Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 15, 2013

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
C. Perkins
Futurewei Inc.
July 14, 2012

The LLN On-demand Ad hoc Distance-vector Routing Protocol - Next Generation (LOADng)

draft-clausen-lln-loadng-05

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).

Status of This Memo

This Internet-Draft will expire on January 15, 2013.

Copyright Notice

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 LOADng Router (originator) for when discovering a route to a destination, forwarding of such RREQs until they reach the destination LOADng 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 towards the intended 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:

o Only the destination is permitted to respond to an RREQ; intermediate LOADng 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 LOADng 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.

o 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:

o 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.

o Different address lengths are supported - from full 16 octet IPv6 addresses over 8 octet EUI64 addresss [EUI64], 6 octet MAC addresses. 4 octet IPv4 addresses to shorter 1 and 2 octet addresses such as [RFC4944]. The only requirement is, that within a given routing domain, all addresses are of the same address

length.

o Control messages are carried by way of the Generalized MANET Packet/Message Format [RFC5444].

o Using [RFC5444], control messages can include TLV (Type-Length-Value) elements, permitting protocol extensions to be developed.

LOADng supports routing using arbitrary additive metrics, which can be specified as extensions to this protocol.

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, Section 2.2, and Section 2.3 and the terminology defined in Section 2.4.

2.1. Message and Message Field Notation

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

MsgType.field

where:

MsgType - is the type of message (e.g., RREQ or RREP);

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

The different messages, their fields and their meaning are described in Section 6. The encoding of messages for transmission by way of [RFC5444] packets/messages is described in Appendix A, and Appendix B illustrates the bit layout of a selection of LOADng control messages.

The motivation for separating the high-level messages and their content from the low-level encoding and frame format for transmission is to allow discussions of the protocol logic to be separated from the message encoding and frame format - and, to support different frame formats.

2.2. Variable Notation

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

MsgType.field - If field is an unsigned integer field in the message MsgType, then MsgType.field 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).

2.3. Other Notation

This document uses the following additional 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).

2.4. Terminology

This document uses the following terminology:

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

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

Link - A link between two LOADng Interfaces exists if either can receive control messages, according to this specification, from the other.

Message - The fundamental entity carrying protocol information, in the form of address objects and TLVs.

Link Metric - The cost (weight) of a link between a pair of LOADng Interfaces.

Route Metric - The sum of the Link Metrics for the links that an

RREQ or RREP has crossed.

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 LOADng Router to discover bi-directional routes to destinations in the routing domain, i.e., to any other LOADng Router, as well as hosts or networks attached to that LOADng Router, in the same routing domain.

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

- Maintain a route only for as long as there is data traffic being sent along that 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 and where the data packet source is local to that LOADng Router (i.e., is an address in the Local Interface Set or Destination Address Set of that LOADng Router), generate a Route Request (RREQ) encoding the destination address, and transmit this RREQ over all of its LOADng Interfaces.

- Upon receiving an RREQ, insert or refresh a tuple in the Routing Set, recording a route 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 or in the Local Interface 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 or in the Local Interface 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, insert or refresh a tuple in the Routing Set, recording 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 LOADng Router).

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

o When forwarding an RREQ or RREP, update the route metric, as contained in that RREQ or RREP message.

A LOADng Router generating an RREQ specifies which metric type it desires. Routers receiving an RREQ will process it and update route metric information in the RREQ according to that metric, if they can. All LOADng Routers, however, will update information in the RREQ so as to be able to support a "hop-count" default metric. If a LOADng Router is not able to understand the metric type, specified in an RREQ, it will update the route metric value to its maximum value, so as to ensure that this is indicated to the further recipients of the RREQ. Once the route metric value is set to its maximum value, no LOADng Router along the path towards the destination may change the value.

4.2. LOADng Routers and LOADng Interfaces

A LOADng Router has a set of at least one, and possibly more, LOADng Interfaces. Each LOADng Interface:

o Is configured with one or more addresses.

