Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 30, 2012

T. Clausen
A. Colin de Verdiere
J. Yi
LIX, Ecole Polytechnique
A. Niktash
Maxim Integrated Products
Y. Igarashi
H. Satoh
Hitachi, Ltd., Yokohama Research Laboratory
U. Herberg
Fujitsu Laboratories of America
C. Lavenu
EDF R&D
T. Lys
ERDF

March 29, 2012

The LLN On-demand Ad hoc Distance-vector Routing Protocol - Next Generation (LOADng)
draft-clausen-lln-loadng-03

Abstract

This document describes the LLN Ad hoc On-Demand - Next Generation (LOADng) distance vector routing protocol, a reactive routing protocol intended for use in Low power and Lossy Networks (LLN). The protocol is derived from AODV (RFC3561) and extended for use in LLNs.

This Internet-Draft will expire on September 30, 2012.

Table of Contents

1. Introduction

The LLN On-demand Ad hoc Distance-vector Routing Protocol - Next Generation (LOADng) is a routing protocol, derived from AODV [RFC3561] and extended for use in Low power and Lossy Networks (LLNs). As a reactive protocol, the basic operations of LOADng include generation of Route Requests (RREQs) by a router (originator) for when discovering a route to a destination, forwarding of such RREQs until they reach the destination router, generation of Route Replies (RREPs) upon receipt of an RREQ by the indicated destination, and unicast hop-by-hop forwarding of these RREPs towards the originator. If a route is detected broken, i.e., if forwarding of a data packet to the recorded next hop on the route to the destination is detected to fail, a Route Error (RERR) message is returned to the originator of that data packet.

Compared to [RFC3561], LOADng is simplified as follows:

- Only the destination is permitted to respond to an RREQ; intermediate routers are explicitly prohibited from responding to RREQs, even if they may have active routes to the sought destination, and all messages (RREQ or RREPs) generated by a given

router share a single unique, monotonically increasing sequence number. This also eliminates Gratuitous RREPs while ensuring loop freedom. The rationale for this simplification is reduced complexity of protocol operation and reduced message sizes.

- A LOADng Router does not maintain a precursor list, thus when forwarding of a data packet to the recorded next hop on the route to the destination fails, an RERR is sent only to the originator of that data packet. The rationale for this simplification is an assumption that few overlapping routes are in use concurrently in a given network.

Compared to [RFC3561], LOADng is extended as follows:

- Optimized Flooding is supported, reducing the overhead incurred by RREQ generation and flooding. If no optimized flooding operation is specified for a given deployment, classical flooding is used by default.

- Different address lengths are supported - from full 16 octet IPv6 addresses over 6 octet Ethernet MAC addresses and 4 octet IPv4 addresses to shorter 1 and 2 octet addresses. The only requirement is, that within a given routing domain, all addresses are of the same address length.

- Control messages can include TLV (Type-Length-Value) elements, permitting protocol extensions to be developed.

LOADng supports routing using arbitrary metrics, which can be specified as extensions using the TLV mechanism. In order to provide a "fallback", in case a router on a route does not understand a given metric, LOADng always provides a default "hop-count-with-weak-links" metric - the philosophy being that "any route, even if not with the metric desired, is better than no route".

2. Terminology and Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in

[RFC2119].

Additionally, this document uses the notations in Section 2.1 and the terminology defined in Section 2.2.

2.1. Notations

The following notations, for elements and variables, are used in this document.

This format uses network byte order (most significant octet first) for all fields. The most significant bit in an octet is numbered bit 0, and the least significant bit of an octet is numbered bit 7 [Stevens].

2.1.1. Elements

This specification defines elements. An element is a group of any number of consecutive bits that together form a syntactic entity represented using the notation <element>. Each element in this document is defined as either:

o a specifically sized field of bits OR

o a composite element, composed of other <element>s.

A composite element is defined as follows:

<element> := specification

where, on the right hand side following :=, specification is represented using the regular expression syntax defined in [SingleUNIX]. Only the following notation is used:

<element1><element2> - Indicates that <element1> is immediately followed by <element2>.

(<element1><element2>) - Indicates a grouping of the elements enclosed by the parentheses.

? - Zero or one occurrences of the preceding element or group.

* - Zero or more occurrences of the preceding element or group.

2.1.2. Variables

Variables are introduced into the specification solely as a means to clarify the description. The following two notations are used:

<foo> - If <foo> is an unsigned integer field, then <foo> is also used to represent the value of that field.

bar - A variable, usually obtained through calculations based on the value(s) of element(s).

2.1.3. Conventions

This document uses the following notational conventions:

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

c = d - A comparison operator, returning true if the value of the left side (c) is equal to the value of the right side (d).

2.2. Terminology

This document uses the following terminology:

LOADng Router - A router that implements this routing protocol. A LOADng router can be equipped with one or multiple distinct interfaces.

Interface - A router's attachment to a communications medium. An interface is assigned one or more addresses.

Packet - The top level entity in this specification. A packet contains a Packet Header and zero or one message.

Message - The fundamental entity carrying protocol information, in

the form of address objects and TLVs.

Link Cost - The cost (weight) between a pair of LOADng Routers, determined by a LOADng Router upon receipt of a packet.

Route Cost - The sum of the Link Costs for the links that an RREQ or RREP has crossed.

Weak Link - A link that is marginally usable, i.e., which MAY be used if no other links are available, but which SHOULD be avoided if at all possible - even if it entails ultimately longer routes. As an example, a Weak Link might be defined as a link with a nominatively high bit-rate (thus, a priori attractive) while suffering a significant loss-rate.

3. Applicability Statement

This protocol:

- Is a reactive routing protocol for Low power and Lossy Networks (LLNs).
- Supports the use of optimized flooding for RREQs.
- Enables any router in the LLN to discover bi-directional routes to destinations in the LLN (i.e., any other router, as well as hosts or networks attached to that router).
- Supports addresses of any length, from 16 octets to a single octet.
- Is layer-agnostic, i.e., may be used at layer 3 as a "route over" routing protocol, or at layer 2 as a "mesh under" routing protocol.
- Supports per-destination route maintenance; if a destination becomes unreachable, rediscovery of that single (bi-directional) route is performed, without need for global topology recalculation.

4. Protocol Overview and Functioning

The objective of this protocol is for each LOADng Router to, independently:

- Discover a bi-directional route to any destination in the network.

- Establish routes only when there is data traffic to be sent along that route.

- Maintain a route only for as long as it is an active route, i.e., there is traffic using the route.

- Generate control traffic based on network events only: when a new route is required, or when an active route is detected broken. Specifically, this protocol does not require periodic signaling.

4.1. Overview

These objectives are achieved, for each LOADng Router, by performing the following tasks:

- When having a data packet to deliver to a destination, for which no tuple in the routing table exists, generate a Route Request (RREQ) encoding the destination address, and transmit this to all of its neighbors.

- Upon receiving an RREQ, install or refresh a tuple in the routing table towards the originator address from the RREQ, as well as to the neighbor LOADng Router from which the RREQ was received. This will install the Reverse Route (towards the originator address from the RREQ).

- Upon receiving an RREQ, inspect the indicated destination address:

  * If that address is an address in the Destination Address Set of the LOADng Router, generate a Route Reply (RREP), which is unicast in a hop-by-hop fashion along the installed Reverse Route.

  * If that address is not an address in the Destination Address Set of the LOADng Router, consider the RREQ as a candidate for forwarding.

- When an RREQ is considered a candidate for forwarding, retransmit it according to the flooding operation, specified for the network.

- Upon receiving an RREP, install a route towards the originator address from the RREP, as well as to the neighbor LOADng Router, from which that RREP was received. This will install the Forward Route (towards the originator address from the RREP). The originator address is either an address from the Local Interface Set of the LOADng Router, or an address from its Destination

Address Set (i.e. an address of a host attached to that router).

