C. Perkins
Futurewei
I. Chakeres
CenGen

December 1, 2012

Dynamic MANET On-demand (AODVv2) Routing
draft-ietf-manet-dymo-24

Abstract

The Dynamic MANET On-demand (AODVv2) routing protocol is intended for use by mobile routers in wireless, multihop networks. AODVv2 determines unicast routes among AODVv2 routers within the network in an on-demand fashion, offering on-demand convergence in dynamic topologies.

Status of this Memo

Copyright Notice

Table of Contents

1. Overview

The Dynamic MANET On-demand (AODVv2) routing protocol [formerly named DYMO] enables on-demand, multihop unicast routing among AODVv2 routers in mobile ad hod networks [MANETs][RFC2501]. The basic operations of the AODVv2 protocol are route discovery and route maintenance. Route discovery is performed when an AODVv2 router must transmit a packet towards a destination for which it does not have a route. Route maintenance is performed to avoid prematurely expunging routes from the route table, and to avoid dropping packets when a route being used to forward packets from the source to a destination breaks.

During route discovery, an AODVv2 router multicasts a Route Request message (RREQ) to find a route toward a particular destination, via the AODVv2 router responsible for this destination. Using a hop-by-hop retransmission algorithm, each intermediate AODVv2 router receiving the RREQ message records a route toward the originator. When the target's AODVv2 router (TargRtr) receives the RREQ, it records a route toward the originator and responds with a Route Reply (RREP) unicast hop-by-hop toward the originating AODVv2 router. Each intermediate AODVv2 router that receives the RREP creates a route toward the target, and unicasts the RREP hop-by-hop toward the originator. When the originator's AODVv2 router receives the RREP, routes have then been established between the originating AODVv2 router and the target AODVv2 router in both directions.

Route maintenance consists of two operations. In order to preserve active routes, AODVv2 routers extend route lifetimes upon successfully forwarding a packet. When a data packet is received for forwarding and there is no valid route for the destination, then the AODVv2 router of the source of the packet is notified via a Route Error (RERR) message. Each upstream router that receives the RERR marks the route as broken. Before such an upstream AODVv2 router could forward a packet to the same destination, it would have to perform route discovery again for that destination.

AODVv2 uses sequence numbers to assure loop freedom [Perkins99], similarly to AODV. Sequence numbers enable AODVv2 routers to determine the temporal order of AODVv2 route discovery messages, thereby avoiding use of stale routing information. Unlike AODV, AODVv2 uses RFC 5444 message and TLV formats.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and

"OPTIONAL" in this document are to be interpreted as described in [RFC2119].

This document also uses some terminology from [RFC5444].

This document defines the following terminology:

Adjacency
: A bi-directional relationship between neighboring routers for the purpose of exchanging routing information. Not every pair of neighboring routers will necessarily form an adjacency. Neighboring routers may form an adjacency based on various information or other protocols; for example, exchange of AODVv2 routing messages, other protocols (e.g. NDP [RFC4861] or NHDP [RFC6130]), or manual configuration. Loss of a routing adjacency may also be based upon similar information; monitoring of adjacencies where packets are being forwarded is required (see Section 8.2).

AODVv2 Router
: An IP addressable device in the ad-hoc network that performs the AODVv2 protocol operations specified in this document.

AODVv2 Sequence Number (SeqNum)
: An AODVv2 Sequence Number is an unsigned integer maintained by each AODVv2 router. This sequence number guarantees the temporal order of routing information to maintain loop-free routes. The value zero (0) is reserved to indicate that the SeqNum for a destination address is unknown.

Current_Time
: The current time as maintained by the AODVv2 router.

disregard
: Ignore for further processing (see Section 5.4), and delete unless it is required to keep the message in the packet for purposes of authentication.

Handling Router (HandlingRtr)
: HandlingRtr denotes the AODVv2 router handling an AODVv2 message.

Incoming Link
: A link over which an AODVv2 has received a message from one of its adjacent routers.

MANET
A Mobile Ad Hoc Network as defined in [RFC2501].

node
An IP addressable device in the ad-hoc network. A node may be an AODVv2 router, or it may be a device in the network that does not perform any AODVv2 protocol operations. All nodes in this document are either AODVv2 Routers or else Router Clients.

Originating Node (OrigNode)
The Originating Node is the node that launched the application requiring communication with the Target Node. If OrigNode is not itself an AODVv2 router, its AODVv2 router (OrigRtr) has the responsibility to generate a AODVv2 RREQ message on behalf of OrigNode when necessary to multicast a route discovery message.

Originating Router (OrigRtr)
The Originating Router is the AODVv2 router that serves OrigNode. OrigRtr generates the RREQ message to discover a route for TargNode.

reactive
A protocol operation is said to be "reactive" if it is performed only in reaction to specific events. As used in this document, "reactive" is essentially synonymous with "on-demand".

Routable Unicast IP Address
A routable unicast IP address is a unicast IP address that when put into the IP.DestinationAddress field is scoped sufficiently to be forwarded by a router. Globally-scoped unicast IP addresses and Unique Local Addresses (ULAs) [RFC6549] are examples of routable unicast IP addresses.

Route Error (RERR)
A RERR message is used to indicate that an AODVv2 router does not have a route toward one or more particular destinations.

Route Reply (RREP)
A RREP message is used to establish a route between the RREQ TargetNode and OrigNode, at all the AODVv2 routers between them.

Route Request (RREQ)
An AODVv2 router uses a RREQ message to discover a valid route to a particular destination address, called the RREQ TargetNode. An AODVv2 router processing a RREQ receives routing information for the RREQ OrigNode.

Router Client
An AODVv2 router may be configured with a list of other IP addresses and networks which correspond to other non-router nodes which require the services of the AODVv2 router for route discovery and maintenance. An AODVv2 is always its own client, so that the list of client IP addresses is never empty.

Sequence Number (SeqNum)
Same as AODVv2 Sequence Number.

Target Node (TargNode)
The Target Node denotes the node for which a route is needed.

Target Router (TargRtr)
The TargetRtr denotes the AODVv2 router which serves TargNode.

Type-Length-Value structure (TLV)
A generic way to represent information as specified in [RFC5444].

Unreachable Node (UnreachableNode)
An UnreachableNode is a node for which a forwarding route is unknown.

valid route
A route that can be used for forwarding; in other words a route that is not Broken or Expired.

3. Notational Conventions

This document uses the conventions found in Table 1 to describe information in the fields from [RFC5444].

Notation	Information Location and/or Meaning
Route[DestAddr]	A route table entry towards DestAddr
Route[Addr]{field}	A field in a route table entry
--	--
RREQ.{field}	Field in RREQ
RREP.{field}	Field in RREP
RERR.{field}	Field in RERR
--	--
MsgHdr	the RFC5444 Message Header
MsgTLV	an RFC5444 Message TLV
MetricTypeTLV	MetricType MsgTLV for Metric AddrTLV
MAL	MsgHdr.<msg-addr-length>
--	--
AddrBlk	an RFC5444 address block
AddrBlk[1]	The first address slot in AddrBlk
AddrBlk[N]	The Nth address slot in AddrBlk
AddrBlk[OrigNode]	AddrBlk[1]
AddrBlk[TargNode]	AddrBlk[2]
AddrTLV	an RFC5444 address block TLV
AddrTLV[1]	the first item in AddrTLV
AddrTLV[N]	the Nth item in AddrTLV
AddrTLV[OrigNode]	AddrTLV[1]
AddrTLV[TargNode]	AddrTLV[2]
HopCountTLV	Metric8 AddrTLV when MetricTypeTLV=3
Metric8TLV	Metric8 AddrTLV
SeqNumTLV	Sequence Number TLV for AddrBlk addresses
RteAddrBlk	the main address block in a RteMsg
RteSeqNumTLV	Sequence Numbers for RteAddrBlk addresses
UnreachAddrBlk	Unreachable Node AddrBlk in RERR
--	--
OrigRtr	RREQ Originating Router
OrigNode	Originating Node
RREQ_Gen	AODVv2 router originating an RREQ
RREP_Gen	AODVv2 router responding to an RREQ
RteMsg	either RREQ or RREP
RteMsg_Orig	Originator of a RteMsg
HandlingRtr	Handling Router
TargRtr	Target Router
TargNode	Target Node
UnreachableNode	Unreachable Node

Table 1

4. Applicability Statement

The AODVv2 routing protocol is designed for stub (i.e., non-transit) or disconnected (i.e., from the Internet) mobile ad hoc networks (MANETs). AODVv2 handles a wide variety of mobility patterns by determining routes on-demand. AODVv2 also handles a wide variety of traffic patterns. In networks with a large number of routers, AODVv2 is best suited for relatively sparse traffic scenarios where any particular router forwards packets to only a small percentage of the AODVv2 routers in the network, due to the on-demand nature of route discovery and route maintenance.

Although AODVv2 is closely related to AODV [RFC3561], and has some of the features of DSR [RFC4728], AODVv2 is not interoperable with either of those other two protocols.

AODVv2 is applicable to memory constrained devices, since little routing state is maintained in each AODVv2 router. Only routing information related to routes between active sources and destinations is maintained, in contrast to proactive routing protocols that require routing information to all routers within the MANET be maintained.

AODVv2 supports routers with multiple interfaces, as long as each interface has its own IP address. In addition to routing for their local processes, AODVv2 routers can also route on behalf of other non-routing nodes (i.e., "hosts", or, in this document, "clients"), reachable via those interfaces. Any such node which is not itself an AODVv2 router SHOULD NOT be served by more than one AODVv2 router.

Multi-homing is difficult unless the sequence number is expanded to include the IP address as well as OwnSeqNum. Otherwise, comparing sequence numbers would not work to evaluate freshness. Even when the IP address is included, there isn't a good way to compare sequence numbers from different IP addresses, but at least a handling node can determine whether the two given sequence numbers are comparable. If the route table can store multiple routes for the same destination, then multi-homing can work with sequence numbers augmented by IP addresses.

AODVv2 routers perform route discovery to find a route toward a particular destination. Therefore, AODVv2 routers MUST must be configured to respond to RREQs for a certain set of addresses. When AODVv2 is the only protocol interacting with the forwarding table, AODVv2 MAY be configured to perform route discovery for all unknown unicast destinations.