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

o Has a number of router parameters.

o Has an Information Base.

o 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 LOADng 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 LOADng Interface of the LOADng Router. Each tuple contains a list of

one or more addresses of that LOADng 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., addresses of this LOADng Router, or of hosts and networks directly attached to this LOADng Router and which use it to connect to the routing domain. These addresses may in particular belong to devices which do not implement LOADng, and thus cannot process LOADng messages. A LOADng Router provides 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 Local Interface Set and the Destination Address Set are 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, where the data packet source is local to that LOADng Router (i.e., is an address in the Local Interface Set or Destination Address Set of that

LOADng Router), but where 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 Acknowledgment (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

The following parameters are used in this specification.

5.1. Protocol and Port Numbers

When using LOADng as an IP routing protocol, the considerations of [RFC5498] apply.

5.2. Router Parameters

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

RREQ_RETRIES - is the maximum number of subsequent RREQs that a particular LOADng 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 LOADng 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.

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

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_TRAVERSAL_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.

RREQ_MAX_JITTER - is the default value of MAXJITTER used in [RFC5148] for RREQ messages forwarded by this LOADng Router.

MAX_HOP_COUNT - is the maximum number of transmissions permitted by any RREQ or RREP message.

5.3. Interface Parameters

Different LOADng Interfaces (on the same or on different LOADng Routers) MAY employ different interface parameter values and MAY change their interface parameter values dynamically. A particular case is where all LOADng Interfaces on all LOADng Routers within a given LOADng routing domain employ the same set of interface parameter values.

RREP_ACK_REQUIRED - is a boolean flag, which indicates (if set) that the LOADng 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 amount of time after transmission of an RREP, that a LOADng Router SHOULD wait for an RREP_ACK from a neighbor LOADng Router, before considering the link to this neighbor to be unidirectional.

6. Protocol Message Content

The protocol messages, generated and processed by LOADng, are described in this section using the notational conventions described in Section 2. The encoding of messages for transmission by way of [RFC5444] packets/messages is described in Appendix A, and Appendix B illustrates the bit layout of a selection of LOADng control messages. Unless stated otherwise, the message fields described below are set by the LOADng Router, which generates the message, and MUST NOT be changed by intermediate LOADng Routers.

6.1. Route Request (RREQ) Messages

A Route Request (RREQ) message has the following fields:

RREQ.addr-length is an unsigned integer field, encoding the length of the originator and destination addresses as follows:

RREQ.addr-length := the length of an address in octets - 1

RREQ.seq-num is an unsigned integer field, containing the sequence number (see Section 8) of the LOADng Router, generating the RREQ message.

RREQ.metric-type is an unsigned integer field and specifies the type of metric requested by this RREQ.

RREQ.route-metric is a unsigned integer field, of length defined by RREQ.metric-type, which specifies the route metric of the route (the sum of the link metrics of the links), through which this RREQ has traveled.

RREQ.hop-count is an unsigned integer field and specifies the total number of hops which the message has traversed from the RREQ.originator.

RREQ.originator is an identifier of <address-length> + 1 octets, specifying the address of the LOADng Interface over which this RREQ was generated, and to which a route (the "reverse route") is supplied by this RREQ. In case the message is generated by a

LOADng Router on behalf of an attached host, the <originator> address corresponds to an address of that host, otherwise it corresponds to an address of the sending LOADng Interface of the LOADng Router.

RREQ.destination is an identifier of <address-length> + 1 octets, specifying the address to which the RREQ should be sent, i.e., the destination address for which a route is sought.

The following fields of an RREQ message are immutable, i.e., they MUST NOT be changed during processing or forwarding of the message: RREQ.addr-length, RREQ.seq-num, RREQ.originator, and RREQ.destination.

The following fields of an RREQ message are mutable, i.e., they will be changed by intermediate routers during processing or forwarding, as specified in Section 12.2 and Section 12.3: RREQ.metric-type, RREQ.route-metric, and RREQ.hop-count.

Any additional field that is added to the message by an extension to this protocol, e.g., by way of TLVs, MUST be considered immutable, unless the extension specifically defines the field as mutable.