- Upon receiving an RREP, forward it, as unicast, to the recorded next hop along the corresponding Reverse Route until the RREP gets to the LOADng Router that has the destination address from the RREP in its Local Interface Set or Destination Address Set.

A router generating an RREQ specifies which metric it desires. Routers receiving an RREQ will process it and update route cost information in the RREQ according to that metric, if they can. All routers, however, will update information in the RREQ so as to be able to support the "hop-count-with-weak-links" default metric. If a router is not able to understand the specified metric in an RREQ, it will change the metric type in the RREQ to "hop-count-with-weak-links" so as to ensure that it be indicated what metric is supported by the path taken by that copy of the RREQ.

4.2. Routers and Interfaces

In order for a LOADng Router to participate in a LLN, it MUST have at least one, and possibly more, LOADng interfaces. Each LOADng interface:

- Is configured with one or more interface addresses.

In addition to a set of LOADng interfaces as described above, each LOADng Router:

- Has a number of router parameters.

- Has an Information Base.

- Generates and processes RREQ, RREP, RREP_ACK and RERR messages, according to this specification.

4.3. Information Base Overview

Necessary protocol state is recorded by way of five information sets: the "Routing Set", the "Local Interface Set", the "Blacklisted Neighbor Set", the "Destination Address Set", and the "Pending Acknowledgment Set".

The Routing Set contains tuples, each representing the next-hop on, and the cost of, a route towards a destination address. Additionally, the Routing Set records the sequence number of the last message, received from the destination. This information is extracted from the message (RREQ or RREP) that generated the tuple so as to enable routing. The routing table is to be updated using this Routing Set. (A router MAY choose to use any or all destination addresses in the Routing Set to update the routing table, this

selection is outside the scope of this specification.)

The Local Interface Set contains tuples, each representing a local interface of the router. Each tuple contains a list of one or more addresses of that interface.

The Blacklisted Neighbor Set contains tuples representing neighbor LOADng Routers with which unidirectional connectivity has been recently detected.

The Destination Address Set contains tuples representing addresses, for which the LOADng Router is responsible; i.e., be addresses of this LOADng Router, or of hosts and networks directly attached to this router and which use it to connect to the LLN. These addresses may in particular belong to devices which do not implement LOADng, and thus cannot process LOADng messages. This router SHOULD provide connectivity to these addresses by generating RREPs in response to RREQs directed towards them.

The Pending Acknowledgment Set contains tuples, representing transmitted RREPs for which an RREP_ACK is expected, but where this RREP_ACK has not yet been received.

The Routing Set, the Blacklisted Neighbor Set and the Pending Acknowledgment Set are updated by this protocol. The Destination Address Set is used, but not updated, by this protocol.

4.4. Signaling Overview

This protocol generates and processes the following routing messages:

Route Request (RREQ) - Generated by a LOADng Router when it has a

data packet to deliver to a given destination, but when it does not have an available tuple in its Routing Set indicating a route to that destination. An RREQ contains:

* The address (destination) to which a Forward Route is to be discovered by way of soliciting the LOADng Router with that destination address in its Local Interface Set or in its Destination Address Set to generate an RREP.

* The address for which a Reverse Route is to be installed (originator) by RREQ forwarding and processing, i.e., the source address of the data packet which triggered the RREQ generation.

* The sequence number of the LOADng Router, generating the RREQ.

An RREQ is flooded through the network, according to the flooding operation specified for the network.

Route Reply (RREP) - Generated as a response to an RREQ by the LOADng Router which has the address (destination) from the RREQ in its Local Interface Set or in its Destination Address Set. An RREP is sent in unicast towards the originator of that RREQ. An RREP contains:

* The address (originator) to which a Forward Route is to be installed when forwarding the RREP.

* The address (destination) towards which the RREP is to be sent. More precisely, the destination address indicates the unicast route which the RREP follows.

* The sequence number of the LOADng Router, generating the RREP.

Route Reply Acknowledgement (RREP_ACK) - Generated by a LOADng Router as a response to an RREP, in order to signal to the neighbor that transmitted the RREP that the RREP was successfully received. Receipt of an RREP_ACK indicates that the link between these two neighboring LOADng Routers is bidirectional. An RREP_ACK is unicast to the neighbor from which the RREP has arrived, and is not forwarded. RREP_ACKs are generated only in

response to an RREP which, by way of a flag, has explicitly indicated that an RREP_ACK is desired.

Route Error (RERR) - Generated by a LOADng Router when a link on an active route to a destination is detected as broken by way of inability to forward a data packet towards that destination. An RERR is unicast to the source of the undeliverable data packet.

5. Protocol Parameters and Constants

The following router parameters and constants are used in this specification.

LL-LLN-Routers - is a link-local-scoped multicast address of a group, which all LOADng Routers MUST join if LOADng is used as route-over protocol using IP.

NET_TRAVERSAL_TIME - is the maximum time that a packet is expected to take when transversing from one end of the network to the other.

RREQ_RETRIES - is the maximum number of subsequent RREQs that a particular router may generate in order to discover a route to a destination, before declaring that destination unreachable.

RREQ_RATELIMIT - is the maximum number of RREQs that a particular router is allowed to send per time interval.

R_HOLD_TIME - is the minimum time a Routing Tuple SHOULD be kept in the Routing Set after it was last refreshed. This MAY be a network-wide constant, but MAY also be a variable whose value is defined by an auxiliary mechanism, e.g., by an extension to this protocol.

MAX_DIST - is the value (tuple) representing the maximum possible distance (R_dist field).

RREP_ACK_REQUIRED - is a boolean flag, which indicates (if set) that the router is configured to expect that each RREP it sends be

confirmed by an RREP_ACK or (if cleared) that no RREP_ACK is expected.

RREP_ACK_TIMEOUT - is the minimum time after transmission of an RREP, that a LOADng Router SHOULD wait for an RREP_ACK from a neighbor LOADng Router, before considering that the link to this neighbor as unidirectional.

B_HOLD_TIME - is the time during which the link between the neighbor LOADng Router and this LOADng Router MUST be considered as non-bidirectional, and that therefore RREQs received from that neighbor LOADng Router MUST be ignored after being added. B_HOLD_TIME should be greater than 2 x NET_TRANSVERSAL_TIME x RREQ_RETRIES, to ensure that subsequent RREQs will reach the destination via a route, excluding this link.

USE_BIDIRECTIONAL_LINK_ONLY - is a boolean flag, which indicates if the LOADng Router only uses verified bi-directional links for data packet forwarding. It is set by default. If cleared, then the LOADng Router can use links which have not been verified to be bi-directional.

HOP_COUNT_WITH_WEAK_LINKS - is the value representing the default hop count with weak links metric, see Section 16.

6. Information Base

Each LOADng Router maintains an Information Base, containing the information sets necessary for protocol operation, as described in the following sections. The organization of information into these

information sets is non-normative, given so as to facilitate description of message generation, forwarding and processing rules in this specification. An implementation may choose any representation or structure for when maintaining this information.

6.1. Routing Set

The Routing Set records the next hop on the route to each known destination, when such a route is known. It consists of Routing Tuples:

(R_dest_addr, R_next_addr, R_dist, R_metric,
R_seq_num, R_valid_time, R_bidirectional, R_local_iface_addr)

where:

R_dest_addr - is the address of the destination, either the address of an interface of a destination LOADng Router, or the address of an interface reachable via the destination LOADng Router, but which is outside the LLN.

R_next_addr - is the address of the "next hop" on the selected route to the destination.

R_dist - is the distance associated with the selected route to the destination with address R_dest_addr. R_dist is a tuple containing Route Cost, Weak Links and (depending on the metric used) additional fields; see Section 16.

R_metric - specifies how R_dist is defined and calculated, as well as the comparison operator '<=' for determining which of two route costs is lower. This is specified in Section 16.