At all times within an AODVv2 MANET, only one AODVv2 router SHOULD be serve any particular routing client. The coordination among multiple AODVv2 routers to distribute routing information correctly for a shared address (i.e. an address that is advertised and can be reached via multiple AODVv2 routers) is not described in this document. The AODVv2 router operation of shifting responsibility for a routing client from one AODVv2 router to another is mentioned in Appendix C. Each AODVv2 router, if serving router clients other than itself, is configured with information about the IP addresses of its clients. No AODVv2 router is required to have information about the relationship between any other AODVv2 router and its router clients. Address assignment procedures are entirely out of scope for AODVv2.

AODVv2 only utilizes bidirectional links. In the case of possible unidirectional links, either blacklists (see Section 5.2) or other means (e.g. adjacency establishment with only neighboring routers that have bidirectional communication as indicated by NHDP [RFC6130]) of assuring and monitoring bi-directionality is recommended. Otherwise, persistent packet loss or persistent protocol failures could occur. The Cost(L) of bidirectional link L may depend upon the direction across the link for which the cost is measured.

The routing algorithm in AODVv2 may be operated at layers other than the network layer, using layer-appropriate addresses. The routing algorithm makes of some persistent state; if there is no persistent storage available for this state, recovery can impose a performance penalty in case of AODVv2 router reboots.

5. Data Structures

5.1. Route Table Entry

The route table entry is a conceptual data structure. Implementations may use any internal representation so long as it provides access to the same information as specified below.

Conceptually, a route table entry has the following fields:

Route.Address
: The (host or network) destination address of the node(s) associated with the routing table entry

Route.PfxLen
: The value is the length of the netmask/prefix. If the value of the Route.PfxLen is nonzero and different than the length of addresses in the address family used by the AODVv2 routers, the associated address is a routing prefix, rather than a host address.

Route.SeqNum
The AODVv2 SeqNum associated with a route table entry

Route.NextHopAddress
An IP address of the adjacent AODVv2 router on the path toward the Route.Address

Route.NextHopInterface
The interface used to send packets toward the Route.Address

Route.LastUsed
The time that this route was last used

Route.ExpirationTime
The time at which this route must expire

Route.Broken
A flag indicating whether this Route is broken. This flag is set to true if the next-hop becomes unreachable or in response to processing to a RERR (see Section 8.4)

Route.MetricType
The type of the metric for the route towards Route.Address

Route.Metric
The cost of the route towards Route.Address

A route table entry (i.e., a route) may be in one of the following states:

Active
An Active route is in current use for forwarding packets

Idle
An Idle route can be used for forwarding packets, even though it is not in current use

Expired
After a route has been idle for too long, it expires, and may no longer be used for forwarding packets

Broken
A route marked as Broken cannot be used for forwarding packets but still has valid destination sequence number information.

Timed
The expiration of a Timed route is controlled by the Route.ExpirationTime time of the route table entry, not MAX_IDLETIME. Until that time, a Timed route can be used for forwarding packets. Afterwards, the route must be Expired (or expunged).

The route's state determines the operations that can be performed on the route table entry. During use, an Active route is maintained continuously by AODVv2 and is considered to remain active as long as it is used at least once during every ACTIVE_INTERVAL. When a route is no longer Active, it becomes an Idle route. After a route remains Idle for MAX_IDLETIME, it becomes an Expired route; after that, the route is not used for forwarding, but the sequence number information can be maintained until the destination sequence number has had no updates for MAX_SEQNUM_LIFETIME. After MAX_SEQNUM_LIFETIME, old sequence number information is considered no longer valuable and the route is expunged.

MAX_SEQNUM_LIFETIME is the time after a reboot during which an AODVv2 router MUST NOT transmit any routing messages. Thus, if all other AODVv2 routers expunge routes to the rebooted router after that time interval, the rebooted AODVv2 router's sequence number will not be considered stale by any other AODVv2 router in the MANET.

When the link to a route's next hop is broken, the route is marked as being Broken, and the route may no longer be used.

5.2. Bidirectional Connectivity During Route Discovery and Blacklists

To avoid repeated failure of Route Discovery, an AODVv2 router (HandlingRtr) handling a RREP message MAY attempt to verify connectivity to the next upstream router towards AODVv2 router originating an RREQ message, by including the Unicast Response Request message TLV (see Section 15.2) in the RREP. Any unicast packet will satisfy the Response Request, for example an ICMP REPLY message. If the verification fails, HandlingRtr SHOULD put the upstream neighbor in a blacklist. RREQs received from a blacklisted node SHOULD NOT be retransmitted by HandlingRtr. However, the upstream neighbor should not be permanently blacklisted; after a certain time (MAX_BLACKLIST_TIME), it should once again be considered as a viable upstream neighbor for route discovery operations.

For this purpose, a list of blacklisted nodes along with their time of removal should be maintained:

BlacklistNode
The IP address of the node that did not verify bidirectional connectivity.

BlacklistRmTime
The time at which BlacklistNode will be removed from the blacklist.

5.3. Router Clients and Client Networks

An AODVv2 router may offer routing services to other nodes that are not AODVv2 routers. The AODVv2 Sequence Number is (by definition) the same for the AODVv2 router and each of its clients.

For this purpose, a list of IP addresses nodes along with relevant prefixes must be configured on each AODVv2:

Client IP address
The IP address of the node that requires routing service from the AODVv2 router.

Client Prefix Length
The length of the routing prefix associated with the client IP address.

If the Client Prefix Length is not the full length of the Client IP address, then the prefix defines a Client Network. If an AODVv2 router is configured to serve a Client Network, then the AODVv2 router MUST serve every node that has an address within the range defined by the routing prefix of the Client Network. The list of Routing Clients for an AODVv2 router is never empty, since an AODVv2 router is always its own client as well.

5.4. AODVv2 Packet Header Fields and Information Elements

In its default mode of operation, AODVv2 uses the UDP port 269 [RFC5498] to carry protocol packets. In addition, IP Protocol Number 138 has been reserved for MANET protocols [RFC5498]. Most AODVv2 messages are sent with the IP destination address set to the link-local multicast address LL-MANET-Routers [RFC5498] unless otherwise specified. Therefore, all AODVv2 routers MUST subscribe to LL-MANET-Routers [RFC5498] to receiving AODVv2 messages. In order to reduce multicast overhead, retransmitting multicast packets in MANETs SHOULD be done according to methods specified in [RFC6621]. AODVv2 does not specify which method should be used to restrict the set of AODVv2 routers that have the responsibility to retransmit multicast packets. Note that multicast packets MAY be sent via unicast. For example, this may occur for certain link-types (non-broadcast media), for manually configured router adjacencies, or in order to improve robustness.

The IPv4 TTL (IPv6 Hop Limit) field for all packets containing AODVv2 messages is set to 255. If a packet is received with a value other than 255, any AODVv2 message contained in the packet MUST be disregarded by AODVv2. This mechanism, known as "The Generalized TTL Security Mechanism" (GTSM) [RFC5082] helps to assure that packets have not traversed any intermediate routers.

IP packets containing AODVv2 protocol messages SHOULD be given priority queuing and channel access.

AODVv2 messages are transmitted in packets that conform to the packet and message format described in [RFC5444]. Here is a brief description of the format.

A packet formatted according to RFC5444 contains zero or more messages.

A message contains a message header, message TLV block, and zero or more address blocks.

Each address block may also have associated TLV blocks.

If a packet contains only a single AODVv2 message and no packet TLVs, it need not include a packet-header [RFC5444]. The length of an address (32 bits for IPv4 and 128 bits for IPv6) inside an AODVv2 message is indicated by the msg-addr-length (MAL) in the msg-header, as specified in [RFC5444].

When multiple messages are aggregated into a single packet according to RFC 5444 formatting, and the aggregation of messages is also authenticated (e.g., with IPsec), it becomes unfeasible to delete individual messages. In such cases, instead of deleting individual messages, they are maintained in the aggregation of messages, but simply ignored for further processing. In such cases where individual messages cannot be deleted, in this document "disregarded" means "ignored". Otherwise, any such "disregarded" AODVv2 messages SHOULD be deleted from the aggregated messages in the RFC 5444 packet.

5.5. AODVv2 Sequence Numbers

AODVv2 sequence numbers allow AODVv2 routers to evaluate the freshness of routing information. Proper maintenance of sequence numbers assures that the destination sequence number value stored by intermediate AODVv2 routers is monotonically increasing along any path from any source to the destination. As a consequence, loop freedom is assured.

Each AODVv2 router in the network MUST maintain its own sequence number (OwnSeqNum, a 16-bit unsigned integer). An AODVv2 router increments its OwnSeqNum as follows. Most of the time, OwnSeqNum is incremented by simply adding one (1). But to increment OwnSeqNum when it has the value of the largest largest possible number representable as a 16-bit unsigned integer (i.e., 65,535), it MUST be set to one (1). In other words, the sequence number after 65,535 is 1.

An AODVv2 router SHOULD maintain OwnSeqNum in persistent storage. If an AODVv2 router’s OwnSeqNum is lost, it MUST take the following actions to avoid the danger of routing loops. First, the AODVv2 router MUST invalidate all route table entries, by setting Route.Broken for each entry. Furthermore the AODVv2 router MUST wait for at least MAX_SEQNUM_LIFETIME before transmitting or retransmitting any AODVv2 RREQ or RREP messages. If an AODVv2 protocol message is received during this waiting period, the AODVv2 router SHOULD perform normal route table entry updates. If a data packet is received for forwarding to another destination during this waiting period, the AODVv2 router MUST transmit a RERR message indicating that no route is available. At the end of the waiting period the AODVv2 router sets its OwnSeqNum to one (1) and begins performing AODVv2 protocol functions again.

5.6. Enabling Alternate Metrics

Route selection in AODVv2 MANETs depends upon associating metric information with each route table entry. When presented with candidate route update information, deciding whether to use the update involves evaluating the metric. Some applications may require the consideration of metric information other than Hop Count, which has traditionally been the default metric associated with routes in MANET. In fact, it is well known that reliance on Hop Count can cause selection of the worst possible route in many situations.

It is beyond the scope of this document to describe how applications specify route selection at the time they launch processing. One possibility would be to provide a route metric preference as part of the library routines for opening sockets. In view of the above considerations, it is important to enable route selection based on metric information other than Hop Count -- in other words, based on "alternate metrics". Each such alternate metric identifies a "cost" of using the associated route, and there are many different kinds of cost (latency, delay, financial, energy, etc.).

