count plus TTL_INCREMENT. Thereafter, following each timeout the TTL is incremented by TTL_INCREMENT until TTL = TTL_THRESHOLD is reached. Beyond this TTL = NET_DIAMETER is used as before.

As a further optimization, timeouts MAY be determined dynamically via measurements, instead of using a statically configured value related to NODE_TRAVERSAL_TIME. To accomplish this, the RREQ may carry the timestamp via an extension field as defined in Section 15 to be carried back by the RREP packet (again via an extension field). The difference between the current time and this timestamp will determine the route discovery latency. The timeout may be set to be a small factor of the average of the last few route discovery latencies for the concerned destination. These latencies may be recorded as additional fields in the routing table.

If the optimizations described in this section are used, an expired routing table entry should not be expunged too early. Otherwise, the soft states corresponding to the route (e.g., Last Hop Count) will be lost. In such cases, a longer routing table entry expunge time may be specified. In general, any routing table entry waiting for a RREP should not be expunged before the timeout for receiving RREP.

9.3. Forwarding Route Requests

When a node receives a broadcast RREQ, it first checks to see whether it has received a RREQ with the same Source IP Address and a Broadcast ID field of equal unsigned integer value within the last BCAST_ID_SAVE milliseconds. If such a RREQ has been received, the node silently discards the newly received RREQ. The rest of this subsection describes actions taken for RREQs that are not discarded.

9.3.1. Processing Route Requests

When a node receives a RREQ, the node checks to see whether it has a route to the destination. If the node does not have a route, it rebroadcasts the RREQ from its interface(s) but using its own IP address in the IP header of the outgoing RREQ. The TTL or hop limit field in the outgoing IP header is decreased by one. The Hop Count field in the broadcast RREQ message is incremented by one, to account for the new hop through the intermediate node. The node also creates or updates a reverse route to the Source IP Address in its routing table with next hop equal to the IP address of the neighboring node that sent the broadcast RREQ (often not equal to the Source IP Address field in the RREQ message). This reverse route might be used for an eventual RREP back to the node which originated the RREQ (identified by the Source IP Address). If no route exists for the Source IP Address, or if an existing route will expire too soon, the reverse route is put into the route table with lifetime REV_ROUTE_LIFE milliseconds.

If, on the other hand, the node does have the requested route, it compares the destination sequence number (dest-seqno) for that route with the Destination Sequence Number field of the incoming RREQ. If the node's existing dest-seqno is smaller than the Destination Sequence Number field of the RREQ, the node again rebroadcasts the RREQ just as if it did not have a route to the destination at all.

If the node has a route to the destination, and the node's existing dest-seqno is greater than or equal to the Destination Sequence Number of the RREQ, then the node generates a RREP as discussed further in section 9.4.

9.4. Generating Route Replies

If a node receives a route request for a destination, and has a fresh enough route to satisfy the request, the node generates a RREP message and unicasts it back to the node indicated by the Source IP Address field of the received RREQ. If the node is not the destination node, it copies over the destination sequence number from the route table entry. If the generating node is the destination itself, it uses a destination sequence number at least equal to a sequence number generated after the last detected change in its neighbor set and at least equal to the destination sequence number in the RREQ. If the destination node has not detected any change in its set of neighbors since it last incremented its destination sequence number, it MAY use the same destination sequence number.

As part of the process of generating the RREP, the generating node creates or updates an entry in its routing table for the Source IP Address, if necessary as described in section 9.3. The Source Sequence Number is put into the route entry, along with the Hop Count from the RREQ. The Lifetime for the route table entry is set to the current time plus ACTIVE_ROUTE_TIMEOUT milliseconds.

If the generating node is not the destination node, then the generating node places its distance in hops from the destination in the Hop Count field. If the generating node is the destination node, it places the value zero in the Hop Count field. The Hop Count field is incremented by one at each hop as the RREP is forwarded to the source. When the RREP reaches the source, the Hop Count will represent the distance, in hops, of the destination from the source.

If the node is not the destination node, it calculates the Lifetime field of the RREP by subtracting the current time from the expiration time in its route table entry. Otherwise, if the generating node is also the destination node, it copies the value MY_ROUTE_TIMEOUT into the Lifetime field of the RREP. Each node MAY make a separate determination about its value MY_ROUTE_TIMEOUT.

If the generating node is not the node indicated by the Destination IP Address, then it puts the next hop towards the destination in the precursor list for the reverse path route entry. In addition, the generating node puts the last hop node (from which it received the RREQ, as indicated by the source IP address field in the IP header) into the precursor list for the forward path towards the destination.