R_seq_num - is the value of the <seq-num> field of the RREQ or RREP which installed or last updated this tuple. For the routing tuples installed by previous hop information of RREQ or RREP, R_seq_num MUST be set to -1.

R_valid_time - specifies the time until which the information recorded in this tuple is considered valid.

R_bidirectional - is a boolean flag, which specifies if the routing tuple is verified as representing a bi-directional route. Data traffic SHOULD only be routed through a routing tuple with R_bidirectional flag equals TRUE, unless the router is configured as accepting routes without bi-directionality verification explicitly by setting the USE_BIDIRECTIONAL_LINK_ONLY to FALSE.

R_local_iface_addr - is the address of the local interface, through which the destination can be reached.

6.2. Local Interface Set

A router's Local Interface Set records its local interfaces. It consists of Local Interface Tuples, one per interface:

(I_local_iface_addr_list)

where:

I_local_iface_addr_list - is an unordered list of the network addresses of this interface.

The implementation MUST initialize the Local Interface Set with at least one tuple containing at least one address of an interface. Moreover, the implementation MUST update the Local Interface Set if there is a change of the interfaces of a LOADng router (i.e. a new interface, a removed interface, or a change of addresses of an interface).

6.3. Blacklisted Neighbor Set

The Blacklisted Neighbor Set records the neighbor interface addresses of a LOADng Router, with which connectivity has been detected to be unidirectional. Specifically, the Blacklisted Neighbor Set records neighbors from which an RREQ has been received (i.e., through which a Forward Route would possible) but to which it has been determined that it is not possible to communicate (i.e., forwarding Route Replies via this neighbor fails, rendering installing the Forward Route impossible). It consists of Blacklisted Neighbor Tuples:

(B_neighbor_address, B_valid_time)

where:

B_neighbor_address - is the address of the blacklisted neighbor interface.

B_valid_time - specifies the time until which the information recorded in this tuple is considered valid.

6.4. Destination Address Set

The Destination Address Set records addresses, for which a LOADng Router will generate RREPs in response to received RREQs, in addition to its own interface addresses (as listed in the Local Interface

Set). The Destination Address Set thus represents those destinations (i.e. hosts), for which this LOADng Router is providing connectivity. It consists of destination address tuples:

(D_address)

where:

D_address - is the address of a destination (a host or a network), attached to this LOADng Router and for which this LOADng Router provides connectivity through the LLN.

The Destination Address Set is used for generating signaling, but is not itself updated by signaling specified in this document. Updates to the Destination Address Set are due to changes of the environment of a LOADng Router - hosts or external networks being connected to or disconnected from a LOADng Router. The Destination Address Set may be administrationally provisioned, or provisioned by external protocols.

6.5. Pending Acknowledgment Set

The Pending Acknowledgment Set contains information about RREPs which have been transmitted with the ACK_REQUIRED flag set, and for which an RREP_ACK has not yet been received. It consists of Pending Acknowledgment Tuples:

(P_next_hop, P_originator, P_seq_num, P_ack_timeout)

where:

P_next_hop - is the address of the neighbor interface to which the RREP was sent.

P_originator - is the address of the originator of the RREP.

P_seq_num - corresponds to the <seq-num> field of the sent RREP.

P_ack_timeout - is the time after which the neighbor is considered not to have a bidirectional link to this router and MUST be added to the Blacklisted Neighbor Set; the tuple MUST then be discarded.

7. LOADng Router Sequence Numbers

Each LOADng Router maintains a single sequence number, which must be included in each RREQ or RREP message it generates. Each router MUST make sure that no two messages (both RREQ and RREP) are generated

with the same sequence number, and MUST generate sequence numbers

such that these are monotonically increasing. This sequence number is used as freshness information for when comparing routes to the router having generated the message.

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

S2 < S1 AND S1 - S2 <= MAXVALUE/2 OR

S1 < S2 AND S2 - S1 > MAXVALUE/2

8. Packet Format

The packet format, used by this protocol, is described in this section using the notational conventions described in Section 2. Example packets are illustrated in Appendix A.

The general format for all packets, generated, forwarded and processed by this specification, is as follows:

```
<packet> := <type>
            <addr-length>
            <tlv-block>
            <message>
```

where:

<type> is a 4 bit unsigned integer field and specifies the type of the <message> field, specified in Section 8.2.

<addr-length> is a 4 bit unsigned integer field, encoding the length of the destination and originator addresses of the <message> field (<destination> and <originator>) as follows:

<addr-length> := the length of an address in octets - 1

<addr-length> is thus 1 for 16 bit short addresses [RFC4944], 3 for IPv4 addresses, 7 for 64 bit extended addresses [RFC4944] or 15 for IPv6 addresses.

<tlv-block> is specified in Section 8.1.

<message> is specified in Section 8.2.

8.1. TLV Block

The TLV Block contains zero or more Type-Length-Value elements (TLVs). A TLV allows the association of an arbitrary attribute with a packet. The attribute (value) is made up from an integer number of consecutive octets. Different attributes have different types; attributes which are unknown when parsing can be skipped, as specified by flags associated with a given TLV.

<tlv-block> := <tlv-count>
(<tlv-type><tlv-flags><tlv-length><tlv-value>)*

where:

<tlv-count> is a 4 bit unsigned integer field, specifying the number of TLVs included.

<tlv-type> is a 4 bit unsigned integer field, specifying the type of the TLV.

<tlv-flags> is a 4 bit field specifying processing and forwarding rules related to the TLV processing:

bit 0 (pifunknown): If cleared (0), indicates that if a LOADng Router does not understand the <tlv-type>, then it MUST NOT process or forward the packet and the packet MUST be silently dropped. If set (1), indicates that if a LOADng Router does not understand the <tlv-type>, then it MAY process the packet,

and all TLVs with <tlv-type> fields which it understands.

bit 1 (kifunknown): If cleared (0), indicates that if a LOADng Router does not understand the <tlv-type>, then it MUST remove the TLV from the packet prior to processing and (for packets, intended to be forwarded) forwarding. If set (1), indicates that the LOADng Router MAY keep the TLV when processing (which is then determined by the value of the pifunknown flag) and (for packets, intended to be forwarded) forwarding.

bit 2-3 (RESERVED): SHOULD be set to zero on transmission and SHOULD be ignored upon receipt.

<tlv-length> is an 8 bit unsigned integer field, specifying the length of the following <tlv-value> field in octets.

<tlv-value> is a field of length <length> octets.

8.2. Message Format

This section specifies the format of the <message> field for message types RREQ, RREP, RREP_ACK and RERR.

8.2.1. RREQ and RREP Message Format

The format of Route Request (RREQ) and Route Reply (RREP) messages is identical, RREQ and RREP messages being distinguished by the <type> field in the packet. They are as follows:

```
<message> := <flags>
             <seq-num>
             <metric>
             <weak-links>
             <hop-count>
             <destination>
             <originator>
```

where:

<flags> is a 4 bit unsigned integer field and specifies the interpretation of the remainder of the message.

For RREQ messages:

bit 0-3 (RESERVED): SHOULD be set to zero on transmission and SHOULD be ignored upon receipt.

For RREP messages:

bit 0 (ackrequired): When set ('1'), an RREP_ACK MUST be generated by the recipient of an RREP if the RREP is successfully processed. When cleared ('0'), an RREP_ACK MUST NOT be generated in response to processing of the RREP.

bit 1-3 (RESERVED): SHOULD be set to zero on transmission and SHOULD be ignored upon receipt.

<seq-num> is a 16 bit unsigned integer field, containing the sequence number (see Section 7) of the LOADng Router, generating the RREQ or RREP message.

<metric> is a 4 bit unsigned integer field and specifies how the route cost is to be calculated, as well as the comparison operator '<=' used for when determining which among two route costs is lower. The route cost calculation MAY be based on the <weak-links> and <hop-count> fields of the packet. It MAY also use additional information, encoded in TLVs.