The most significant change when enabling use of alternate metrics is to require the possibility of multiple routes to the same destination, where the "cost" of each of the multiple routes is measured by a different alternate metric. The other change relevant to AODVv2 is that the method by which route updates are tested for usefulness has to be slightly generalized to depend upon a more abstract method of evaluation which, in this document, is named "Cost(R)", where 'R' is the route information to be evaluated. From the above, the route table information for 'R' must always include the type of metric by which Cost(R) is evaluated, so the metric type does not have to be shown as a distinct parameter for Cost(R). Since determining loop freedom is known to depend on comparing the Cost(R) of route update information to the Cost(R) of an existing stored route using the same metric, AODVv2 must also be able to invoke an abstract routine which in this document is called "LoopFree(R1, R2)". LoopFree(R1, R2) returns TRUE when, given that R2 is loop-free and Cost(R2) is the cost of route R2, Cost(R1) is known to guarantee loop freedom of the route R1. In this document, LoopFree(R1,R2) will only be invoked for routes R1 and R2 which use the same metric.

Generally, HopCount may still be considered the default metric for use in MANETs, notwithstanding the above objections. Each metric has to have a Metric Type, and the Metric Type is allocated by IANA as specified in [RFC6551]. Each Route has to include the Metric Type as part of the route table entry for that route. Hop Count has Metric Type assignment 3. The Cost of a route using Metric Type 3 is naturally the Hop Count between the router and the destination. For routes R1 and R2 using Metric Type 3, LoopFree (R1, R2) is TRUE when Cost(R2) <= (Cost(R1) + 1). The specification of Cost(R) and LoopFree(R1,R2) for metric types other than 3 is beyond the scope of this document.

Whenever an AODV router receives metric information in an incoming message, the value of the metric is as measured by the transmitting router, and does not reflect the cost of traversing the incoming link. In order to simplify the description of storing accrued route costs in the route table, the Cost() function is also defined to return the value of traversing a link 'L'. In other words, the domain of the Cost() function is enlarged to include links as well as routes. For Metric Type 3, (i.e., the HopCount metric) Cost(L) = 1 for all links. The specification of Cost(L) for metric types other than 3 is beyond the scope of this document. Whether the argument of the Cost() function is a link or a route will, in this document, always be clear. As a natural result of the way routes are looked up according to conformant metric type, all intermediate routers handling a RteMsg will assign the same metric type to all metric information in the RteMsg.

For some metrics, a maximum value is defined, namely MAX_METRIC[i] where 'i' is the Metric Type. AODVv2 does not store routes that cost more than MAX_METRIC[i]. MAX_METRIC[3] is defined to be MAX_HOPCOUNT, where as before 3 is the Metric Type of the HopCount metric.

6. AODVv2 Operations on Route Table Entries

In this section, operations are specified for updating the route table due to timeouts and route updates within AODVv2 messages. The route update information in AODVv2 messages includes the destination IP address (DestIP), SeqNum and prefix length associated with DestIP, and the Metric from DestIP to the node transmitting the AODVv2 message. DestIP information and prefix length are encoded within an RFC 5444 Address Block, and the SeqNum and Metric associated with each DestIP are encoded in RFC 5444 AddrTLVs. Optionally, there may be AddedNode route updates included in AODVv2 messages, as specified in Section 13.7. In this section, RteMsg is either RREQ or RREP, RteMsg.Addr denotes the [i]th address in an RFC 5444 AddrBlk of the RteMsg, RteMsg.PfxLen denotes the associated prefix length for RteMsg.Addr, and RteMsg.{field} denotes the corresponding value in the named AddrTLV block associated with RteMsg.Addr. All SeqNum comparisons use signed 16-bit arithmetic.

6.1. Evaluating Incoming Routing Information

If the incoming RteMsg does not have a MetricTypeTLV, then the metric information contained by RteMsg is considered to be of type DEFAULT_METRIC_TYPE. Whenever an AODVv2 router (HandRtr) handles an incoming RteMsg (i.e., RREQ or RREP), for every relevant address (RteMsg.Addr) in the RteMsg, HandRtr searches its route table to see if there is a route table entry with the same MetricType of the RteMsg, matching RteMsg.Addr. If not, HandRtr creates a route table entry for RteMsg.Addr as described in Section 6.2. Otherwise, HandRtr compares the incoming routing information in RteMsg against the already stored routing information in the route table entry (Route) for RteMsg.Addr, as described below.

Suppose a route table entry (Route[RteMsg.Addr]) uses the same metric type as the incoming routing information, and contains Route.SeqNum, Route.Metric, and Route.Broken. Suppose the incoming routing information for Route.Addr is RteMsg.SeqNum and RteMsg.Metric. The incoming routing information is compared as follows:

1. Stale:: RteMsg.SeqNum < Route.SeqNum :
   If RteMsg.SeqNum < Route.SeqNum the incoming information is stale. Using stale routing information is not allowed, since that might result in routing loops. HandRtr MUST disregard the routing information for RteMsg.Addr.

2. Unsafe against loops:: (TRUE != LoopFree (RteMsg, Route)) :
   If RteMsg is not Stale (as in (1)), RteMsg.Metric is next considered to insure loop freedom. If (TRUE != LoopFree (RteMsg, Route)) (see Section 5.6), then the incoming RteMsg information is not guaranteed to prevent routing loops, and it MUST NOT be used.

3. Longer::
   (RteMsg.Metric >= Route.Metric) && (Route.Broken==FALSE)
   When RteMsg.SeqNum is the same as in a valid route table entry, and LoopFree (RteMsg, Route) assures loop freedom, incoming information still does not offer any improvement over the existing route table information if RteMsg.Metric >= Route.Metric. Using such incoming routing information to update a route table entry is not recommended.

4. Offers improvement::
   Incoming routing information that does not match any of the above criteria is better than existing routing table information and SHOULD be used to improve the route table. The following pseudo-code illustrates whether incoming routing information should be used to update an existing route table entry as described in Section 6.2.

```
  (RteMsg.SeqNum > Route.SeqNum) OR
 {(RteMsg.SeqNum == Route.SeqNum) AND
[(RteMsg.Metric < Route.Metric) OR
((Route.Broken == TRUE) && LoopFree (RteMsg, Route))]}
```

   The above logic corresponds to placing the following conditions on the incoming route update (compared to the existing route table entry) before it can be used:

   * it is more recent, or

   * it is not stale and is shorter, or

   * it can safely repair a broken route.

6.2. Applying Route Updates To Route Table Entries

To apply the route update, the route table entry is populated with the following information:

- Route.Address := RteMsg.Addr

- If (RteMsg.PfxLen != 0), then Route.PfxLen := RteMsg.PfxLen

- Route.SeqNum := RteMsg.SeqNum

- Route.NextHopAddress := IP.SourceAddress (i.e., an address of the node from which the RteMsg was received)

- Route.NextHopInterface is set to the interface on which RteMsg was received

- Route.Broken flag := FALSE

- If RteMsg.MetricType is included, then Route.MetricType := RteMsg.MetricType. Otherwise, Route.MetricType := DEFAULT_METRIC_TYPE.

- Route.MetricType := RteMsg.MetricType

- Route.Metric := RteMsg.Metric

- Route.LastUsed := Current_Time

- If RteMsg.VALIDITY_TIME is not included, then Route.ExpirationTime := MAXTIME, otherwise Route.ExpirationTime := Current_Time + RteMsg.VALIDITY_TIME

With these assignments to the route table entry, a route has been made available, and the route can be used to send any buffered data packets and subsequently to forward any incoming data packets for Route.Addr. An updated route entry also fulfills any outstanding route discovery (RREQ) attempts for Route.Addr.

6.3. Route Table Entry Timeouts

During normal operation, AODVv2 does not require any explicit timeouts to manage the lifetime of a route. However, the route table entry MUST be examined be before using it to forward a packet, as discussed in Section 8.1. Any required expiry or deletion can occur at that time. Nevertheless, it is permissible to implement timers and timeouts to achieve the same effect.

At any time, the route table can be examined and route table entries can be expunged according to their current state at the time of examination, as follows.

- o An Active route MUST NOT be expunged.
- o An Idle route SHOULD NOT be expunged.
- o An Expired route MAY be expunged (least recently used first).
- o A route MUST be expunged if (Current_Time - Route.LastUsed) >= MAX_SEQNUM_LIFETIME.
- o A route MUST be expunged if Current_Time >= Route.ExpirationTime

If precursor lists are maintained for the route (as described in Section 13.3) then the precursor lists must also be expunged at the same time that the route itself is expunged.

7. Routing Messages RREQ and RREP (RteMsgs)

AODVv2 message types RREQ and RREP are together known as Routing Messages (RteMsgs) and are used to discover a route between an Originating and Target Node, denoted here by OrigNode and TargNode. The constructed route is bidirectional, enabling packets to flow between OrigNode and TargNode. RREQ and RREP have similar information and function, but have some differences in their rules for handling. The main difference between the two messages is that RREQ messages are typically multicast to solicit a RREP, whereas RREP is typically unicast as a response to RREQ.

When an AODVv2 router needs to forward a data packet from a node (OrigNode) in its set of router clients, and it does not have a forwarding route toward the packet’s IP destination address (TargNode), the AODVv2 router (in this section, called RREQ_Gen) generates a RREQ (as described in Section 7.3) to discover a route toward TargNode. Subsequently RREQ_Gen awaits reception of an RREP message (see Section 7.4) or other route table update (see Section 6.2) to establish a route toward TargNode. The RREQ message contains routing information to enable RREQ recipients to route packets back to OrigNode, and the RREP message contains routing information enabling RREP recipients to route packets to TargNode.

7.1. Route Discovery Retries and Buffering

After issuing a RREQ, as described above RREQ_Gen awaits a RREP providing a bidirectional route toward Target Node. If the RREP is not received within RREQ_WAIT_TIME, RREQ_Gen may retry the Route Discovery by generating another RREQ. Route Discovery SHOULD be considered to have failed after DISCOVERY_ATTEMPTS_MAX and the corresponding wait time for a RREP response to the final RREQ. After the attempted Route Discovery has failed, RREQ_Gen MUST wait at least RREQ_HOLDDOWN_TIME before attempting another Route Discovery to the same destination.

To reduce congestion in a network, repeated attempts at route discovery for a particular Target Node SHOULD utilize an binary exponential backoff.

Data packets awaiting a route SHOULD be buffered by RREQ_Gen. This buffer SHOULD have a fixed limited size (BUFFER_SIZE_PACKETS or BUFFER_SIZE_BYTES). Determining which packets to discard first is a matter of policy at each AODVv2 router; in the absence of policy constraints, by default older data packets SHOULD be discarded first. Buffering of data packets can have both positive and negative effects (albeit usually positive). Nodes without sufficient memory available for buffering SHOULD be configured to disable buffering by configuring BUFFER_SIZE_PACKETS == 0 and BUFFER_SIZE_BYTES == 0. Doing so will affect the latency required for launching TCP applications to new destinations.