9.5. Forwarding Route Replies

When a node receives a RREP message, it first increments the Hop Count field in the RREP. If this node is the node indicated in the Source IP Address field, it then creates a forward route entry to the node indicated by the Destination IP Address field, using the node from which it received the RREP as the next hop. It can then begin using this route to send data packets to the destination.

If this node is not the source node, it creates the forward route entry as described above. It consults its route table entry for the source node to determine the next hop for the RREP packet, and then forwards the RREP towards the source. If the node later receives a RREP for the same source node, it only updates its forward route and forwards the RREP to the source node if the RREP contains either a greater destination sequence number, or the same destination sequence number but smaller hopcount to the destination.

When any node generates or forwards a RREP, the precursor list for the corresponding destination is also updated by adding to it the next hop node to which the RREP is forwarded.

9.6. Hello Messages

A node MAY offer connectivity information by broadcasting local Hello messages as follows. Every HELLO_INTERVAL milliseconds, the node checks whether it has sent a broadcast (e.g., a RREQ or an appropriate layer 2 message) within the last HELLO_INTERVAL. If it has not, it MAY generate a broadcast RREP with TTL = 1, called a Hello message, with the message fields set as follows:

Destination IP Address
The node's IP address.

Destination Sequence Number
The node's latest sequence number.

Hop Count 0

Lifetime ALLOWED_HELLO_LOSS * HELLO_INTERVAL

A node MAY determine connectivity by listening for packets from its set of neighbors. If it receives no packets for more than ALLOWED_HELLO_LOSS * HELLO_INTERVAL milliseconds, the node SHOULD assume that the link to this neighbor is currently broken. When this happens, the node SHOULD proceed as in Section 9.8.

9.7. Maintaining Local Connectivity

Each forwarding node SHOULD keep track of its active next hops (i.e., which next hops have been used to forward packets towards some destination within the last ACTIVE_ROUTE_TIMEOUT milliseconds). This is done by updating the Lifetime field of a routing table entry used to forward data packets to current time plus ACTIVE_ROUTE_TIMEOUT milliseconds. For purposes of efficiency, each node may try to learn which of these active next hops are really in the neighborhood at the current time using one or more of the available link or network layer mechanisms, as described below.

- Any suitable link layer notification, such as those provided by IEEE 802.11, can be used to determine connectivity, each time a packet is transmitted to an active next hop. For example, absence of a link layer ACK or failure to get a CTS after sending RTS, even after the maximum number of retransmission attempts, will indicate loss of the link to this active next hop.

- Passive acknowledgment can be used when the next hop is expected to forward the packet, by listening to the channel for a transmission attempt made by the next hop. If transmission is not detected within NEXT_HOP_WAIT milliseconds or the next hop is not a forwarding node (and thus is never supposed to transmit the packet) one of the following methods should be used to determine connectivity.

  * Receiving an ICMP ACK message from the next hop. The ICMP ACK message SHOULD be sent to a forwarding node by a next hop which is also the destination as in the in the IP header of the packet. This should be done only when this destination has not sent any packets to the concerned forwarding node within the last HELLO_INTERVAL milliseconds.

  * A RREQ unicast to the next hop, asking for a route to the next hop.

  * An ICMP Echo Request message unicast to the next hop.

If a link to the next hop cannot be detected by any of these methods, the forwarding node SHOULD assume that the link is broken, and take corrective action by following the methods specified in Section 9.8.

9.8. Route Error (RERR) Messages

A node initiates a RERR message if it either detects a link break for an active next hop in its routing table, or if it receives a RERR from a neighbor for an active route.

If the precursor list for the next hop is non-empty, the node broadcasts a RERR. The unreachable destinations included in the RERR message are the next hop, and any additional destinations which are now unreachable due to the loss of the link. The contents of the RERR fields are set as follows:

Length
: Number of unreachable destinations included in the RERR message.

Unreachable Destination IP Address
: The destination which is now unreachable. Note that there can be more than one such field, as indicated by the Length field above.

When a node receives a RERR packet, for each unreachable destination included in the packet, the node determines whether the source node forwarding the RERR packet is the next hop used to reach that destination. If so, the node copies the value in the Hop Count route table field into the Last Hop Count field, and marks the Hop Count for that destination as infinity. It then checks its precursor list for each of the destinations. If one or more of the precursor lists are non-empty, it creates a RERR packet and includes as unreachable each destination with a non-empty precursor list. It then broadcasts this RERR message.