<weak-links> is a 4 bit unsigned integer field and specifies the total number of weak links on the route from the originator to the destination. This field MAY be updated when a packet is forwarded, see Section 11.2.

<hop-count> is an 8 bit unsigned integer field and specifies the total number of hops which the packet has transversed from the <originator> to the <destination>. This field MUST be updated,

when a packet is forwarded, see Section 12.3 and Section 13.3.

<destination> is an identifier of <address-length> + 1 octets, specifying the address to which the RREQ or RREP should be sent. (I.e., for an RREQ, this address would be the interface address for which a route is sought. For an RREP, this address is equivalent to the <originator> address of the RREQ that triggered the RREP.)

<originator> is an identifier of <address-length> + 1 octets, specifying the interface address for which this message was generated, and to which a route is supplied by this message. For an RREQ, the route supplied corresponds to the "reverse route", whereas for an RREP the route supplied corresponds to the "forward route". In case the message is generated on a LOADng router on behalf of an attached host, the <originator> address corresponds to an interface address of that host, otherwise it corresponds to an address of the sending interface of the LOADng router.

8.2.2. RREP_ACK Message Format

The format of a Route Reply Acknowledgement (RREP_ACK) message is as follows:

```
<message> := <flags>
             <seq-num>
             <originator>
```


where:

<flags> is a 4 bit unsigned integer field and specifies the interpretation of the remainder of the message:

bit 0-3 (RESERVED): SHOULD be set to zero on transmission and SHOULD be ignored upon receipt.

<seq-num> is a 16 bit unsigned integer field and contains the value of the <seq-num> field from the RREP for which this RREP_ACK is sent.

<originator> is an identifier of <address-length> + 1 octets and contains the value of the <originator> field from the RREP for which this RREP_ACK is sent.

8.2.3. RERR Message Format

The format of a Route Error (RERR) message is as follows:

```
<message> := <error-code>
             <originator>
             <destination>
```

where:

<error-code> is a 4 bit unsigned integer field and specifies the reason for the error message being generated, according to Table 4.

<originator> is an identifier of <address-length> + 1 octets, specifying the source address of a data packet, for which delivery to <destination> failed. The unicast destination of the RERR message is the LOADng Router which has <destination> listed in a Local Interface Tuple or in a Destination Address Tuple.

<destination> is an identifier of <address-length> + 1 octets, specifying the address of the destination, which has become unreachable, and for which an error is reported.

9. Route Maintenance

Tuples in the Routing Set are maintained by way of five different mechanisms:

- RREQ/RREP exchange, specified in Section 12 and Section 13.

- Data traffic delivery success.
- Data traffic delivery failure.
- External signals indicating that a tuple in the Routing Set

necessitates updating.

- Information expiration.

Routing Tuples in the Routing Set contain a validity time, which specifies the time until which the information recorded in this tuple is considered valid. After this time, the information in such tuples is to be considered as invalid, for the processing specified in this document.

Routing Tuples for actively used routes (i.e., a route via which traffic is currently transiting) SHOULD NOT be removed, unless there is evidence that they no longer provide connectivity - i.e., unless a link on that route has broken.

To this end, one or more of the following mechanisms (non-exhaustive list) MAY be used:

- If a lower layer mechanism provides signals, such as when delivery to a presumed neighbor LOADng Router fails, this signal MAY be used to indicate that a link has broken, trigger early expiration of a Routing Tuple from the Routing Set, and to initiate Route Error Signaling (see Section 14). Conversely, absence of such a signal when attempting delivery MAY be interpreted as validation that the corresponding Routing Tuple(s) are valid, and their R_valid_time refreshed correspondingly. Note that when using such a mechanism, care should be taken to prevent that an intermittent error (e.g., an incidental wireless collision) triggers corrective action and signaling. This depends on the nature of the signals, provided by the lower layer, but can include the use of a hysteresis function or other statistical mechanisms.

- Conversely, for each successful delivery of a packet to a neighbor or a destination, if signaled by a lower layer or a transport mechanism, or each positive confirmation of the presence of a neighbor by way of an external neighbor discovery protocol, MAY be interpreted as validation that the corresponding Routing Tuple(s) are valid, and their R_valid_time refreshed correspondingly.

Furthermore, a LOADng Router may experience that a route currently used for forwarding data packets is no longer operational, and must act to either rectify this situation locally (Section 13) or signal this situation to the source of the data packets for which delivery

was unsuccessful (Section 14).

10. Unidirectional Link Handling

Each LOADng Router MUST monitor the bidirectionality of the links to its neighbors and set the R_bidirectional flag of related routing tuples when processing Route Replies (RREP). To this end, one or more of the following mechanisms MAY be used (non exhaustive list):

- If a lower layer mechanism provides signals, such as when delivery to a presumed neighbor LOADng Router fails, this signal MAY be used to detect that a link to this neighbor is broken or is unidirectional; the LOADng Router MUST then blacklist the neighbor, see Section 10.1.

- If a mechanism such as NDP [RFC4861] is available, the LOADng Router MAY use it.

- RREP_ACK message exchange, as described in Section 15.

- Upper-layer mechanisms, such as transport-layer acknowledgements, MAY be used to detect unidirectional or broken links.

When a LOADng Router detects, via one of these mechanisms, that a link to a LOADng neighbor router is unidirectional or broken, the router MUST blacklist this neighbor, see Section 10.1. Conversely, if a LOADng Router detects via one of these mechanisms that a previously blacklisted LOADng Router has a bidirectional link to this router, it MAY remove it from the blacklist before the <B_valid_time> of the corresponding tuple.

10.1. Blacklist Usage

The Blacklist is maintained according to Section 6.3. When a LOADng Router is detected to have a unidirectional link to the LOADng Router, it is blacklisted, i.e., a tuple (B_neighbor_address, B_valid_time) is created thus:

- B_neighbor_address := the address of the blacklisted neighbor

- B_valid_time := current_time + B_HOLD_TIME

When a LOADng neighbor router is blacklisted, i.e., when there is a corresponding (B_neighbor_address, B_valid_time) tuple in the Blacklisted Neighbor Set, it is temporarily not considered as a neighbor, and thus:

- o Every RREQ received from this neighbor MUST be discarded;

11. Common Rules for RREQ and RREP Messages

RREQ and RREP messages, both, supply routes between their recipients and the originator of the RREQ or RREP message. The two message types therefore share common processing rules, and differ only in the following:

- o RREQ messages are multicast or broadcast, intended to be received by all LOADng Routers in the network, whereas RREP messages are all unicast, intended to be received only by routers on a specific route towards a specific destination.

- o Receipt of an RREQ message MAY trigger generation of an RREP message.

- o Receipt of an RREP message MAY trigger generation of an RREP_ACK message.

For the purpose of the processing description in this section, the following additional notation is used:

<= is the comparison operator specified by the <metrics> field in the RREQ or RREP message and described in Section 16.

received-route-cost is a variable, representing the cost of the route, as calculated based on the received message, see Section 16.

used-metric is a variable, representing the metric used for calculating received-route-cost, see Section 16.

previous-hop is the address of the LOADng Router, from which the RREQ or RREP message was received.

> is the comparison operator for <seq-num> specified in Section 8.

11.1. Identifying Invalid RREQ or RREP Messages

A received RREQ or RREP message is invalid, and MUST be discarded without further processing, if any of the following conditions are

true:

- The address length specified by this message (i.e., <addr-length> + 1) differs from the length of the address(es) of this router.

- The address contained in the <originator> field is an address of this router.
- The <hop-count> field is greater than MAX_HOP_COUNT (i.e., 255).
- The <weak-links> field is greater than MAX_WEAK_LINKS (i.e., 15).
- There is a tuple in the Routing Set where:
  - R_dest_addr = <originator>
  - R_seq_num > <seq-num>
- For RREQ messages only, an RREQ MUST be considered invalid if the previous-hop is blacklisted (i.e. its address is in a tuple in the Blacklisted Neighbor Set, see Section 10.1).