If a route discovery attempt has failed (i.e., DISCOVERY_ATTEMPTS_MAX attempts have been made without receiving a RREP) to find a route toward the Target Node, any data packets buffered for the corresponding Target Node MUST BE dropped and a Destination Unreachable ICMP message (Type 3) SHOULD be delivered to the source of the data packet. The code for the ICMP message is 1 (Host unreachable error). If RREQ_Gen is not the source (OrigNode), then the ICMP is sent over the interface from which OrigNode sent the packet to the AODVv2 router.

7.2. RteMsg Structure

RteMsgs have the following general format:

```
       +-----------------------------------------------------------------+
       |                      RFC 5444 Packet Header                     |
       +-----------------------------------------------------------------+
       |              RFC 5444 Message Header <msg-hopcount>             |
       +-----------------------------------------------------------------+
       |    RFC 5444 MsgHdr, opt. DestOnly TLV, opt. MetricTypeTLV       |
       +-----------------------------------------------------------------+
       |       RteAddrBlk {[1]:=RREQ.OrigNode,[2]:=RREQ.TargNode)}       |
       +-----------------------------------------------------------------+
       |          RteSeqNumTLV (OrigRtr.Seqnum, TargNode.Seqnum)         |
       +-----------------------------------------------------------------+
       |               Added Node Address Block (Optional)               |
       +-----------------------------------------------------------------+
       |                 Added Node Address TLV (SeqNum)                 |
       +-----------------------------------------------------------------+
       |           Added Node Address TLV (Metric[MetricType])           |
       +-----------------------------------------------------------------+
```

Figure 1: RREQ and RREP (RteMsg) message structure

Message Header
  This is typically mostly boilerplate but can contain MsgTLVs as below.

DestOnly TLV
  RREQ only: no Intermediate RREP.

MetricType TLV
  Metric Type for Metric AddrTLV

RteAddrBlk
  This Address Block contains the IP addresses for RREQ Originating and Target Node (OrigNode and TargNode). Note that for both RREP and RREQ, the OrigNode and TargNode are as identified in the context of the RREQ message originator.

RteSeqNumTLV (Sequence Number AddrTLV)
  This Address Block TLV is REQUIRED and carries the destination sequence numbers associated with either OrigNode or TargNode or both.

(Optional) Added Node AddrBlk
  AODVv2 allows the inclusion of routing information for other nodes in addition to OrigNode and TargNode.

(Optional) SeqNum AddrTLV If the Added Node AddrBlk is present, the SeqNum AddrTLV is REQUIRED, to carry the destination sequence numbers associated with the Added Nodes.

(Optional) Metric AddrTLV If the Added Node AddrBlk is present, this AddrTLV is REQUIRED, to carry the metric information associated with the Added Nodes. See Below.

The metric AddrTLV may be either a Metric8 AddrTLV or an Metric16 AddrTLV.

7.3. RREQ Generation

RREQ_Gen generates the RREQ according to the following steps, with order of protocol elements illustrated schematically in Figure 1.

1. RREQ_Gen MUST increment its OwnSeqNum by one (1) according to the rules specified in Section 5.5. This assures that all nodes with existing routing information will use RREQ_Gen's new information to update existing routing table information.

2. OrigNode MUST be a unicast address. If RREQ_Gen is not OrigNode, then OwnSeqNum will be used as the value of OrigNode.SeqNum. will be used by AODVv2 routers to create a route toward the OrigNode, enabling a RREP from TargRtr, and eventually used for proper forwarding of data packets.

3. If RREQ_Gen requires that only TargRtr is allowed to generate a RREP, then RREQ_Gen includes the "Destination RREP Only" TLV as part of the RFC 5444 message header. This also assures that TargRtr increments its sequence number. Otherwise, intermediate AODVv2 routers MAY respond to the RREQ_Gen's RREQ if they have an valid route to TargNode (see Section 13.2).

4. msg-hopcount MUST be set to 0.

   * This RFC 5444 constraint causes the typical RteMsg payload incur additional enlargement.

5. RREQ_Gen adds the TargNode.Addr to the RREQ.

6. If a previous value of the TargNode's SeqNum is known (e.g., from an invalid routing table entry using longest-prefix matching), RREQ_Gen SHOULD include TargNode.SeqNum in all but the last RREQ attempt. If TargNode.SeqNum is not included, it is assumed to be unknown by AODVv2 routers handling the RREQ; if the optional feature Intermediate RREP is enabled, then any route to TargNode will satisfy the RREQ [I-D.perkins-irrep].

7. RREQ_Gen adds OrigNode.Addr, its prefix, and the RREQ_Gen.SeqNum (OwnSeqNum) to the RREQ.

8. If OrigNode.Metric is included it is set to the cost of the route between OrigNode and RREQ_Gen.

An example RREQ message format is illustrated in Appendix A.1.

7.4. RREP Generation

An AODVv2 router (TargRtr, called in this section RREP_Gen) generates a RREP in order to provide a route to the Target Node (TargNode) of a RREQ, thus satisfying the routing requirement for packets to flow between OrigNode and TargNode. This section specifies the generation of an RREP by the RREP_Gen. The basic format of an RREP conforms to the structure for RteMsgs as illustrated in Figure 1. Optionally, RREP messages may be generated by AODVv2 routers other than TargRtr; this optional message generation is known as "Intermediate RREP" generation, and is specified in Internet Draft [I-D.perkins-irrep]. If TargNode is not a unicast IP address the RREP MUST NOT be generated, and processing for the RREQ is complete.

Otherwise RREP_Gen generates the RREP as follows:

1. RREP_Gen first uses the routing information to update its route table entry for OrigNode if necessary as specified in Section 6.2.

2. RREP_Gen MUST increment its OwnSeqNum by one (1) according to the rules specified in Section 5.5.

3. RREP.AddrBlk[OrigNode] := RREQ.AddrBlk[OrigNode]

4. RREP.AddrBlk[TargNode] := RREQ.AddrBlk[TargNode]

5. RREP.SeqNumTLV[OrigNode] := RREQ.SeqNumTLV[OrigNode]

6. RREP.SeqNumTLV[TargNode] := OwnSeqNum

7. If Route[TargNode].PfxLen/8 is equal to the number of bytes in the addresses of the RREQ (4 for IPv4, 16 for IPv6), then no <prefix-length> is included with the iRREP. Otherwise, RREP.PfxLen[TargNode] := RREQ.PfxLen[TargNode] according to the rules of RFC 5444 AddrBlk encoding.

8. RREP.MetricType[TargNode] := Route[TargNode].MetricType

9. RREP.Metric[TargNode] := Route[TargNode].Metric

10. <msg-hop-limit> SHOULD be set to RteMsg.<msg-hop-count>.

11. IP.DestinationAddr := Route[OrigNode].NextHop

The message format for RREP is illustrated in Appendix A.2.

7.5. Handling a Received RteMsg

Before an AODVv2 router (HandlingRtr) can process a received RteMsg (i.e., RREQ or RREP), it first must verify that the RteMsg is permissible according to the following steps. For RREQ, RteMsg_Gen is OrigRtr, also called RREQ_Gen. For RREP, RteMsg_Gen is TargRtr, also called RREP_Gen.

1. HandlingRtr MUST handle AODVv2 messages only from adjacent routers as specified in Section 5.4. AODVv2 messages from other sources MUST be disregarded.

2. If the RteMsg.<msg-hop-limit> is equal to 0, then the message is disregarded.

3. If the RteMsg.<msg-hop-count> is present, and RteMsg.<msg-hop-count> >= MAX_HOPCOUNT, then the message is disregarded.

4. HandlingRtr examines the RteMsg to ascertain that it contains the required information: TargNode.Addr, OrigNode.Addr, RteMsg_Gen.Metric and RteMsg_Gen.SeqNum. If the required information does not exist, the message is disregarded.

5. HandlingRtr checks that OrigNode.Addr and TargNode.Addr are valid routable unicast addresses. If not, the message is disregarded.

6. HandlingRtr checks that the Metric Type associated with OrigNode.Metric and TargNode.Metric is known, and that Cost(L) can be computed. If not, the message is disregarded.

   * DISCUSSION: alternatively, can change the AddrBlk metric to use HopCount, measured from<msg-hop-limit>.

7. If MAX_METRIC[RteMsg.MetricType] <= (RteMsg_Gen.Metric + Cost(L)), where 'L' is the incoming link, the RteMsg is disregarded.

An AODVv2 router (HandlingRtr) handles a permissible RteMsg according to the following steps.

1. HandlingRtr MUST process the routing information contained in the RteMsg as speciied in Section 6.1.

2. HandlingRtr MAY process AddedNode routing information (if present) as specified in Section 13.7.1 Otherwise, if AddedNode information is not processed, it MUST be deleted.

3. By sending the updated RteMsg, HandlingRtr advertises that it will route for addresses contained in the outgoing RteMsg based on the information enclosed. HandlingRtr MAY choose not to send the RteMsg, though not resending this RteMsg could decrease connectivity in the network or result in a nonoptimal path. The circumstances under which HandlingRtr might choose to not re-transmit a RteMsg are not specified in this document. Some examples might include the following:

   * HandlingRtr is already heavily loaded and does not want to advertise routing for the contained addresses

   * HandlingRtr recently transmitted identical routing information (e.g. in a RteMsg advertising the same metric)

   * HandlingRtr is low on energy and has to reduce energy expended for sending protocol messages or packet forwarding

   Unless HandlingRtr is prepared to send an updated RteMsg, it halts processing. Otherwise, processing continues as follows.

4. HandlingRtr MUST decrement RteMsg.<msg-hop-limit>. If RteMsg.<msg-hop-limit> is then zero (0), no further action is taken.

5. HandlingRtr MUST increment RteMsg.<msg-hop-count>.

Further actions to send an updated RteMsg depend upon whether the RteMsg is an RREP or an RREQ

7.5.1. Additional Handling for Outgoing RREQ

- If the upstream router is in the Blacklist, and Current_Time < BlacklistRmTime, then HandlingRtr MUST NOT transmit any outgoing RREQ, and processing is complete.

- Otherwise, if the upstream router is in the Blacklist, and Current_Time >= BlacklistRmTime, then the upstream router SHOULD be removed from the Blacklist, and message processing continued.