10. Node Operation - Multicast

This section describes the scenarios under which nodes generate control messages for multicast communication, and how the fields in the messages are handled.

10.1. Maintaining Multicast Tree Utilization Records

For each multicast tree to which a node belongs, either because it is a member of the group or because it is a router for the multicast tree, the node maintains a list of next hops -- i.e., those 1-hop neighbors that are likewise a part of the multicast tree. This list of next hops is used for forwarding messages received for the multicast group. A node will forward a multicast message to every such next hop, except that neighbor from which the message arrived. If there are multiple next hops, the forwarding operation MAY be performed by broadcasting the multicast packet to the node's neighbors; only the neighbors that belong to the multicast tree and have the sending node as a next hop and have not already seen that data packet continue to forward the multicast packet.

10.2. Generating Route Requests

A node sends a RREQ either when it determines that it should be a part of a multicast group, and it is not already a member of that group, or when it has a message to send to the multicast group but does not have a route to that group. If the node wishes to join the multicast group, it sets the 'J' flag in the RREQ; otherwise, it leaves the flag unset. The destination address of the RREQ is always set to the multicast group address. If the node knows the group leader and has a route to it, the node places the group leader's address in the Multicast Group Leader extension (Section 15.2), and unicasts the RREQ to the corresponding next hop for that destination. Otherwise, if the node does not have a route to the group leader, or if it does not know who the multicast group leader is, it broadcasts the RREQ and does not include the extension field.

The process of waiting for a RREP to a RREQ with a multicast destination address is the same as that described in Section 9.2. The node may resend the RREQ up to RREQ_RETRIES additional times if a RREP is not received. If a RREQ was unicast to a group leader and a RREP is not received within RREP_WAIT_TIME milliseconds, the node broadcasts subsequent RREQs for that multicast group across the network. If a RREP is not received after RREQ_RETRIES additional requests, the node may assume that there are no other members of that particular group within the connected portion of the network. If it wanted to join the multicast group, it then becomes the multicast group leader for that multicast group and initializes the sequence number of the multicast group. Otherwise, if it only wanted to send packets to that group without actually joining the group, it drops the packets it had for that group.

When the node wishes to join or send a message to a multicast group, it first consults its Group Leader Table. Based on the existence of an entry for the multicast group in this table, the node then formulates and sends the RREQ as described at the beginning of this section.

10.3. Forwarding Route Requests

The operation of nodes forwarding RREQs for multicast is similar to that for the reception and forwarding of RREQs as described in Section 9.3, with one exception. If the RREQ is a join request, it creates a multicast group next hop entry for the node from which it received the RREQ. The generation of the route reply (RREP) message is discussed in the following section.

10.4. Generating Route Replies

If a node receives a join RREQ for a multicast group, and it is already a member of the multicast tree for that group, the node updates its Multicast Route Table and then generates a RREP message. It unicasts the RREP back to the node indicated by the Source IP Address field of the received RREQ. The RREP contains the current sequence number for the multicast group and the IP address of the group leader. Additionally, it initializes the Hopcount field of the RREP to zero. Further information about the multicast group leader is entered into the Multicast Group Information extension (see Section 15.4).

A node can only respond to a join RREQ if it is a member of the multicast tree. If a node receives a multicast route request that is not a join message, it can reply if it has a current route to the multicast tree. Otherwise it will continue forwarding the request. If a node receives a join RREQ for a multicast group and it is not already a member of the multicast tree for that group, it will rebroadcast the RREQ to its neighbors.

In the event that a node receives a unicasted multicast route request that specifies its own IP address as the destination address (i.e., the source node believes this destination node to be the multicast group leader), but the node is in fact not the group leader, it can simply ignore the RREQ. The source node will time out after RREP_WAIT_TIME milliseconds and will broadcast a new RREQ without the group leader address specified.

Regardless of whether the multicast group leader or a multicast tree member generates the RREP, the RREP fields are set as follows:

Hop Count 0

Destination IP Address
The IP address of the multicast group.

Destination Sequence Number
The current multicast group sequence number.

Lifetime
The time for which nodes receiving the RREP consider the route to be valid (only used it the RREQ is not a join request).

The Multicast Group Information extension described in Section 15.4 is also included for join requests. If the node generating the RREP is not on the multicast tree (because the RREQ was not a join RREQ), it places its distance from the multicast tree in the Hop Count field, instead of 0.