A LOADng Router MAY recognize additional reasons for identifying that an RREQ or RREP message is invalid for processing, e.g., to allow a security protocol to perform verification of signatures and prevent processing of unverifiable RREQ or RREP message by this protocol.

11.2. RREQ and RREP Message Processing

A received, and valid, RREQ or RREP message is processed as follows:

1. Included TLVs are processed/removed/updated according to their specification.
2. If the RREQ or RREP message was received over a "weak link", increment the <weak-links> field in the received RREQ or RREP by one.
3. If the <metric>, indicated in the message, is known to this LOADng Router, then:

* Set the variable used-metric to the value of <metric>.

4. Otherwise, if the <metric>, indicated in the message, is unknown to this LOADng Router:

   * Set the variable used-metric to HOP_COUNT_WITH_WEAK_LINKS.

5. Set the variable received-route-cost to the route cost, calculated according to used-metric.

6. Find the Routing Tuple (henceforth, matching Routing Tuple) where:

   * R_dest_addr = <originator>

   * R_metric = used-metric

7. If no matching Routing Tuple is found, then create a new matching Routing Tuple (the "reverse route" for RREQ messages or "forward route" for RREP messages) with:

   * R_dest_addr := <originator>

   * R_next_addr := previous-hop

   * R_metric := used-metric

   * R_dist := MAX_DIST

   * R_seq_num := -1

   * R_valid_time := current time + R_HOLD_TIME

   * R_bidirectional := FALSE

   * R_local_iface_addr := the interface address through which the packet was received.

8. The matching Routing Tuple, existing or new, is compared to the received RREQ or RREP message:

1. If

   + received-route-cost < R_dist; AND

   + R_seq_num = <seq-num>

   OR

   + <seq-num> > R_seq_num

   Then:

   + The message is used for updating the Routing Set according to Section 11.3.

   + If there is no matching Routing Tuple in the Routing Set with R_dest_addr = previous-hop, create a new matching Routing Tuple with:

   - R_dest_addr := previous-hop

   - R_next_addr := previous-hop

   - R_metric := HOP_COUNT_WITH_WEAK_LINKS

   - R_dist := (HC, WL), where HC = 1 and WL = 1 if the message was received over a "weak link". Otherwise, WL = 0

   - R_seq_num := -1

   - R_valid_time := current time + R_HOLD_TIME

   - R_bidirectional := TRUE, if the processed message is an RREP, otherwise FALSE.

   - R_local_iface_addr := the interface address through which the packet was received.

2. Otherwise, the RREQ or RREP message is not processed further,

and is not considered for forwarding.

11.3. Updating Routing Tuples In Response to RREQ and RREP

A Routing Tuple in the Routing Set is updated when a received RREQ or RREP message provides a better route to the <originator> than the route current recorded for a given metric. The Routing Tuple, where:

- R_dest_addr = <originator>; AND
- R_metric = used-metric

is updated thus:

- R_next_addr := previous-hop
- R_dist := received-route-cost
- R_seq_num := <seq-num>
- R_valid_time := current time + R_HOLD_TIME
- R_bidirectional := TRUE, if the message being processed is an RREP.

12. Route Requests (RREQs)

Route Requests (RREQs) are generated by a LOADng Router when it has data packets to deliver to a destination for which it has no matching bi-directional tuple in the Routing Set (i.e., with R_bidirectional set to TRUE). Only when the router is configured explicitly as being able to use routing tuples without bi-directionality verification (i.e., with R_bidirectional set to FALSE) by setting USE_BIDIRECTIONAL_LINK_ONLY flag to FALSE, can the router use the routing tuple without initiating an RREQ. The RREQ is transmitted to all directly reachable neighbor LOADng Routers.

After originating an RREQ, a LOADng Router waits for a corresponding RREP. If no such RREP is received within 2*NET_TRAVERSAL_TIME

milliseconds, the LOADng Router MAY issue a new RREQ for the sought destination (with an incremented seq_num) up to a maximum of RREQ_RETRIES times. A LOADng Router SHOULD NOT originate more than RREQ_RATELIMIT RREQs per second. A LOADng Router MAY use mechanisms such as exponential backoff to determine the rate at which it originates RREQs.

12.1. RREQ Generation

A packet with an RREQ message is generated according to Section 8.2 with the following content:

- <type> := RREQ;
- <addr-length> set to the length of the address, as specified in Section 8;
- <metric> set to indicate how route costs are to be calculated and compared, according to Table 3;
- <weak-links> := 0;
- <seq-num> set to the next unused sequence number, maintained by this router;
- <hop-count> := 1;
- <destination> := the address to which a route is sought;
- <originator> := one address of the LOADng Router interface that generates the RREQ. If the LOADng Router is generating RREQ on behalf of a host connected to this LOADng Router, the sender address of the host is used;

- TLVs, as neccessary for the <metric> (if any), see Section 16.

12.2. RREQ Processing

On receiving an RREQ message, a LOADng Router MUST process the message according to this section:

1. If the message is invalid for processing, as defined in Section 11.1, the message MUST be discarded without further processing. The message is not considered for forwarding.

2. Otherwise, the message is processed according to Section 11.2.

3. If <destination> in the RREQ message is not listed in I_local_iface_addr_list of any Local Interface Tuple, or does correspond to D_address of any Destination Address Tuple of this LOADng Router, then the message is considered for forwarding according to Section 12.3.

4. Otherwise, an RREP can be generated, see Section 13.1. The RREQ is not considered for forwarding.

12.3. RREQ Forwarding

An RREQ, considered for forwarding, MUST be updated as follows, prior to it being transmitted:

1. <metric> := used-metric (as set in Section 11.2)

2. <hop-count> := <hop-count> + 1

3. TLVs used by <metric> updated according to the specification of <metric> included in the RREQ, see Section 16.

An RREQ is forwarded according to the flooding operation, specified for the network. This MAY be by way of classic flooding, or the flooding operation for a given network MAY employ a reduced relay set mechanism such as [SMF] or any other information diffusion mechanism such as [RFC6206]. Care must be taken that NET_TRAVERSAL_TIME is chosen so as to accommodate for the maximum time that may take for an RREQ to transverse the network, accounting for in-router delays incurring due to or imposed by such algorithms.

12.4. RREQ Transmission

RREQs, initially generated or forwarded, are sent to all neighbor LOADng Routers. If LOADng is operating as an IP routing protocol, the destination address for this RREQ MUST be the link local

multicast address LL-LLN-Routers, and the source address MUST be the address of the interface over which the RREQ is sent.

When an RREQ is transmitted, all receiving LOADng Routers will process the RREQ message and MAY consider the RREQ message for forwarding at the same, or at almost the same, time. If using data link and physical layers that are subject to packet loss due to collisions, such RREQ messages SHOULD be jittered as described in [RFC5148].

13. Route Replies (RREPs)

Route Replies (RREPs) are generated by a LOADng Router in response to an RREQ, and is sent by the LOADng Router which has, in either its Destination Address Set or in its Local Interface Set, the address which is contained in the <destination> element of the received RREQ. RREPs are sent, hop by hop, in unicast towards the originator of the corresponding RREQ, along the Reverse Route installed by that RREQ. A router, upon forwarding an RREP, installs the Forward Route towards the <destination>.

Thus, with forwarding of RREQs installing the Reverse Route and forwarding of RREPs installing the Forward Route, bi-directional routes are provided between the <originator> and <destination> indicated in the RREQ.