- If TargNode is a client of HandlingRtr, then a RREP is generated by the HandlingRtr (i.e., TargRtr) and unicast to the upstream router towards the RREQ OrigNode, as specified in Section 7.4. Afterwards, TargRtr processing for the RREQ is complete.

- If HandlingRtr is not the TargetNode, then the outgoing RREQ (as altered by the procedure defined above) SHOULD be sent to the IP multicast address LL-MANET-Routers [RFC5498]. If the RREQ is unicast, the IP.DestinationAddress is set to the NextHopAddress.

7.5.2. Additional Handling for Outgoing RREP

- If HandlingRtr is not OrigRtr then the outgoing RREP is sent to the Route.NextHopAddress for the RREP.AddrBlk[OrigNode]. If no forwarding route exists to OrigNode, then a RERR SHOULD be transmitted to RREP.AddrBlk[TargNode]. See Table 1 for notational conventions; OrigRtr, OrigNode, and TargNode are routers named in the context of OrigRtr, that is, the router originating the RREQ to which the RREP is responding.

8. Route Maintenance

AODVv2 routers attempt to maintain active routes. When a routing problem is encountered, an AODVv2 router (namely, RERR_Gen) attempts to quickly notify upstream routers. Two kinds of routing problems may trigger generation of a RERR message. The first case happens when the router receives a packet but does not have a route for the destination of the packet. The second case happens immediately upon detection of a broken link (see Section 8.2) of an Active route, to quickly notify AODVv2 routers that that route is no longer available. When the RERR message is generated, it MUST be the only message in the RFC 5444 packet.

8.1. Handling Route Lifetimes During Packet Forwarding

Before using a route to forward a packet, an AODVv2 router MUST check the status of the route as follows.

If the route is marked has been marked as Broken, it cannot be used for forwarding.

If Current_Time > Route.ExpirationTime, the route table entry has expired, and a RERR SHOULD be generated.

Similarly, if (Route.ExpirationTime == MAXTIME), and if Current_Time - Route.LastUsed > (ACTIVE_INTERVAL+MAX_IDLETIME), the route has expired, and a RERR SHOULD be generated.

Furthermore, if Current_Time - Route.LastUsed > (MAX_SEQNUM_LIFETIME), the route table entry MUST be expunged.

Otherwise, if none of the above route error conditions are indicated, Route.LastUsed := Current_Time, and the packet is forwarded to the route’s next hop.

Optionally, if a precursor list is maintained for the route, see Section 13.3 for precursor lifetime operations.

8.2. Active Next-hop Router Adjacency Monitoring

Nodes SHOULD monitor connectivity to adjacent next-hop AODVv2 routers on forwarding routes. This monitoring can be accomplished by one or several mechanisms, including:

- o Neighborhood discovery [RFC6130]
- o Route timeout
- o Lower layer trigger that a neighboring router is no longer reachable
- o Other monitoring mechanisms or heuristics

Upon determining that a next-hop AODVv2 router has become unreachable, RERR_Gen follows the procedures specified in Section 8.3.2.

8.3. RERR Generation

An RERR message is generated by a AODVv2 router (in this section, called RERR_Gen) in order to to notify upstream routers that packets cannot be delivered to certain destinations. An RERR message has the following general structure:

```
+-----------------------------------------------------------------+
|                      RFC 5444 Packet Header                     |
+-----------------------------------------------------------------+
|     RFC 5444 Message Header <msg-hoplimit> <msg-hopcount>       |
+-----------------------------------------------------------------+
|      UnreachableNode AddrBlk (Unreachable Node addresses)       |
+-----------------------------------------------------------------+
|               UnreachableNode SeqNum AddrBlk TLV                |
+-----------------------------------------------------------------+
```

Figure 2: RERR message structure

Message Header
RFC 5444 MsgHdr may contain the following options:

* <msg-hop-limit>

* <msg-hop-count>

* PktSource MsgTLV

UnreachableNode AddrBlk
This Address Block contains the IP addresses unreachable by AODVv2 router transmitting the RERR.

Sequence Number AddrBlk TLV
This Address Block TLV carries the destination sequence number associated with the UnreachableNodes when that information is available.

UnreachableNode.PfxLen
The prefix length associated with an UnreachableNode.

There are two kinds of events indicating that packets cannot be delivered to certain destinations. The two cases differ in the way that the neighboring IP destination address for the RERR (i.e., RERR_dest) is chosen, and in the way that the set of UnreachableNodes is identified.

In both cases, the MsgHdr.<msg-hop-limit> MUST be set to MAX_HOPCOUNT. MsgHdr.<msg-hop-count> SHOULD be be included and set to 0, to facilitate use of various route repair strategies including Intermediate RREP [I-D.perkins-irrep].

8.3.1. Case 1: Undeliverable Packet

The first case happens when the router receives a packet but does not have a valid route for the destination of the packet. In this case, there is exactly one UnreachableNode to be included in the RERR's AddrBlk. RERR_dest SHOULD be the multicast address LL-MANET-Routers, but RERR_Gen MAY instead set RERR_dest to be the next hop towards the source IP address of the packet which was undeliverable. In the latter case, the PktSource MsgTLV MUST be included, containing the the source IP address of the undeliverable packet. If a value for the UnreachableNode's SeqNum (UnreachableNode.SeqNum) is known, it MUST be placed in the RERR. Otherwise, if no Seqnum AddrTLV is included, all nodes handling the RERR will assume their route through RERR_Gen towards the UnreachableNode is no longer valid and flag those routes as broken. RERR_Gen MUST discard the packet or message that triggered generation of the RERR.

8.3.2. Case 2: Broken Link

The second case happens when the link breaks to an active downstream neighbor (i.e., the next hop of an active route). In this case, RERR_dest MUST be the multicast address LL-MANET-Routers, except when the optional feature of maintaining precursor lists is used as specified in Section 13.3. All Active, Idle and Expired routes that use the broken link MUST be marked as Broken. The set of UnreachableNodes is initialized by identifying those Active routes which use the broken link. For each such Active Route, Route.Dest is added to the set of Unreachable Nodes. After the Active Routes using the broken link have all been included as UnreachableNodes, idle routes MAY also be included, as long as the packet size of the RERR does not exceed the MTU of the physical medium.

If the set of UnreachableNodes is empty, no RERR is generated. Otherwise, RERR_Gen generates a new RERR, and the address of each UnreachableNode (IP.DestinationAddress from a data packet or RREP.TargNode.Address) is inserted into an AddrBlock. If a prefix is known for the UnreachableNode.Address, it SHOULD be included. Otherwise, the UnreachableNode.Address is assumed to be a host address with a full length prefix. The value for each UnreachableNode's SeqNum (UnreachableNode.SeqNum) MUST be placed in a SeqNum AddrTLV. If none of UnreachableNode.Addr entries are associated with known prefix lengths, then the AddrBLK SHOULD NOT include any prefix-length information. Otherwise, for each UnreachableNode.Addr that does not have any associated prefix-length information, the prefix-length for that address MUST be assigned to zero.

8.4. Receiving and Handling RERR Messages

When an AODVv2 router (HandlingRtr) receives a RERR message, it uses the information provided to invalidate affected routes. If the information in the RERR may be useful to upstream neighbors using those routes, HandlingRtr subsequently sends another RERR to those neighbors. This operation has the effect of retransmitting the RERR information and is counted as another "hop" for purposes of properly modifying Msg.<msg-hop-limit> and Msg.<msg-hop-count>.

HandlingRtr examines the incoming RERR to assure that it contains Msg.<msg-hop-limit> and at least one UnreachableNode.Address. If the required information does not exist, the incoming RERR message is disregarded and further processing stopped. Otherwise, for each UnreachableNode.Address, HandlingRtr searches its route table for a route using longest prefix matching. If no such Route is found, processing is complete for that UnreachableNode.Address. Otherwise, HandlingRtr verifies the following:

1. The UnreachableNode.Address is a routable unicast address.

2. Route.NextHopAddress is the same as RERR IP.SourceAddress.

3. Route.NextHopInterface is the same as the interface on which the RERR was received.

4. The UnreachableNode.SeqNum is unknown, OR Route.SeqNum <= UnreachableNode.SeqNum (using signed 16-bit arithmetic).

If the route satisfies all of the above conditions, HandlingRtr sets the Route.Broken flag for that route. Furthermore, if Msg.<msg-hop-limit> is greater than 0, then HandlingRtr adds the UnreachableNode address and TLV information to an AddrBlk for for delivery in the outgoing RERR message to one or more of HandlingRtr's upstream neighbors.

If there are no UnreachableNode addresses to be transmitted in an RERR to upstream routers, HandlingRtr MUST discard the RERR, and no further action is taken.

Otherwise, Msg.<msg-hop-limit> is decremented by one (1) and processing continues as follows:

- If precursor lists are (optionally) maintained, the outgoing RERR SHOULD be sent to the active precursors of the broken route as specified in Section 13.3.

- Otherwise, if the incoming RERR message was received at the LL-MANET-Routers [RFC5498] multicast address, the outgoing RERR SHOULD also be sent to LL-MANET-Routers.

- Otherwise, if the PktSource MsgTLV is present, and HandlingRtr has a Route to PktSource.Addr, then HandlingRtr MUST send the outgoing RERR to Route[PktSource.Addr].NextHop.

- Otherwise, the outgoing RERR MUST be sent to LL-MANET-Routers.

9. Unknown Message and TLV Types

If a message with an unknown type is received, the message is disregarded.

For handling of messages that contain unknown TLV types, ignore the information for processing, preserve it unmodified for forwarding.

10. Simple Internet Attachment

Simple Internet attachment means attachment of a stub (i.e., non-transit) network of AODVv2 routers to the Internet via a single Internet AODVv2 router (called IAR).

As in any Internet-attached network, AODVv2 routers, and their clients, wishing to be reachable from hosts on the Internet MUST have IP addresses within the IAR's routable and topologically correct prefix (e.g. 191.0.2.0/24).