10.5. Forwarding Route Replies

If an intermediate node receives a RREP in response to a RREQ that it has transmitted (or retransmitted on behalf of some other node), it increments the Hop Count and Multicast Group Hopcount fields and forwards the RREP along the path to the source of the RREQ.

When the node receives more than one RREP for the same RREQ, it saves the route information with the greatest sequence number, and beyond that the lowest hop count; it discards all other RREPs. This node forwards the first RREP towards the source of the RREQ, and then forwards later RREPs only if they have a greater sequence number or smaller metric.

10.6. Route Activation

When a node broadcasts a RREQ message, it is likely to receive more than one reply since any node in the multicast tree can respond. If the RREQ was not a join request, then once the source node receives the first RREP, it may begin using this route to forward data packets. On the other hand, if the RREQ was a join request, the RREP message sets up route pointers as it travels back to the source node. These route pointers may eventually graft a branch onto the multicast tree. If multiple branches to the same destination are created in such a manner, a loop will be formed. Hence, in order to prevent the formation of any such loops, it is necessary to activate only one of the routes created by the RREP messages. The RREP containing the largest destination sequence number is chosen to be the added branch to the multicast tree. In the event that a node receives more than one RREP with the same (largest) sequence number, it selects the first one with the smallest hop count, i.e., the shortest distance to a member of the multicast tree.

After waiting RREP_WAIT_TIME milliseconds, the node must select the route it wishes to use as its link to the multicast tree. This is accomplished by sending a Multicast Activation (MACT) message. The Destination IP Address field of the MACT packet is set to the IP address of the multicast group. The node unicasts this message to the selected next hop, effectively activating the route. It then sets the Activated flag in the next hop Multicast Route Table entry associated with that node. After receiving this message, the node to which the MACT was sent activates the route entry for the link in its multicast route table, thereby finalizing the creation of the tree branch. All neighbors not receiving this message time out and delete that node as a next hop for the multicast group in their route tables, having never activated the route entry for that next hop.

Two scenarios exist for a neighboring node receiving the MACT message. If this node was previously a member of the multicast tree, it does not propagate the MACT message any further. However, if the next hop selected by the source node's MACT message was not previously a multicast tree member, it will have propagated the original RREQ further up the network in search of nodes which are tree members. Thus it is possible that this node also received more than one RREP, as noted in section 10.5.

When the node receives a MACT selecting it as the next hop, it unicasts its own MACT to the node it has chosen as its next hop, and so on up the tree, until a node which was already a part of the multicast tree is reached.

10.7. Multicast Tree Pruning

A multicast group member can revoke its member status at any time. However, it can only actually leave the multicast tree if it is not a tree router for any other nodes in the multicast group (i.e., if it is a leaf node). If a node wishing to leave the multicast group is a leaf node, it unicasts to its next hop on the tree a MACT message with the 'P' flag set and with the Destination IP Address set to the IP address of the multicast group. It then deletes the multicast group information for that group from its Multicast Route Table. When its next hop receives this message, it deletes the sending node's information from its list of next hops for the multicast tree. If the removal of the sending node causes this node to become a leaf node, and if this node is also not a member of the multicast group, it may in turn prune itself by sending its own MACT message up the tree.

When the multicast group leader wishes to leave the multicast group, it proceeds in a manner similar to the one just described. If it is a leaf node, it may leave the group and unicast a prune message to its next hop. The next hop will act in the manner described in Section 10.10, since the prune message is coming from its upstream neighbor. Otherwise, if the group leader is not a leaf node, it may not prune itself from the tree. It takes the actions described in Section 10.9, where it selects one of its next hops and unicasts to it the MACT with set 'G' flag.

10.8. Repairing Link Breakages

Branches of the multicast tree become invalid if a broken link results in an infinite metric being associated with the route. When a broken link is detected between two nodes on the multicast tree, the node downstream of the break (i.e., the node which is further from the multicast group leader) is responsible for initiating the repair of the broken link. In order to repair the tree, the downstream node broadcasts a RREQ with destination IP address set to the IP address of the multicast group and with the 'J' flag set. The destination sequence number of the RREQ is the last known sequence number of the multicast group. The node also includes the Multicast Group Rebuild Extension. The Multicast Group Hop Count field of this extension is set to the distance of the source node from the multicast group leader. A node MUST have a hop count to the multicast group leader less than or equal to the indicated value in order to respond. This hop count requirement prevents nodes on the same side of the break as the node initiating the repair from replying to the RREQ.