6.2. Route Reply (RREP) Messages

A Route Reply (RREP) message has the following fields:

RREP.addr-length is an unsigned integer field, encoding the length of the originator and destination addresses as follows:

    RREP.addr-length := the length of an address in octets - 1

RREP.seq-num is an unsigned integer field, containing the sequence number (see Section 8) of the LOADng Router, generating the RREP message.

RREP.metric-type is an unsigned integer field and specifies the type of metric, requested by this RREP.

RREP.route-metric is a unsigned integer field, of length defined by RREP.metric-type, which specifies the route metric of the route (the sum of the link metrics of the links) through which this RREP has traveled.

RREP.ackrequired is a boolean flag which, when set ('1'), at least one 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.

RREP.hop-count is an unsigned integer field and specifies the total number of hops which the message has traversed from the <originator> to the <destination>.

RREP.originator is an identifier of <address-length> + 1 octets, specifying the address for which this RREP was generated, and to which a route (the "forward route") is supplied by this RREP. In case the message is generated on a LOADng Router on behalf of an attached host, the <originator> address corresponds to an address of that host, otherwise it corresponds to an address of the LOADng Interface of the LOADng Router, over which the RREP was generated.

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

The following fields of an RREP message are immutable, i.e., they MUST NOT be changed during processing or forwarding of the message: RREP.addr-length, RREP.seq-num, RREP.originator, and RREP.destination.

The following fields of an RREP message are mutable, i.e., they will be changed by intermediate routers during processing or forwarding, as specified in Section 13.2 and Section 13.3: RREP.metric-type, RREP.route-metric, RREP.ackrequired, and RREP.hop-count.

Any additional field that is added to the message by an extension to this protocol, e.g., by way of TLVs, MUST be considered immutable, unless the extension specifically defines the field as mutable.

6.3. Route Reply Acknowledgement (RREP_ACK) Messages

A Route Reply Acknowledgement (RREP_ACK) message has the following fields:

RREP_ACK.addr-length is an unsigned integer field, encoding the length of the destination and originator addresses as follows:

RREP_ACK.addr-length := the length of an address in octets - 1

RREP_ACK.seq-num is an unsigned integer field and contains the value of RREP.seq-num from the RREP for which this RREP_ACK is sent.

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

RREP_ACK messages are sent only across a single link and are never forwarded.

6.4. Route Error (RERR) Messages

A Route Error (RERR) message has the following fields:

RERR.addr-length is an unsigned integer field, encoding the length of RERR.destination and RERR.unreachableAddress, as follows:

RERR.addr-length := the length of an address in octets - 1

RERR.errorcode is an unsigned integer field and specifies the reason for the error message being generated, according to Table 1.

RERR.unreachableAddress is an identifier of RERR.addr-length + 1 octets, specifying an address, which has become unreachable, and for which an error is reported by way of this RERR message.

RERR.destination is an identifier of RERR.address-length + 1 octets, specifying the destination address of this RERR message. RERR.destination is, in general, the source address of a data packet, for which delivery to RERR.unreachableAddress failed, and the unicast destination of the RERR message is the LOADng Router

which has RERR.destination listed in a Local Interface Tuple or in a Destination Address Tuple.

The following fields of an RERR message are immutable, i.e., they MUST NOT be changed during processing or forwarding of the message: RERR.addr-length, RERR.errorcode, RERR.unreachableAddress, and RERR.destination.

Any additional field that is added to the message by an extension to this protocol, e.g., by way of TLVs, MUST be considered immutable, unless the extension specifically defines the field as mutable.

7. 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.