The IAR is responsible for generating RREQ messages to find nodes within the MANET on behalf of nodes on the Internet, as well as responding to route requests from the AODVv2 MANET on behalf of the nodes on the Internet.

```
       /-------------------------\
      / +----------------+        \
     /  |  AODVv2 Router |         \
     |  |  191.0.2.2/32  |         |
     |  +----------------+         |            Routable
     |                       +-----+--------+   Prefix
     |                       |   Internet   |  /191.0.2/24
     |                       | AODVv2 Router| /
     |                       |  191.0.2.1   |/       /----------------\
     |                       | serving net  +-------+     Internet     \
     |                       |  191.0.2/24  |       \                 /
     |                       +-----+--------+        \----------------/
     |         +----------------+  |
     |         |  AODVv2 Router |  |
     |         |  191.0.2.3/32  |  |
     \         +----------------+  /
      \                           /
       \-------------------------/
```

Figure 3: Simple Internet Attachment Example

When an AODVv2 router within the AODVv2 MANET wants to discover a route toward a node on the Internet, it uses the normal AODVv2 route discovery for that IP Destination Address. The IAR MUST respond to RREQ on behalf of all Internet destinations.

When a packet from a node on the Internet destined for a node in the AODVv2 MANET reaches the IAR, if the IAR does not have a route toward that destination it will perform normal AODVv2 route discovery for that destination.

11. Multiple Interfaces

AODVv2 may be used with multiple interfaces; therefore, the particular interface over which packets arrive MUST be known whenever a packet is received. Whenever a new route is created, the interface through which the Route.Address can be reached is also recorded in the route table entry.

When multiple interfaces are available, a node transmitting a multicast packet with IP.DestinationAddress set to LL-MANET-Routers SHOULD send the packet on all interfaces that have been configured for AODVv2 operation.

Similarly, AODVv2 routers SHOULD subscribe to LL-MANET-Routers on all their AODVv2 interfaces.

12. AODVv2 Control Packet/Message Generation Limits

To avoid messaging overload, each AODVv2 router's rate of packet/message generation SHOULD be limited. The rate and algorithm for limiting messages (CONTROL_TRAFFIC_LIMITS) is left to the implementor and should be administratively configurable. AODVv2 messages SHOULD be discarded in the following order of preference: RREQ, RREP, and finally RERR.

13. Optional Features

Some optional features of AODVv2, associated with AODV, are not required by minimal implementations. These features are expected to be useful in networks with greater mobility, or larger node populations, or requiring shorter latency for application launches. The optional features are as follows:

- Expanding Rings Multicast
- Intermediate RREPs (iRREPs): Without iRREP, only the destination can respond to a RREQ.
- Precursor lists.
- Reporting Multiple Unreachable Nodes. An RERR message can carry more than one Unreachable Destination node for cases when a single link breakage causes multiple destinations to become unreachable from an intermediate router.

- RREP_ACK.
- Message Aggregation.
- Inclusion of Added Routing Information.

13.1. Expanding Rings Multicast

For multicast RREQ, Msg.<msg-hop-limit> MAY be set in accordance with an expanding ring search as described in [RFC3561] to limit the RREQ propagation to a subset of the local network and possibly reduce route discovery overhead.

13.2. Intermediate RREP

This specification has been published as a separate Internet Draft [I-D.perkins-irrep].

13.3. Precursor Lists and Notifications

This section specifies an interoperable enhancement to AODVv2 (and possibly other reactive routing protocols) enabling more economical notifications to active sources of traffic upon determination that a route needed to forward such traffic to its destination has become Broken.

13.3.1. Overview

In many circumstances, there might be several sources of traffic for any particular destination. Each such source of traffic is known as a "precursor" for the destination, as well as all upstream routers between the forwarding AODVv2 router and the traffic source. For each active destination, an AODVv2 router MAY choose to keep track of the upstream neighbors that have provided traffic for that destination; there is no need to keep track of upstream routers any farther away than the next hop.

Moreover, any particular link to an adjacent AODVv2 router may be a path component of multiple routes towards various destinations. The precursors for all destinations using the next hop across any link are collectively known as the precursors for that next hop.

When an AODVv2 router determines that an active link to one of its downstream neighbors has broken, the AODVv2 router detecting the broken link must mark multiple routes as Broken, for each of the newly unreachable destinations, as described in Section 8.3. Each route that relies on the newly broken link is no longer valid. Furthermore, the precursors of the broken link should be notified (using RERR) about the change in status of their route to a destination downstream along the broken next hop.

13.3.2. Precursor Notification Details

During normal operation, each AODVv2 router wishing to maintain precursor lists as described above, maintains a precursor table and updates the table whenever the node forwards traffic to one of the destinations in its route table. For each precursor in the precursor list, a record must be maintained to indicate whether the precursor has been used for recent traffic (in other words, whether the precursor is an Active precursor). So, when traffic arrives from a precursor, the Current_Time is used to mark the time of last use for the precursor list element associated with that precursor.

When an AODVv2 router detects that a link is broken, then for each precursor using that next hop, the node MAY notify the precursor using either unicast or multicast RERR:

unicast RERR to each Active precursor
: This option is useful when there are few Active precursors compared to the number of neighboring AODVv2 routers.

multicast RERR to RERR_PRECURSORS
: RERR_PRECURSORS is, by default, LL-MANET-Routers [RFC5498]. This option is typically preferable since fewer packet transmissions are required.

Each active upstream neighbor (i.e., precursor) MAY then execute the same procedure until all active upstream routers have received the RERR notification.

13.4. Multicast RREP Response to RREQ

The RREQ Target Router (RREP_Gen) MAY, as an alternative to unicasting a RREP, be configured to distribute routing information about the route toward the RREQ TargNode (TargRtr's client) more widely. That is, RREP_Gen MAY be configured respond to a route discovery by generating a RREP, using the procedure in Section 7.4, but multicasting the RREP to LL-MANET-Routers [RFC5498]. Afterwards, RREP_Gen processing for the incoming RREQ is complete.

Broadcast response to incoming RREQ was originally specified to handle unidirectional links, but it is expensive. Due to the significant overhead, AODVv2 routers MUST NOT use multicast RREP unless configured to do so by setting the administrative parameter USE_MULTICAST_RREP.

13.5. RREP_ACK

Instead of relying on existing mechanisms for requesting verification of link bidirectionality during Route Discovery, RREP_Ack is provided as an optional feature and modeled on the RREP_Ack message type from AODV [RFC3561].

Since the RREP_ACK is simply echoed back to the node from which the RREP was received, there is no need for any additional RFC 5444 address information (or TLVs). Considerations of packet TTL are as specified in Section 5.4. The message format is illustrated in section Appendix A.4.

13.6. Message Aggregation

The aggregation of multiple messages into a packet is specified in RFC 5444 [RFC5444].

Implementations MAY choose to briefly delay transmission of messages for the purpose of aggregation (into a single packet) or to improve performance by using jitter [RFC5148].

13.7. Added Routing Information in RteMsgs

DSR [RFC4728] includes source routes as part of the data of its RREPs and RREQs. Doign so allows additional topology information to be multicast along with the RteMsg, and potentially allows updating for stale routing information at MANET routers along new paths between source and destination. To maintain this functionality, AODVv2 has defined a somewhat more general method that enables inclusion of source routes in RteMsgs.

Appending routing information can eliminate some route discovery attempts to the nodes whose information is included, if handling AODVv2 routers use this information to update their routing tables.

Note that, since the initial merger of DSR with AODV to create this protocol, further experimentation has shown that including the additional routing information is not always helpful. Sometimes it seems to help, and other times it seems to reduce overall performance. The results depend upon packet size and traffic patterns.

13.7.1. Including Added Node Information

An AODVv2 router (HandlingRtr) MAY optionally append AddedNode routing information to a RteMsg. This is controllable by an option (APPEND_INFORMATION) which SHOULD be administratively configurable or controlled according to the traffic characteristics of the network.

The following notation is used to specify the methods for inclusion of routing information for addtional nodes.

AddedNode
The IP address of an additional node that can be reached via the AODVv2 router adding this information. Each AddedNode.Address MUST include its prefix. Each AddedNode.Address MUST also have an associated Node.SeqNum in the address TLV block.

AddedNode.SeqNum
The AODVv2 sequence number associated with this routing information.

AddedNode.Metric
The cost of the route needed to reach the associated AddedNode.Address. This field is increased by Cost(L) at each intermediate AODVv2 router, where 'L' is the incoming link. If, for the Metric Type of the AddrBlk, it is not known how to compute Cost(L), the AddedNode.Addr information MUST be deleted from the AddedNode AddrBlk.

The VALIDITY_TIME of routing information for appended address(es) MUST be included, to inform routers about when to expire this information. A typical value for VALIDITY_TIME is (ACTIVE_INTERVAL+ MAX_IDLETIME) - (Current_Time - Route.LastUsed) but other values (less than MAX_SEQNUM_TIME) MAY be chosen. The VALIDITY_TIME TLV is defined in [RFC5497].

SeqNum and Metric AddrTLVs about any appended address(es) MUST be included.

Routing information about the TargNode MUST NOT be added to the AddedAddrBlk. Also, duplicate address entries SHOULD NOT be added. Only the best routing information (Section 6.1) for a particular address SHOULD be included; if route information is included for a destination address already in the AddedAddrBlk, the previous information SHOULD NOT be included in the outgoing RteMsg.

13.7.2. Handling Added Node Information

An intermediate node (i.e., HandlingRtr) obeys the following procedures when processing AddedNode.Address information and other associated TLVs that are included with a RteMsg. For each AddedNode (except the TargetNode) in the RteMsg, the AddedNode.Metric information MUST be increased by Cost(L), where 'L' is the incoming link. If, for the Metric Type of the AddrBlk, it is not known how to compute Cost(L), the AddedNode.Addr information MUST be deleted from the AddedNode AddrBlk. If the resulting Cost of the route to the AddedNode is greater than MAX_METRIC[i], the AddedNode information is discarded. If the resulting Distance value for another node is greater than MAX_METRIC[i], the associated address and its information are removed from the RteMsg.

After handling the OrigNode's routing information, then each address that is not the TargetNode MAY be considered for creating and updating routes. Creating and updating routes to other nodes can eliminate RREQ for those IP destinations, in the event that data needs to be forwarded to the IP destination(s) now or in the near future.

For each of the additional addresses considered, HandlingRtr first checks that the address is a routable unicast address. If the address is not a unicast address, then the address and all related information MUST be removed.

If the routing table does not have a matching route with a known Route.SeqNum for this additional address using longest-prefix matching, then a route MAY be created and updated as described in Section 6.2. If a route table entry exists with a known Route.SeqNum, the incoming routing information is compared with the route table entry following the procedure described in Section 6.1. If the incoming routing information is used, the route table entry SHOULD be updated as described in Section 6.2.

If the routing information for an AddedNode.Address is not used, then it is removed from the RteMsg.

If route information is included for a destination address already in the AddedAddrBlk, the previous information SHOULD NOT be included in the outgoing RteMsg.