The RREQ is broadcast using an expanding rings search. Because of the high probability that other nearby nodes can be used to rebuild the route, the original RREQ is broadcast with a TTL (time to live) field value equal to two more than the Multicast Group Hop Count. In this way, the effects of the link breakage may be localized. If no reply is received within RREP_WAIT_TIME milliseconds, all subsequent RREQs (up to RREQ_RETRIES additional attempts) will be broadcast across the entire network. Any node that is a part of the multicast tree and that has a hop count to the multicast group leader smaller than that contained in the RREQ can return a RREP. If there is more than one RREP received at the originating node, route deletions occur as described in the previous section.

At the end of the discovery period, the node selects its next hop and unicasts a MACT message to that node to activate the link, as described in Section 10.7. Additionally, since the node was repairing a tree break, it is likely that it is now a different distance from the group leader than it was before the break. If this is the case, it must inform its DOWNSTREAM next hops of their new distance from the group leader. It does this by broadcasting a MACT message with the 'U' flag set, and the Hop Count field set to the node's new distance from the group leader. This 'U' flag indicates that multicast tree nodes should update their distance from the group leader. If these nodes have downstream next hops, they in turn must send a MACT message with a set 'U' flag to their next hops, and so on. The Hop Count field is incremented by one each time the packet is received. When a node on the multicast tree receives the MACT message with the 'U' flag set, in determines whether this packet arrived from its UPSTREAM neighbor. If it did not, the node discards the packet.

When a link break occurs, it is possible that the tree will be repaired through different intermediate nodes. Hence, if the node UPSTREAM of the break is not a group member, and if the loss of that link causes it to become a leaf node, it sets a prune timer to wait for the link to be repaired. This PRUNE_TIMEOUT should be larger than RREP_WAIT_TIMEOUT to give the link time to be repaired. If, when this timer expires, the node has not received a MACT message selecting it to be a part of the repaired tree branch, it prunes itself from the tree by sending a MACT with set 'P' flag to its next hop, as previously described.

10.9. Tree Partitions

It is possible that after a link breaks, the tree cannot be repaired due to a network partition. If the node attempting to repair a tree link breakage does not receive a response after RREQ_RETRIES attempts, it can be assumed that the network has become partitioned and the multicast tree cannot be repaired at this time. In this situation, if the node which initiated the route rebuilding is a multicast group member, it becomes the new multicast group leader for its part of the multicast tree partition. It broadcasts a Group Hello for this multicast group. The 'U' flag in the Group Hello is set, indicating that there has been a change in the group leader information. All nodes receiving this message update their Group Leader Table to indicate the new group leader information. Nodes which are a part of the multicast tree also update the group leader information for that group in their Multicast Route Table to indicate the new group leader.

On the other hand, if the node which had initiated the repair is not a multicast group member, there are two possibilities. If it only has one next hop for the multicast tree, it prunes itself from the tree by unicasting a MACT message, with the 'P' flag set, to its next hop. The node receiving this message will note that the message came from its upstream link, i.e., from a node that is closer to the group leader than it is. If the node receiving this message is a multicast group member, it will become the new group leader and will broadcast a Group Hello message as indicated above. Otherwise, if it is not a multicast group member and it only has one other next hop link, it will similarly prune itself from the tree. This process continues until a multicast group member is reached.

The other possibility is that the node which initiated the rebuilding is not a group member and has more than one next hop for the tree. In this case, it cannot prune itself, since doing so would partition the tree. It instead chooses one of its next hops and unicasts a MACT with the 'G' flag set to that node. This flag indicates that the next group member to receive this message should become the new group leader. It then changes the direction of that link to be UPSTREAM. If the node's next hop is a group member, this node becomes the group leader. Otherwise, the node unicasts its own MACT message with the 'G' flag set to one of its next hops, and changes the direction of that link. Once a group member is reached, the new group leader is determined.

10.10. Reconnecting the Two Trees

In the event that a link break can not be repaired, the multicast tree remains partitioned until the two parts of the network become connected once again. A node from one partition of the network knows that it has come into contact with a node from the other partition of the network by noting the difference in the Group Hello message multicast group leader information. The multicast group leader with the lower IP address initiates the tree repair. For the purposes of this explanation, call this node GL1. GL1 unicasts a RREQ with both the 'J' and 'R' flags set to the group leader of the other network partition (GL2), using the node from which it received the Group Hello as the next hop. This RREQ contains the current value of GL1's multicast group sequence number. If any node that receives the RREQ is a member of GL2's multicast tree, it MUST forward the RREQ along its upstream link, i.e. towards GL2. This prevents any loops from being formed after the repair. Upon receiving the RREQ, GL2 takes the larger of its and the received multicast group sequence number, increments this value by one, and responds with a RREP. This is the group leader which will become the leader of the reconnected multicast tree. The 'R' flag of the RREP is set, indicating that this RREP is in response to a repair request.