13.1. RREP Generation

At least one RREP MUST be generated in response to a (set of) received RREQ messages with identical (<originator>,<seq-num>). An RREP can be generated immediately as a response to each RREQ processed, or can be generated after a certain delay after the arrival of the first RREQ, in order to use the "best" received RREQ (received over lowest-cost route, over the route with least Weak Links etc). A LOADng Router MAY generate further RREPs for subsequent RREQs received with the same (<originator>,<seq-num>) pairs, if these indicate a better route. The content of an RREP is as follows:

- <type> := RREP;
- <flag> bit-0 ackrequired flag set to ('1') if RREP_ACK is required by the router (i.e. if RREP_ACK_REQUIRED is set to TRUE). Otherwise, bit-0 is cleared ('0');
- <addr-length> set to the length of the address, as specified in Section 8;

- <seq-num> set to the next unused sequence number, maintained by this LOADng Router;
- <metric> set to the same value as the <metric> in the corresponding RREQ;
- <weak-links> := 0;
- <hop-count> := 1;
- <destination> := the address to which this RREP message is to be sent; this corresponds to the <originator> address from the RREQ message, in response to which this RRREP message is generated;
- <originator> := the address of the LOADng Router, generating the RREP. If the LOADng Router is generating RREP on behalf of the hosts connected to it, or on behalf of one of the addresses contained in the routers Destination Address Set, the host address is used.
- TLVs, as neccessary for the <metric> (if any), see Section 16.

The specification of the TLVs included in the <tlv-block> of the RREQ responsible to generate the RREP MUST stipulate if, and under which conditions, these are to be included in the <tlv-block> of the RREP.

13.2. RREP Processing

On receiving an RREP message, a LOADng Router MUST process the message according to this section:

1. If the message is invalid for processing, as defined in Section 11.1, the message MUST be discarded without further processing. The message is not considered for forwarding.
2. Otherwise, the message is processed according to Section 11.2.
3. If the RREP message has the ackrequired flag set, an RREP_ACK message MUST be sent to the previous-hop, according to Section 15.1.
4. If the <destination> in the RREP message is not listed in I_local_iface_addr_list of any Local Interface Tuple and does not

correspond to D_address of any Destination Address Tuple of this LOADng Router, the RREP message is considered for forwarding according to Section 13.3.

13.3. RREP Forwarding

An RREP message, considered for forwarding, MUST be updated as follows, prior to it being transmitted:

1. <metric> := used-metric (as set in Section 11.2)

2. <hop-count> := <hop-count> + 1

3. TLVs used by <metric> updated according to the specification of <metric> included in the RREQ, see Section 16.

4. If this LOADng Router is configured to use RREP_ACKs in order to check the bidirectionality of the links (i.e. RREP_ACK_REQUIRED is set to TRUE), the ackrequired flag MUST be set to (1), according to Section 15.

The RREP message is then unicast to the next hop towards the <destination> indicated in the RREP.

13.4. RREP Transmission

An RREP is, ultimately, destined for the LOADng Router listed in the <destination> field, and is forwarded in unicast towards this LOADng Router. The RREP MUST, however, be transmitted so as to allow it to be processed in each intermediate LOADng Router to:

- Install proper forward routes;

- Permit that <hop-count> and <weak-links> be updated to reflect the route; AND

- Permit that TLVs included may be processed/added/removed according to their specification.

14. Route Errors (RERRs)

If a LOADng Router fails to deliver a data packet to a next hop or a destination, it MUST generate a Route Error (RERR), and send this RERR along the Reverse Route towards the source of the data packet for which delivery was unsuccessful (to the last router along the Reverse Route, if the data packet was originated by a host behind that router).

14.1. RERR Generation

A packet with an RERR message is generated by the LOADng Router, detecting the link breakage, with the following content:

- <type> := RERR;
- <error-code> := the most appropriate error code from among those recorded in Table 4;
- <addr-length> := the length of the address, as specified in Section 8;
- <originator> := the source address from the unsuccessfully delivered data packet.
- <destination> := the destination address from the unsuccessfully delivered data packet.

14.2. RERR Processing

For the purpose of the processing description below, the following additional notation is used:

previous-hop is the address of the LOADng Router, from which the RERR was received.

Upon receiving an RERR, a LOADng Router MUST perform the following steps:

1. Included TLVs are processed/removed/updated according to their specification.
2. Find the Routing Tuple (henceforth "matching Routing Tuple") in

the Routing Set where:

* R_dest_addr = <destination>

* R_next_addr = previous-hop

3. If no matching Routing Tuple is found, the RERR is not processed further, and is not considered for forwarding.

4. Otherwise, if one matching Routing Tuple is found, this matching Routing Tuple is updated as follows:

   * R_valid_time := expired

   The RERR message is, then, considered for forwarding.

14.3. RERR Forwarding

An RERR is, ultimately, destined for the LOADng Router on which the address from the <originator> field is listed in I_local_iface_addr_list of any Local Interface Tuple or which corresponds to D_address of any Destination Address Tuple.

An RERR, considered for forwarding is therefore processed as follows:

1. Find the Destination Address Tuple (henceforth, matching Destination Address Tuple) in the Destination Address Set where:

   * D_address = the address from the <originator> field of the RERR.

2. If one or more matching Destination Address Tuples are found, the RERR message is discarded and not retransmitted, as it has reached the final destination.

3. Otherwise, find the Local Interface Tuple (henceforth, matching Local Interface Tuple) in the Local Interface Set where:

* I_local_iface_addr_list contains the address from the <originator> field of the RERR.

4. If a matching Local Interface Tuple is found, the RERR message is discarded and not retransmitted, as it has reached the final destination.

5. Otherwise, if no matching Destination Address Tuples or Local Interface Tuples are found, the RERR message is transmitted according to Section 14.4.

14.4. RERR Transmission

An RERR is transmitted, as unicast, to the LOADng Router, recorded the next hop for the <originator> indicated in the RERR message. The RERR MUST be transmitted hop-by-hop such that it can be processed in each intermediate LOADng Router. This serves to:

- Allow intermediate routers to update their Routing Sets, i.e., remove tuples for this destination.

- Permit that TLVs included may be processed/added/removed according to their specification.

15. Route Reply Acknowledgements (RREP_ACKs)

A LOADng Router SHOULD use RREP_ACK exchange to monitor bidirectionality of links with neighbor routers, except if another mechanism, as described in Section 10, provides for such bidirectionality information.

A LOADng Router MUST signal in a transmitted RREP that it is expecting an RREP_ACK, by setting the ackrequired flag in the RREP. When doing so, the LOADng Router MUST also add a tuple (P_next_hop, P_originator, P_seq_num, P_ack_timeout) to the Pending Acknowledgement Set, and set P_ack_timeout to RREP_ACK_TIMEOUT.

15.1. RREP_ACK Generation

Upon reception of an RREP message with the ackrequired flag set, a LOADng Router MUST generate an RREP_ACK and send this RREP_ACK in unicast to the neighbor which originated the RREP.

A packet with an RREP_ACK message is generated by a LOADng Router with the following content:

- <type> := RREP_ACK;
- <addr-length> := the length of the address, as specified in Section 8;
- <seq-num> := the <seq-num> field of the received RREP;
- <originator> := the <originator> field of the received RREP.

15.2. RREP_ACK Processing

On receiving an RREP_ACK from a LOADng neighbor router, a LOADng Router MUST do the following:

1. The TLV fields are added/removed/updated according to their specification.
2. Find the Routing Tuple (henceforth, matching Routing Tuple) where:
   * R_dest_addr = previous-hop;

   and update the tuple with:

   * R_bidirectional := TRUE

3. Check whether a corresponding RREP is pending, i.e. if the Pending Acknowledgment Set contains a tuple (P_next_hop, P_originator, P_seq_num, P_ack_timeout) such as:
   * P_next_hop is the address of the LOADng neighbor router from which the RREP_ACK was received.
   * P_originator corresponds to the <originator> field of the

RREP_ACK.

* P_seq_num corresponds to the <seq-num> field of the RREP_ACK.

4. If such a tuple exists, then the RREP has been correctly acknowledged and the tuple MUST be discarded.

5. Otherwise, i.e. if no such tuple exists, then no further processing is required.

15.3. RREP_ACK Forwarding

An RREP_ACK is intended only for a specific direct neighbor, and MUST NOT be forwarded.