14. Administratively Configured Parameters and Timer Values

AODVv2 contains several parameters which MUST be administratively configured. The list of these follows:

Name	Description
CLIENT_ADDRESSES	List of addresses and routing prefixes, for which this AODVv2 router is responsible. If the list is empty, this AODVv2 router is only responsible for its own addresses.
USE_MULTICAST_RREP	Whether or not to use multicast RREP (see Section 13.4).
DEFAULT_METRIC_TYPE	3 (Hop Count {see [RFC6551]}
AODVv2_INTERFACES	List of the interfaces participating in AODVv2 routing protocol.

Table 2: Required Administratively Configured Parameters

AODVv2 requires certain timing information to be associated with route table entries. The default values are as follows:

Name	Value
ACTIVE_INTERVAL	5 second
MAX_IDLETIME	200 seconds
MAX_SEQNUM_LIFETIME	300 seconds
ROUTE_RREQ_WAIT_TIME	2 seconds
UNICAST_MESSAGE_SENT_TIMEOUT	1 second
RREQ_HOLDDOWN_TIME	10 seconds

Table 3: Default Timing Parameter Values

The above timing parameter values have worked well for small and medium well-connected networks with moderate topology changes.

The timing parameters SHOULD be administratively configurable for the network where AODVv2 is used. Ideally, for networks with frequent topology changes the AODVv2 parameters should be adjusted using either experimentally determined values or dynamic adaptation. For example, in networks with infrequent topology changes MAX_IDLETIME may be set to a much larger value.

Name	Value	Description
MAX_HOPCOUNT	20 hops	This value MUST be larger than the AODVv2 network diameter. Otherwise, routing messages may not reach their intended destinations.
MAX_METRIC[i]	Not Specified in This Document	If defined, this is the maximum permissible value for Metric Type 'i' (see [RFC6551]).
MAXTIME	TBD	The maximum expressible value for clock time.
DISCOVERY_ATTEMPTS_MAX	3	The number of route discovery attempts to make before indicating that a particular address is not reachable.
MTU	TBD -- depends on address family	Determines the maximum number of RFC 5444 AddrBlk entries

Table 4: Default Parameter Values

In addition to the above parameters and timing values, several administrative options exist. These options have no influence on correct routing behavior, although they may potentially reduce AODVv2 protocol messaging in certain situations. The default behavior is to NOT enable any of these options; and although many of these options can be administratively controlled, they may be better served by intelligent control. The following table enumerates several of the options.

Name	Description
APPEND_INFORMATION	Whether or not appending routing information for AddedNodes to a RteMsg is enabled.
BUFFER_SIZE_PACKETS	2
BUFFER_SIZE_BYTES	MAX_PACKET_SIZE [TBD]
APPEND_IDLE_UNREACHABLE	Whether to append Unreachable information about idle routes to RERR.

CONTROL_TRAFFIC_LIMIT	TBD [50 msgs/sec?]

Table 5: Administratively Controlled Options

Note: several fields have limited size (bits or bytes). These sizes and their encoding may place specific limitations on the values that can be set. For example, MsgHdr.<msg-hop-count> is a 8-bit field and therefore MAX_HOPCOUNT cannot be larger than 255.

15. IANA Considerations

This section specifies several message types, message tlv-types, and address tlv-types. Also, a new registry of 16-bit alternate metric types is specified.

15.1. AODVv2 Message Types Specification

Name	Type
Route Request (RREQ)	10 - TBD
Route Reply (RREP)	11 - TBD
Route Error (RERR)	12 - TBD
Route Reply Acknowledgement (RREP_ACK)	13 - TBD

Table 6: AODVv2 Message Types

15.2. Message and Address Block TLV Type Specification

Name	Type	Length	Value
Unicast Response Request (RespReq)	10 - TBD	0 octets	Indicates to the handling (receiving) AODVv2 router that the previous hop (IP.SourceAddress) expects a unicast reply message within UNICAST_MESSAGE_SENT_TIMEOUT. --
Destination RREP Only (DestOnly)	11 - TBD	0 octets	Indicates that intermediate RREPs are prohibited. --

Packet source IP address (PktSource)	12 - TBD	4 or 16 octets	Provides the IP address for RERR messages generated due to inability to deliver a packet. --
Metric Type	13 - TBD	1 octet	Type of metric in the Metric8 or Metric16 AddrTLV.

Table 7: Message TLV Types

15.3. Address Block TLV Specification

Name	Type	Length	Value
VALIDITY_TIME	1[RFC5497]	1 octet	The maximum amount of time that information can be maintained before being deleted. The VALIDITY_TIME TLV is defined in [RFC5497]. --
Sequence Number (SeqNum)	10 - TBD	2 octets	The latest AODVv2 sequence number associated with the address.
Metric8	11 - TBD	1 octet	8-bit Cost of the route to reach the destination address.
Metric16	12 - TBD	2 octets	16-bit Cost of the route to reach the destination address.

Table 8: Address Block TLV (AddrTLV) Types

The same number space should be used for both Metric8 and Metric16 metric types.

15.4. Metric Type Number Allocation

Metric types are identified according to the assignments as specified in [RFC6551]. The metric type of the Hop Count metric is assigned to be 3, in order to maintain compatibility with that existing table of values from RFC 6551. If non-additive metrics are to be used, the specification for assessing the usability of route updates (see Section 6.1) may require changes.

Name	Type	Size
Reserved	0	Undefined
Unallocated	1 -- 2	TBD
Hop Count	3 - TBD	1 octet
Unallocated	4 -- 254	TBD
Reserved	255	Undefined

Table 9: Metric Types

16. Security Considerations

The objective of the AODVv2 protocol is for each router to communicate reachability information about addresses for which it is responsible. Positive routing information (i.e. a route exists) is distributed via RteMsgs and negative routing information (i.e. a route does not exist) via RERRs. AODVv2 routers that handle these messages store the contained information to properly forward data packets, and they generally provide this information to other AODVv2 routers.

This section does not mandate any specific security measures. Instead, this section describes various security considerations and potential avenues to secure AODVv2 routing.

The most important security mechanisms for AODVv2 routing are integrity/authentication and confidentiality.

In situations where routing information or router identity are suspect, integrity and authentication techniques SHOULD be applied to AODVv2 messages. In these situations, routing information that is distributed over multiple hops SHOULD also verify the integrity and identity of information based on originator of the routing information.

A digital signature could be used to identify the source of AODVv2 messages and information, along with its authenticity. A nonce or timestamp SHOULD also be used to protect against replay attacks. S/MIME and OpenPGP are two authentication/integrity protocols that could be adapted for this purpose.

In situations where confidentiality of AODVv2 messages is important, cryptographic techniques can be applied.

In certain situations, for example sending a RREP or RERR, an AODVv2 router could include proof that it has previously received valid routing information to reach the destination, at one point of time in the past. In situations where routers are suspected of transmitting maliciously erroneous information, the original routing information along with its security credentials SHOULD be included.

Note that if multicast is used, any confidentiality and integrity algorithms used MUST permit multiple receivers to handle the message.

Routing protocols, however, are prime targets for impersonation attacks. In networks where the node membership is not known, it is difficult to determine the occurrence of impersonation attacks, and security prevention techniques are difficult at best. However, when the network membership is known and there is a danger of such attacks, AODVv2 messages must be protected by the use of authentication techniques, such as those involving generation of unforgeable and cryptographically strong message digests or digital signatures. While AODVv2 does not place restrictions on the authentication mechanism used for this purpose, IPsec Authentication Message (AH) is an appropriate choice for cases where the nodes share an appropriate security association that enables the use of AH.

In particular, routing messages SHOULD be authenticated to avoid creation of spurious routes to a destination. Otherwise, an attacker could masquerade as that destination and maliciously deny service to the destination and/or maliciously inspect and consume traffic intended for delivery to the destination. RERR messages SHOULD be authenticated in order to prevent malicious nodes from disrupting active routes between communicating nodes.

If the mobile nodes in the ad hoc network have pre-established security associations, the purposes for which the security associations are created should include that of authorizing the processing of AODVv2 control packets. Given this understanding, the mobile nodes should be able to use the same authentication mechanisms based on their IP addresses as they would have used otherwise.

If the mobile nodes in the ad hoc network have pre-established security associations, the purposes for which the security associations Most AODVv2 messages are transmitted to the multicast address LL-MANET-Routers [RFC5498]. It is therefore required for security that AODVv2 neighbors exchange security information that can be used to insert an ICV [RFC6621] into the AODVv2 message block [RFC5444]. This enables hop-by-hop security, which is proper for these message types that may have mutable fields. For destination-only RREP discovery procedures, AODVv2 routers that share a security association SHOULD use the appropriate mechanisms as specified in RFC 6621. The establishment of these security associations is out of scope for this document.

17. Acknowledgments

AODVv2 is a descendant of the design of previous MANET on-demand protocols, especially AODV [RFC3561] and DSR [RFC4728]. Changes to previous MANET on-demand protocols stem from research and implementation experiences. Thanks to Elizabeth Belding-Royer for her long time authorship of AODV. Additional thanks to Luke Klein-Berndt, Pedro Ruiz, Fransisco Ros, Henning Rogge, Koojana Kuladinithi, Ramon Caceres, Thomas Clausen, Christopher Dearlove, Seung Yi, Romain Thouvenin, Tronje Krop, Henner Jakob, Alexandru Petrescu, Christoph Sommer, Cong Yuan, Lars Kristensen, and Derek Atkins for reviewing of AODVv2, as well as several specification suggestions.

This revision of AODVv2 separates the minimal base specification from other optional features to expedite the process of assuring compatibility with the existing LOADng specification [I-D.clausen-lln-loadng] (minimal reactive routing protocol specification). Thanks are due to T. Clausen, A. Colin de Verdiere, J. Yi, A. Niktash, Y. Igarashi, Satoh. H., and U. Herberg for their development of LOADng and sharing details for assuring appropriateness of AODVv2 for their application.

18. References

18.1. Normative References

[RFC1812] Baker, F., "Requirements for IP Version 4 Routers", RFC 1812, June 1995.

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

[RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, October 2007.

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

[RFC5497] Clausen, T. and C. Dearlove, "Representing Multi-Value Time in Mobile Ad Hoc Networks (MANETs)", RFC 5497, March 2009.

[RFC5498] Chakeres, I., "IANA Allocations for Mobile Ad Hoc Network (MANET) Protocols", RFC 5498, March 2009.