As the RREP is propagated back to GL1, nodes add the incoming and outgoing links to the Multicast Route Table next hop entries if these entries do not already exist. The nodes also activate these entries, thereby adding the branch on to the multicast tree. If a node that was previously a member of GL1's tree receives the RREP, it MUST forward the packet along its link to its previous group leader (GL1). It then updates its group leader information to reflect GL2 as the new group leader, changes the direction of the next hop link associated with GL1 to DOWNSTREAM, and sets the direction of the link on which it received the RREP to UPSTREAM. When GL1 receives the RREP, it updates its group leader information and sets the link from which it received the RREP as its upstream link. The tree is now reconnected. The next time GL2 broadcasts a Group Hello, it sets the 'U' flag to indicate that there is a change in the group leader information and group members should update the corresponding information. All network nodes update their Group Leader Table to reflect the new group leader information.

10.11. Group Hello Messages

If a node sends a RREQ to join a multicast group ('J' flag set) and after RREQ_RETRIES attempts does not receives a response, it then becomes the multicast group leader. The node initializes the multicast group sequence number and then broadcasts a Group Hello message to inform network nodes that it is now the group leader for the multicast group. To ensure nodes maintain consistent and up-to-date information about who the multicast group leaders are, any node which is a group leader for a multicast group broadcasts such a Group Hello across the network every GROUP_HELLO_INTERVAL milliseconds. The contents of the GRPH fields are set as follows:

M Flag 0

Hop Count 0

Group Leader IP Address
The IP Address of the group leader.

Multicast Group IP Address
The IP Address of the Multicast Group for which the node is the group leader.

Multicast Group Sequence Number
One plus the last known sequence number of the multicast group.

Nodes receiving the Group Hello increment the Hop Count field by one before forwarding the message. When a node not on the multicast tree receives the GRPH message, it sets the M flag. This indicates

11. Broadcast

When a node wishes to generate a broadcast, it sends the broadcast packet to address 255.255.255.255. AODV does not define any valid behavior for transmissions to any directed broadcast address.

Every node maintains a list to keep track of which broadcast packets have already been received and retransmitted. The list contains, for each distinct broadcast packet received, the source IP address and the IP ident value from the IP header of the broadcast packet.

When a node receives a packet broadcast to address 255.255.255.255, it checks the source IP address and the IP ident value of the broadcast packet's IP header. The node then checks to see whether the broadcast packet has already been received in the past, and thus whether it has already retransmitted the broadcast packet. If there is no existing list entry containing the same IP source address and IP ident value, the node retransmits the broadcast packet. If there is such a list entry with matching source IP address and IP ident field, the node silently discards the broadcast packet.

List entries SHOULD be kept for at least BROADCAST_RECORD_TIME before the node expunges the record. BROADCAST_RECORD_TIME is a configurable parameter, but it MUST be at least equal to RREP_WAIT_TIME.

12. Quality of Service

AODV currently provides some minimal controls to enable mobile nodes in an ad hoc network to specify, as part of a RREQ, certain Quality of Service parameters that a route to a destination must satisfy.

In particular, a RREQ MAY include a Maximum Delay extension (see Section 15.5) or a Minimum Bandwidth extension (see Section 15.6).

If, after establishment of such a route, any node along the path detects that the requested Quality of Service parameters can no longer be maintained, that node MUST originate a ICMP QOS_LOST message back to the node which had originally requested the now unavailable parameters.

13. AODV and Aggregated Networks

AODV has been designed for use by mobile nodes with IP addresses that are not necessarily related to each other, to create an ad hoc network. However, in some cases a collection of mobile nodes MAY operate in a fixed relationship to each other and share a common subnet prefix, moving together within an area where an ad hoc network has formed. Call such a collection of nodes a ''subnet''. In this case, it is possible for a single node within the subnet to advertise reachability for all other nodes on the subnet, by responding with a RREP message to any RREQ message requesting a route to any node with the subnet routing prefix. Call the single node the ''subnet router''. In order for a subnet router to operate the AODV protocol for the whole subnet, it has to maintain a destination sequence number for the entire subnet. In any such RREP message sent by the subnet router, the Prefix Length field of the RREP message MUST be set to the length of the subnet prefix. Other nodes sharing the subnet prefix SHOULD NOT issue RREP messages, and SHOULD forward RREQ messages to the subnet leader.