15.4. RREP_ACK Transmission

An RREP_ACK is transmitted, in unicast, to the neighbor LOADng Router from which the RREP was received.

16. Metrics

This specification enables the use of different metrics for when calculating route costs, and specifies one particularly simplified such metric in Section 16.3, for use as a default ensuring interoperability even if routers in a network are configured to use different metrics. It is encouraged that more appropriate metrics be developed for different deployment environments.

16.1. The <= Comparison Operator

The objective of the <= comparison operator is to be able to determine which of two routes is "better", i.e., which route has the lowest cost. A link between a pair of interfaces may have a nominal and administratively assigned cost associated (such as, for example, representing a nominal bandwidth), however may also have a dynamic component making a link with an otherwise low cost a less attractive choice for when establishing a new route (such as, for example, if a high loss-rate is experienced across that link).

16.2. Specifying New Metrics

When defining a metric, the following considerations SHOULD be taken into consideration, and MUST be taken into consideration when requesting a code-point from IANA for the 1-8 range of the Cost Types registry defined in Table 3:

- The definition of the R_dist field, as well as the value of MAX_DIST.

- The mechanism for determining when a link qualifies as a "Weak Link". Examples include when an SNR or SIR is above/below a given threshold, etc. This MAY be by way of lower-layer information, message statistics or any other means.

- The required TLVs for calculating the route cost, as well as the mechanism for determining how to update those fields when an RREP or RREQ is transmitted over an interface.

- The <= comparison operator, which MUST specify a strict ordering of the R_dist space, i.e. R_dist1 can always be compared to R_dist2 and (R_dist1 <= R_dist2 && R_dist2 <= R_dist1) if and only if R_dist1 = R_dist2.

16.3. Default Metric: Hop Count With Weak Links

This section specifies a simple "Hop-Count-With-Weak-Links" metric, which is both the default metric provided for interoperability, and is intended to exemplify of how to specify metrics in general. It represents a simple "hop count" based cost, permitting avoiding weak links. It is RECOMMENDED to define a more appropriate metric for the environment in which the protocol is to operate.

16.3.1. R_dist Definition

R_dist := (HC, WL) where HC is the Hop Count, and WL the number of Weak Links. MAX_DIST := (255, 15).

16.3.2. Weak Link Definition

A link is considered a weak link when information is available from a lower layer, indicating that the link falls below an acceptable threshold according to that lower layer specification. For IEEE 802.15.4, for example, this can be derived from the Link Quality Indicator.

Otherwise, if such information is not available from a lower layer, a link is never considered a Weak Link.

16.3.3. Required TLVs

This metric requires no TLVs.

16.3.4. The <= Comparison Operator

Let (HC, WL) be the pair (hop-count, weak-links) received in one RREQ or RREP, and let (HC', WL') be the pair (hop-count, weak-links) received in another RREQ or RREP. The comparison operator <= is then defined as:

(HC,WL) <= (HC',WL') if and only if:

```
WL < WL'; OR
WL == WL' AND HC <= HC'
```

17. Security Considerations

Currently, this document does not specify any specific security measures. By way of enabling inclusion of TLVs, development of security measures, appropriate for a given deployment, is however supported.

18. IANA Considerations

18.1. Multicast Addresses

IANA is requested to allocate LL-LLN-ROUTERS well-known, link-scoped multicast addresses for both IPv4 and IPv6.

18.2. Packet Types

IANA is requested to create a new registry for packet types, with initial assignments and allocation policies as specified in Table 1.

Type	Description	Allocation Policy
0	Route Request (RREQ)	
1	Route Reply (RREP)	
2	Route Error (RERR)	
3	Route Reply Acknowledgement (RREP_ACK)	
4-8	Unassigned	Expert Review
9-15	Unassigned	Experimental Use

Table 1: Packet Types

18.3. TLV Types

IANA is requested to create a new registry for TLV types, with initial assignments and allocation policies as specified in Table 2.

Type	Description	Allocation Policy
0-7	Unassigned	Expert Review
8-15	Unassigned	Experimental Use

Table 2: TLV Types

18.4. Metrics

IANA is requested to create a new registry for Metrics, with initial assignments and allocation policies as specified in Table 3.

Code	Description	Allocation Policy
0	Hop Count While Avoiding Weak Links (Section 16)	
1-8	Unassigned	Expert Review
9-15	Unassigned	Experimental Use

Table 3: Metrics

When assigning a new Metric, the specification requesting that assignment MUST specify the way in which each LOADng Router calculates the <hop-count> field and TLVs for calculating the route cost in RREQs and RREPs, as well as the criteria for incrementing the <weak-links> field in RREQs and RREPs. The specification MUST also specify the comparison operation '<=' for determining, from among two RREQs (or RREPs) for the same destination, which message represents

the shortest route; note that this comparison operation SHOULD involve the <hop-count> field and MAY use other information such as <weak-links> or content of specific TLV types included in the RREQ or RREP.

18.5. Error Codes

IANA is requested to create a new registry for Error Codes, with initial assignments and allocation policies as specified in Table 4.

Code	Description	Allocation Policy
0	No available route	
1-8	Unassigned	Expert Review
9-15	Unassigned	Experimental Use

Table 4: Error Codes

19. Contributors

This specification is the result of the joint efforts of the following contributors -- listed alphabetically.

- Alberto Camacho, LIX, France, <alberto@albertocamacho.com>
- Thomas Heide Clausen, LIX, France, <T.Clausen@computer.org>
- Axel Colin de Verdiere, LIX, France, <axel@axelcdv.com>
- Kenneth Garey, Maxim Integrated Products, USA, <kenneth.garey@maxim-ic.com>
- Ulrich Herberg, Fujitsu Laboratories of America, USA <ulrich.herberg@us.fujitsu.com>
- Yuichi Igarashi, Hitachi Ltd, Yokohama Research Laboratory, Japan, <yuichi.igarashi.hb@hitachi.com>
- Afshin Niktash, Maxim Integrated Products, USA, <afshin.niktash@maxim-ic.com>

- Hiroki Satoh, Hitachi Ltd, Yokohama Research Laboratory, Japan, <hiroki.satoh.yj@hitachi.com>

- Jiazi Yi, LIX, France, <jiazi@jiaziyi.com>

20. Acknowledgments

The authors would like to acknowledge the team behind AODV, specified in RFC3561 for their contributions. The authors would also like to acknowledge the efforts of K. Kim (picosNet Corp/Ajou University), S. Daniel Park (Samsung Electronics), G. Montenegro (Microsoft Corporation), S. Yoo (Ajou University) and N. Kushalnagar (Intel Corp.) for their work on an initial version of a specification, from which this protocol is derived.

21. References

21.1. Normative References

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

21.2. Informative References

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

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, September 2007.

[RFC6206] Levis, P., Clausen, T., Gnawali, O., and J. Ko, "The Trickle Algorithm", RFC 6206, March 2011.

[SMF] Macker, J., "Simplified Multicast Forwarding", draft-ietf-manet-smf-14 (work in progress), March 2012.

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

RFC 5148, February 2008.

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

[Stevens] Stevens, W., "TCP/IP Illustrated Volume 1 - The Protocols", 1994.