[RFC6551] Vasseur, JP., Kim, M., Pister, K., Dejean, N., and D. Barthel, "Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks", RFC 6551, March 2012.

18.2. Informative References

[I-D.clausen-lln-loadng]
Clausen, T., Verdiere, A., Yi, J., Niktash, A., Igarashi, Y., Satoh, H., Herberg, U., Lavenu, C., Lys, T., Perkins, C., and J. Dean, "The Lightweight On-demand Ad hoc Distance-vector Routing Protocol - Next Generation (LOADng)", draft-clausen-lln-loadng-06 (work in progress), October 2012.

[I-D.perkins-irrep]
Perkins, C. and I. Chakeres, "Intermediate RREP for dynamic MANET On-demand (AODVv2) Routing", draft-perkins-irrep-02 (work in progress), November 2012.

[Perkins99]
Perkins, C. and E. Belding-Royer, "Ad hoc On-Demand Distance Vector (AODV) Routing", Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, pp. 90-100, February 1999.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[RFC2501] Corson, M. and J. Macker, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations", RFC 2501, January 1999.

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

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.

[RFC4728] Johnson, D., Hu, Y., and D. Maltz, "The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4", RFC 4728, February 2007.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC5148] Clausen, T., Dearlove, C., and B. Adamson, "Jitter Considerations in Mobile Ad Hoc Networks (MANETs)", RFC 5148, February 2008.

[RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, July 2008.

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

[RFC6549] Lindem, A., Roy, A., and S. Mirtorabi, "OSPFv2 Multi-Instance Extensions", RFC 6549, March 2012.

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

Appendix A. Example RFC 5444-compliant packet formats

The following three subsections show example RFC 5444-compliant packets for AODVv2 message types RREQ, RREP, and RERR. These proposed message formats are designed based on expected savings from IPv6 addressable MANET nodes, and a layout for the Address TLVs that may be viewed as natural, even if perhaps not the absolute most compact possible encoding.

For RteMsgs, the msg-hdr fields are followed by at least one and optionally two Address Blocks. The first AddrBlk contains OrigNode and TargNode. For each AddrBlk, there must be AddrTLVs of type Seqnum and of type Metric.

In addition to the Seqnum TLV, there MUST be an AddrTLV of type Metric. The msg-hop-count is counts the number of hops followed by the RteMsg from RteMsg_Orig to the current intermediate AODVv2 router handling the RteMsg. Alternate metrics are enabled by the inclusion of the MetricType MsgTLV. When there is no such MetricType MsgTLV present, then the Metric AddrTLV measures HopCount. The Metric AddrTLV also provides a way for the RteMsg_Orig to supply an initial nonzero cost for the route between the RteMsg_Orig and its client node, i.e., either OrigNode or TargNode.

AddedNode information MAY be included in a RteMsg by adding a second AddrBlk. Both Metric AddrTLVs use the same Metric Type.

A.1. RREQ Message Format

The figure below illustrates a packet format for an example RREQ message.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| PV=0 |  PF=0  | msg-type=RREQ | MF=4  | MAL=3 |  msg-size=24  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  msg-size=24  | msg-hop-limit |      msg.tlvs-length=0        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   num-addr=2  |1|0|0|0|0| Rsv | head-length=3 |Head(Orig&Targ)|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Head (bytes for Orig & Target)|   Orig.Tail   |  Target.Tail  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      addr.tlvs-length=11      |  type=SeqNum  |0|1|0|1|0|0|Rsv|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Index-start=0 | tlv-length=2  |     Orig.Node Sequence #      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  type=SeqNum  |0|1|0|1|0|0|Rsv| Index-start=0 | tlv-length=1  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| OrigNodeHopCt |
+-+-+-+-+-+-+-+-+
```

RREQ with SeqNum and Metric AddrTLVs added, and: - two addresses in Address Block - address length = 4 [IPv4], shared initial bytes = 3 - Sequence Number available only for Orig.Node in addr.tlv - Hop Count available only for Orig.Node in Metric8 AddrTLV - Addresses stored in the order OrigNode, TargNode

Figure 4: Example IPv4 RREQ

A.2. RREP Message Format

The figure below illustrates a packet format for an example RREP message.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| PV=0 |  PF=0  | msg-type=RREP | MF=4  | MAL=3 |  msg-size=30  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  msg-size=30  | msg-hop-limit |      msg.tlvs-length=0        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   num-addr=2  |1|0|0|0|0| Rsv | head-length=3 |Head(Orig&Targ)|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Head (bytes for Orig & Target)|   Orig.Tail   |  Target.Tail  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      addr.tlvs-length=13      |  type=SeqNum  |0|1|0|1|0|0|Rsv|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Index-start=0 | tlv-length=2  |     Orig.Node Sequence #      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Target.Node Sequence #     |  type=Metric8 |0|1|0|1|0|0|Rsv|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Index-start=1 | tlv-length=1  | TargNodeHopCt |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

RREP with SeqNum and Metric AddrTLVs added, and: - two addresses in AddrBlk - address length = 4 [IPv4], shared initial bytes = 3 - One Sequence Number (for TargNode) in SeqNum AddrTLV - Hop Count available only for Targ.Node in Metric8 AddrTLV - Addresses stored in the order OrigNode, TargNode

Figure 5: Example IPv4 RREP

A.3. RERR Message Format

The figure below illustrates a packet format for an example RERR message.

```
    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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | PV=0 |  PF=0  | msg-type=RERR | MF=4  | MAL=3 |  msg-size=25  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  msg-size=25  | msg-hop-limit |      msg.tlvs-length=0        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |   num-addr=2  |1|0|0|0|0| Rsv | head-length=3 |Head(Two Dests)|
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Head (for both destinations)  |  Tail(Dest_1) | Tail(Dest_2)  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      addr.tlvs-length=8       |  type=SeqNum  |0|1|0|1|0|0|Rsv|
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Index-start=0 | tlv-length=2  |        Dest_1 Sequence #      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        Dest_2 Sequence #      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

RERR with - Two Unreachable Node address in Address Block - address length = 4 [IPv4], shared initial bytes = 3 - Two Sequence Numbers available in addr.tlv - Addresses stored from Originator to Target

Figure 6: Example IPv4 RERR

A.4. RREP_ACK Message Format

The figure below illustrates a packet format for an example RREP_ACK message.

```
    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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | PV=0 |  PF=0  |msg-type=RREPAk| MF=0  | MAL=3 |  msg-size=3   |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  msg-size=3   |
   +-+-+-+-+-+-+-+-+
```

RREP_ACK - address length = 4 [IPv4]

Figure 7: Example IPv4 RREP_ACK

Appendix B. Changes since revision ...-21.txt

The revisions of this document that were numbered 22 and 23 were produced without sufficient time for preparation, and suffered from numerous editorial errors. Therefore, this list of changes is enumerated based on differences between this revision (24) and revision 21.

- o Alternate metrics enabled:

  * New section added to describe general design approach.

  * Abstract functions "Cost()" and "LoopFree()" defined.

  * MAX_HOPCOUNT typically replaced by MAX_METRIC.

  * DEFAULT_METRIC_TYPE parameter defined, defaulting to HopCount.

  * MetricType MsgTLV defined.

  * Metric8 and Metric16 AddrTLVs defined.

- o Many changes for RFC 5444 compliance

- o New section added for "Notational Conventions" (see Table 1). Many changes to improve readability and accuracy (e.g., eliminate use of "Flooding", "ThisNode", ...).

- o Reorganized and simplified route lifetime management (see Section 5.1).

- o Reorganized document structure, combining closely related small sections and eliminating top-level "Detailed ..." section.

  * RREQ and RREP specification sections coalesced.

  * RERR specification sections coalesced.

  * Eliminated resulting duplicated specification.

  * New section added for "Notational Conventions".

- o Internet-Facing AODVv2 router renamed to be IAR

- o "Optional Features" section (see Section 13) created to contain features not required within base specification, including:

  * Adding RREP-ACK message type instead of relying on reception of arbitrary packets as sufficient response to establish bidirectionality.

  * Expanding Rings Multicast

* Intermediate RREPs (iRREPs): Without iRREP, only the destination can respond to a RREQ.

* Precursor lists.

* Reporting Multiple Unreachable Nodes. An RERR message can carry more than one Unreachable Destination node for cases when a single link breakage causes multiple destinations to become unreachable from an intermediate router.

* Message Aggregation.

* Inclusion of Added Routing Information.

- Sequence number MUST be incremented after generating any RteMsg.

- Resulting simplifications for accepting route updates in RteMsgs.

- Sequence number MUST (instead of SHOULD) be set to 1 after rollover.

- AODVv2 routers MUST (instead of SHOULD) only handle AODVv2 messages from adjacent routers.

- Clarification that Added Routing information in RteMsgs is optional (MAY) to use.

- Clarification that if Added Routing information in RteMsgs is used, then the Route Table Entry SHOULD be updated using normal procedures as described in Section 6.2.

- Clarification in Section 7.1 that nodes may be configured to buffer zero packets.

- Clarification in Section 7.1 that buffered packets MUST be dropped if route discovery fails.

- In Section 8.2, relax mandate for monitoring connectivity to next-hop AODVv2 neighbors (from MUST to SHOULD), in order to allow for minimal implementations

- Remove Route.Forwarding flag; identical to "NOT" Route.Broken.

- Routing Messages MUST be originated with the MsgHdr.<msg-hop-limit> set to MAX_HOPCOUNT.

- Maximum hop count set to MAX_HOPCOUNT, and 255 is reserved for "unknown". Since the current draft only uses hop-count as distance, this is also the current maximum distance.

Appendix C. Shifting Network Prefix Advertisement Between AODVv2 Routers

Only one AODVv2 router within a MANET SHOULD be responsible for a particular address at any time. If two AODVv2 routers dynamically shift the advertisement of a network prefix, correct AODVv2 routing behavior must be observed. The AODVv2 router adding the new network prefix must wait for any existing routing information about this network prefix to be purged from the network. Therefore, it must wait at least ROUTER_SEQNUM_AGE_MAX_TIMEOUT after the previous AODVv2 router for this address stopped advertising routing information on its behalf.

Authors' Addresses

Charles E. Perkins
Futurewei Inc.
2330 Central Expressway
Santa Clara, CA 95050
USA

Phone: +1-408-330-5305
Email: charliep@computer.org

Ian D Chakeres
CenGen
9250 Bendix Road North
Columbia, Maryland 21045
USA

Email: ian.chakeres@gmail.com
URI: http://www.ianchak.com/