14. Using AODV with Other Networks

In some configurations, an ad hoc network may be able to provide connectivity between external routing domains that do not use AODV. If the points of contact to the other networks can act as subnet routers (see Section 13) for any relevant networks within the external routing domains, then the ad hoc network can maintain connectivity to the external routing domains. Indeed, the external routing networks can use the ad hoc network defined by AODV as a transit network.

In order to provide this feature, a point of contact to an external network (call it an Infrastructure Router) has to act as the subnet router for every subnet of interest within the external network for which the Infrastructure Router can provide reachability. This includes the need for maintaining a destination sequence number for that external subnet.

If multiple Infrastructure Routers offer reachability to the same external subnet, those Infrastructure Routers have to cooperate (by means outside the scope of this specification) to provide consistent AODV semantics for ad hoc access to those subnets.

15. Extensions

RREQ and RREP messages have extensions defined in the following format:

```
 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      |    Length     |     type-specific data ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

where:

Type xx

Length The length of the type-specific data, not including the Type and Length fields of the extension.

Extensions with types between 128 and 255 may NOT be skipped. The rules for extensions will be spelled out more fully, and conform with the rules for handling IPv6 options.

15.1. Hello Interval Extension Format

```
 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      |    Length     | Hello Interval ...            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| ... Hello Interval, continued |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type xx

Length The length of the extension field.

Hello Interval
The number of milliseconds between successive transmissions of a Hello message.

The Hello Interval extension MAY be appended to a RREP message with TTL == 1, to be used by a neighboring receiver in determine how long to wait for subsequent such RREP messages (i.e., Hello messages; see section 9.6).

15.2. Multicast Group Leader Extension Format

This extension is appended to a RREQ by a node wishing to repair a multicast tree.

```
 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      |    Length     | Multicast Group Leader IP ... |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   ... Address (continued)     | Previous Hop IP Address ...   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   ... (continued)             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type xx

Length The length of the extension.

Multicast Group Leader IP Address
The IP Address of the Multicast Group Leader.

Previous Hop IP Address
The IP Address of the node which previously received the RREQ. This field is used when the RREQ is unicast to the group leader when a node wishes to join a multicast group.

This extension is used when unicasting the RREQ to the group leader. Each node receiving the RREQ updates the Previous Hop IP Address field to reflect its address.

15.3. Multicast Group Rebuild Extension Format

This extension is appended to a RREQ by a node wishing to repair a multicast tree.

```
 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      |    Length     |  Multicast Group Hop Count    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type xx

Length The length of the extension.

Multicast Group Hop Count
The distance in hops of the node sending the RREQ from the Multicast Group Leader.

This extension is used for rebuilding a multicast tree branch. It is used to ensure that only nodes as least as close to the group leader as indicated by the Multicast Group Hop Count field respond to the request.

15.4. Multicast Group Information Extension Format

The following extension is used to carry additional information for the RREP message (see Section 5) when sent to establish a route to a multicast destination.

```
 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      |    Length     |  Multicast Group Hop Count    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Multicast Group Leader IP Address               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type xx

Length The length of the extension field.

Multicast Group Hop Count
The distance of the node from the Multicast Group Leader.

Multicast Group Leader IP Address
The IP Address of the current Multicast Group Leader.

This extension is included when responding to a RREQ to join a multicast group. The node responding to the RREQ places its distance from the group leader in the Multicast Group Hop Count field.

15.5. Maximum Delay Extension Format

```
 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      |    Length     |          Max Delay            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type xx

Length The length of the extension field.

Max Delay The number of seconds allowed for a transmission from the source to the destination.

The Maximum Delay Extension can be appended to a RREQ by a requesting node in order to place a maximum bound on the acceptable time delay experienced on any acceptable path from the source to the destination.

Before forwarding the RREQ, an intermediate node MUST compare its NODE_TRAVERSAL_TIME to the (remaining) Max Delay indicated in the Maximum Delay Extension. If the Max Delay is less, the node MUST discard the RREQ and not process it any further. Otherwise, the node subtracts NODE_TRAVERSAL_TIME from the Max Delay value in the extension and continues processing the RREQ as specified in Section 9.3.