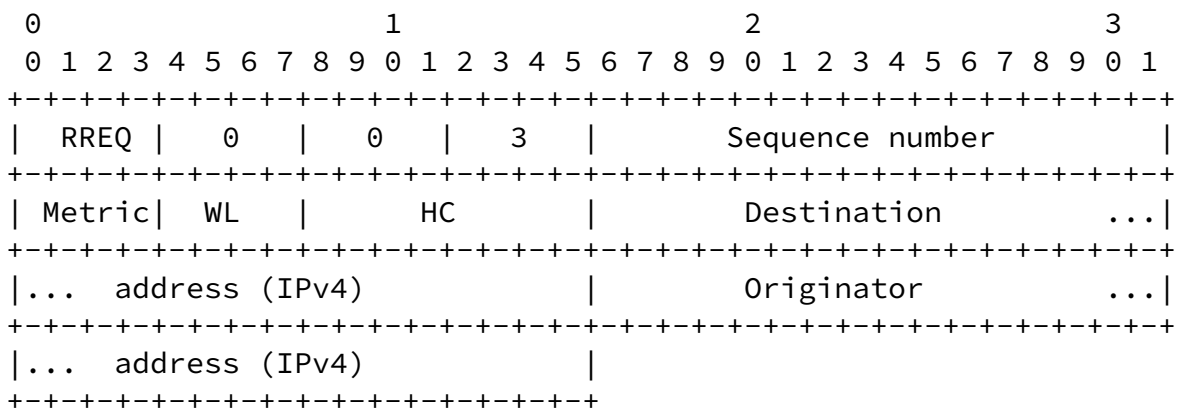
[SingleUNIX] IEEE Std 1003.1, The Open Group, and ISO/IEC JTC1/SC22/WG15, "Single UNIX Specification, Version 3, 2004 Edition", April 2004.

Appendix A. LOADng Control Packet Illustrations

This section presents example packets following this specification.

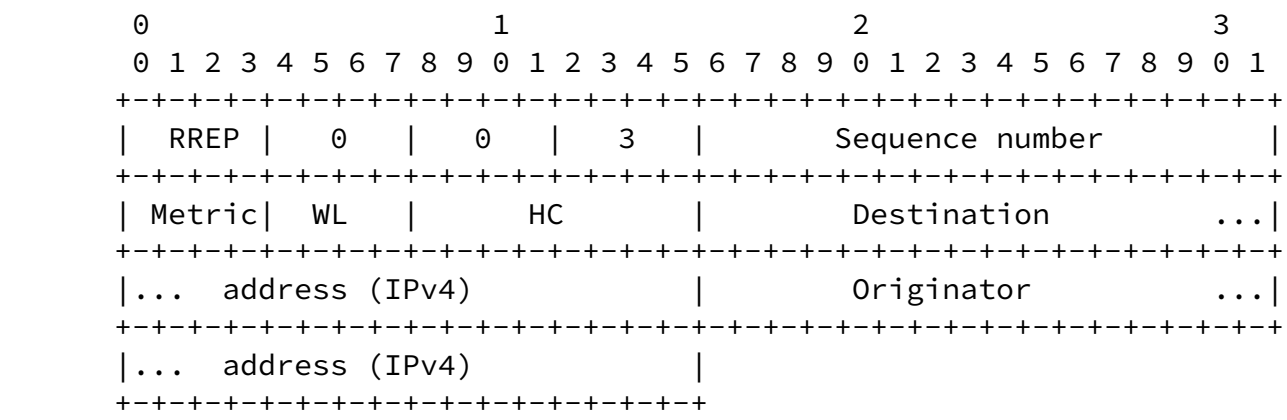
A.1. RREQ

This figure depicts the format of a sample packet with an RREQ message using IPv4 addresses. The packet is as follows:



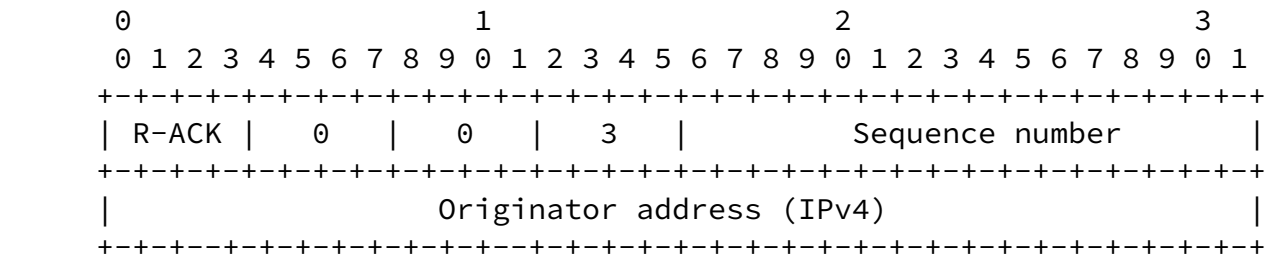
A.2. RREP

This figure depicts the format of a sample packet with an RREP message using IPv4 addresses. The packet is as follows:



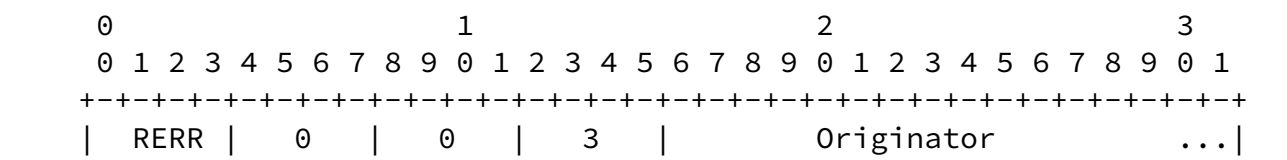
A.3. RREP_ACK

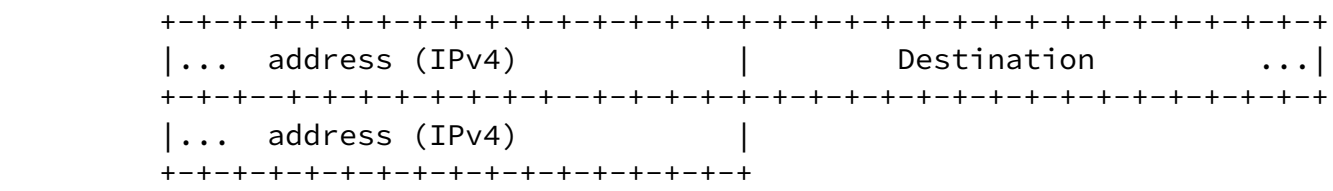
This figure depicts the format of a sample packet with an RREP_ACK message using IPv4 addresses, as follows:



A.4. RERR

This figure depicts the format of a sample packet with an RERR message using IPv4 addresses, as follows:





Authors' Addresses

Thomas Heide Clausen
LIX, Ecole Polytechnique

Phone: +33 6 6058 9349
EMail: T.Clausen@computer.org
URI: http://www.ThomasClausen.org/

Axel Colin de Verdiere
LIX, Ecole Polytechnique

Phone: +33 6 1264 7119
EMail: axel@axelcdv.com
URI: http://www.axelcdv.com/

Jiazi Yi
LIX, Ecole Polytechnique

Phone: +33 1 6933 4031
EMail: jiazi@jiaziyi.com
URI: http://www.jiaziyi.com/

Afshin Niktash
Maxim Integrated Products

Phone: +1 94 9450 1692

EMail: afshin.niktash@maxim-ic.com
URI: http://www.Maxim-ic.com/

Yuichi Igarashi
Hitachi, Ltd., Yokohama Research Laboratory

Phone: +81 45 860 3083
EMail: yuichi.igarashi.hb@hitachi.com
URI: http://www.hitachi.com/

Hiroki Satoh
Hitachi, Ltd., Yokohama Research Laboratory

Phone: +81 44 959 0205
EMail: hiroki.satoh.yj@hitachi.com
URI: http://www.hitachi.com/

Ulrich Herberg
Fujitsu Laboratories of America

Phone: +1 408 530 4528
EMail: ulrich@herberg.name
URI: http://www.herberg.name/

Cedric Lavenu
EDF R&D

Phone: +33 1 4765 2729
EMail: cedric-2.lavenu@edf.fr
URI: http://www.edf.fr/

Thierry Lys
ERDF

Phone: +33 1 8197 6777
EMail: thierry.lys@erdfdistribution.fr
URI: http://www.erdfdistribution.fr/