7.1. 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_metric, R_metric_type, R_hop_count,
   R_seq_num, R_valid_time,  R_bidirectional, R_local_iface_addr)
```

where:

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

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

R_metric - is the metric associated with the selected route to the destination with address R_dest_addr.

R_metric_type - specifies the metric type for this Routing Tuple - in other words, how R_metric is defined and calculated.

R_hop_count - is the hop count of the selected route to the destination with address R_dest_addr.

R_seq_num - is the value of the RREQ.seq-num or RREP.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 Routing 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 LOADng Router is configured as accepting routes without bi-directionality verification explicitly by setting USE_BIDIRECTIONAL_LINK_ONLY to FALSE.

R_local_iface_addr - is an address of the local LOADng Interface, through which the destination can be reached.

7.2. Local Interface Set

A LOADng Router's Local Interface Set records its local LOADng Interfaces. It consists of Local Interface Tuples, one per LOADng Interface:

(I_local_iface_addr_list)

where:

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

The implementation MUST initialize the Local Interface Set with at least one tuple containing at least one address of an LOADng Interface. The Local Interface Set MUST be updated if there is a change of the LOADng Interfaces of a LOADng Router (i.e., a LOADng Interface is added, removed or changes addresses).

7.3. Blacklisted Neighbor Set

The Blacklisted Neighbor Set records the neighbor LOADng 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 LOADng Interface.

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

7.4. 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 LOADng 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 routing domain.

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.

7.5. Pending Acknowledgment Set

The Pending Acknowledgment Set contains information about RREPs which have been transmitted with the RREP.ackrequired 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 LOADng Interface to which the RREP was sent.

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

P_seq_num - is the RREP.seq-num field of the sent RREP.

P_ack_timeout - is the time after which the tuple MUST be discarded.

8. 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 LOADng 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 LOADng 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

9. Route Maintenance

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

- o RREQ/RREP exchange, specified in Section 12 and Section 13.
- o Data traffic delivery success.
- o Data traffic delivery failure.
- o External signals indicating that a tuple in the Routing Set necessitates updating.
- o 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., routes 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. Note that when refreshing a Routing Tuple corresponding to a destination of a data packet, the Routing Tuple corresponding to the next hop toward that destination SHOULD also be refreshed.

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).

If generation of an RERR message is triggered by failure to deliver a data packet to a next-hop, a LOADng Router MUST generate an RERR message, as specified in Section 14, and MAY attempt an alternative delivery method for that (and subsequent) data packets, e.g., as specified in [I-D.DFF].

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 acknowledgments, MAY be used to detect unidirectional or broken links.

When a LOADng Router detects, via one of these mechanisms, that a link to a neighbor LOADng Router is unidirectional or broken, the LOADng 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 LOADng 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 7.3. When a neighbor 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 LOADng Router

- B_valid_time := current_time + B_HOLD_TIME

When a neighbor LOADng 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:

- Every RREQ received from this neighbor LOADng Router 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 LOADng Routers on a specific route towards a specific destination.

- o Receipt of an RREQ message by a LOADng router, which has the RREQ.destination address in its Local Interface Set or Destination Address Set MUST trigger the procedures for generation of an RREP message.

- o Receipt of an RREP message with RREP.ackrequired set MUST trigger generation of an RREP_ACK message.

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

received-route-metric is a variable, representing the route metric, as included in the received RREQ or RREP message, see Section 16.

used-metric-type is a variable, representing the type of metric used for calculating received-route-metric, 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 sequence numbers, as specified in Section 8.

MSG is a shorthand for either a RREQ or RREP message, used for when accessing message fields in the description of the common RREQ and RREP message processing in the following subsections.

link-metric is a variable, representing the link metric between this LOADng Router and the LOADng Router from which the RREQ or RREP message was received, as calculated by the receiving LOADng

Router, see Section 16.