15.6. Minimum Bandwidth Extension Format

```
 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      |    Length     |    Minimum Bandwidth ...      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  ...  Minimum Bandwidth       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type xx

Length The length of the extension field.

Minimum Bandwidth
The amount of bandwidth (in kilobits/sec) needed for acceptable transmission from the source to the destination.

The Minimum Bandwidth Extension can be appended to a RREQ by a requesting node in order to specify the minimal amount of bandwidth that must be made available along acceptable path from the source to the destination.

Before forwarding the RREQ, an intermediate node MUST compare its available link capacity to the Minimum Bandwidth indicated in the extension. If the requested amount of bandwidth is not available, the node MUST discard the RREQ and not process it any further. Otherwise, the node continues processing the RREQ as specified in Section 9.3.

16. Configuration Parameters

This section gives default values for some important values associated with AODV protocol operations. A particular mobile node may wish to change certain of the parameters, in particular the NET_DIAMETER, MY_ROUTE_TIMEOUT, ALLOWED_HELLO_LOSS, RREQ_RETRIES, and possibly the HELLO_INTERVAL. In the latter case, the node should advertise the HELLO_INTERVAL in its Hello messages, by appending a Hello Interval Extension to the RREP message. Choice of these parameters may affect the performance of the protocol.

Parameter Name	Value
ACTIVE_ROUTE_TIMEOUT	3,000 Milliseconds
ALLOWED_HELLO_LOSS	2
BAD_LINK_LIFETIME	2 * RREP_WAIT_TIME
BCAST_ID_SAVE	30,000 Milliseconds
BROADCAST_RECORD_TIME	RREP_WAIT_TIME
GROUP_HELLO_INTERVAL	5,000 Milliseconds
HELLO_INTERVAL	1,000 Millisecond
MTREE_BUILD	2 * REV_ROUTE_LIFE
MY_ROUTE_TIMEOUT	2 * ACTIVE_ROUTE_TIMEOUT
NET_DIAMETER	35
NEXT_HOP_WAIT	NODE_TRAVERSAL_TIME + 10
NODE_TRAVERSAL_TIME	40 Milliseconds
PRUNE_TIMEOUT	ACTIVE_ROUTE_TIMEOUT
REV_ROUTE_LIFE	RREP_WAIT_TIME
RREP_WAIT_TIME	3 * NODE_TRAVERSAL_TIME * NET_DIAMETER / 2
RREQ_RETRIES	2
TTL_START	1
TTL_INCREMENT	2
TTL_THRESHOLD	7

NET_DIAMETER measures the maximum possible number of hops between two nodes in the network. NODE_TRAVERSAL_TIME is a conservative estimate of the average one hop traversal time for packets and should include queueing delays, interrupt processing times and transfer times. ACTIVE_ROUTE_TIMEOUT SHOULD be set to a longer value (at least 10,000 milliseconds) if link-layer indications are used to detect link breakages such as in IEEE 802.11 [3] standard. TTL_START should be set to at least 2 if Hello messages are used for local connectivity information. Performance of the AODV protocol is sensitive to the chosen values of these constants, which often depend on the characteristics of the underlying link layer protocol, radio technologies etc.

17. Security Considerations

Currently, AODV does not specify any special security measures. Route protocols, however, are prime targets for impersonation attacks, and must be protected by use of authentication techniques involving generation of unforgeable and cryptographically strong message digests or digital signatures. It is expected that, in environments where security is an issue, that IPSec authentication headers will be deployed along with the necessary key management to distribute keys to the members of the ad hoc network using AODV.

References

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

[2] Charles E. Perkins. Terminology for Ad-Hoc Networking. draft-ietf-manet-terms-00.txt, November 1997. (work in progress).

[3] IEEE Standards Department. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Standard 802.11--1997, 1997.

Author's Addresses

Questions about this memo can be directed to:

Charles E. Perkins
Communications Systems Laboratory
Nokia Research Center
313 FairChild Drive
Mountain View, CA 94303
USA
+1 650 625 2986
+1 650 691 2170 (fax)
charliep@iprg.nokia.com

Elizabeth M. Royer
Dept. of Electrical and Computer Engineering
University of California, Santa Barbara
Santa Barbara, CA 93106
+1 805 893 7788
+1 805 893 3262 (fax)
eroyer@alpha.ece.ucsb.edu

Samir R. Das
Division of Computer Science
University of Texas at San Antonio
San Antonio, TX 78249
+1 210 458 5537
+1 210 458 4437 (fax)
samir@cs.utsa.edu