route-metric is a variable, representing the route metric, as included in the received RREQ or RREP message, plus the link-metric for the link, over which the RREQ or RREP was received, i.e., the total route cost from the originator to this LOADng

Router.

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:

- o The address length specified by this message (i.e., MSG.addr-length + 1) differs from the length of the address(es) of this LOADng Router.
- o The address contained in the <originator> field is an address of this LOADng Router.
- o There is a tuple in the Routing Set where:
  - * R_dest_addr = MSG.originator
  - * R_seq_num > MSG.seq-num
- o 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 integrity check values 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 MSG.metric-type is known to this LOADng Router, then:

   * Set the variable used-metric-type to the value of MSG.metric-type.

   * Determine the link metric over the link over which the message was received, according to used-metric-type, and set the variable link-metric to the calculated value.

   * Compute the route metric to MSG.originator according to used-metric-type by adding link-metric to the received-route-metric advertised by the received message, and set the variable route-metric to the calculated value.

3. Otherwise, if MSG.metric-type is unknown to this LOADng Router:

   * Set the variable used-metric-type to HOP_COUNT.

   * Set the variable route-metric to its maximum value, see Section 16.

4. Find the Routing Tuple (henceforth, Matching Routing Tuple) where:

   * R_dest_addr = MSG.originator

   * R_metric_type = used-metric-type

5. 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 := MSG.originator

   * R_next_addr := previous-hop

   * R_metric_type := used-metric-type

* R_metric := MAX_DIST

* R_hop_count := MSG.hop-count

* R_seq_num := -1

* R_valid_time := current time + R_HOLD_TIME

* R_bidirectional := FALSE

* R_local_iface_addr := the address of the LOADng Interface through which the message was received.

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

   1. If

      + R_seq_num = MSG.seq-num; AND

      + R_metric > route-metric

      OR

      + R_seq_num = MSG.seq-num; AND

      + R_metric = route-metric; AND

      + R_hop_count > MSG.hop-count

      OR

      + R_seq_num < MSG.seq-num

      Then:

   7. The message is used for updating the Routing Set. The Routing Tuple, where:

- R_dest_addr = MSG.originator; AND

- R_metric_type = used-metric-type

is updated thus:

- R_next_addr := previous-hop

- R_metric := route-metric

- R_hop_count := MSG.hop-count

- R_seq_num := MSG.seq-num

- R_valid_time := current time + R_HOLD_TIME

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

8. If previous-hop is not equal to MSG.originator, and 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

9. The Routing Tuple with R_dest_addr = previous-hop, existing or new, is updated as follows

   - R_metric_type := used-metric-type

   - R_metric := link-metric

   - R_hop_count := 1

   - 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 address of the LOADng Interface through which the message was received.

2. Otherwise, the RREQ or RREP message is not processed further, and is not considered for forwarding.

12. Route Requests (RREQs)

Route Requests (RREQs) are generated by a LOADng Router when it has data packets to deliver to a destination, where the data packet source is local to that LOADng Router (i.e., is an address in the Local Interface Set or Destination Address Set of that LOADng Router), but for which the LOADng router has no matching tuple in the Routing Set. If the router parameter USE_BIDIRECTIONAL_LINK_ONLY is TRUE, a RREQ is furthermore generated even if a matching tuple in the Routing Set exists, but where that tuple has R_bidirectional set to FALSE. The RREQ is transmitted on all the LOADng Routers LOADng Interfaces, i.e., 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

An RREQ message is generated according to Section 6 with the following content:

- RREQ.addr-length set to the length of the address, as specified in Section 6;

o RREQ.metric-type set to the desired metric type;

o RREQ.seq-num set to the next unused sequence number, maintained by this LOADng Router;

o RREQ.hop-count := 0;

o RREQ.destination := the address to which a route is sought;

o RREQ.originator := one address of the LOADng Interface of the LOADng Router that generates the RREQ. If the LOADng Router is generating RREQ on behalf of a host connected to this LOADng Router, the source address of the data packet, generated by that host, is used;

o RREQ.route-metric := 0.

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 RREQ.hop-count equals MAX_HOP_COUNT, the message is not considered for forwarding.

4. If RREQ.destination is not listed in I_local_iface_addr_list of any Local Interface Tuple, or does not 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.

5. Otherwise, an the RREP generation process in Section 13.1 MUST be applied. The RREQ is not considered for forwarding.

12.3. RREQ Forwarding

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

received-route-metric is a variable, representing the route metric, as included in the received RREQ message, see Section 16.

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

link-metric is a variable, representing the link metric between this LOADng Router and the LOADng Router, from which the RREQ message was received, as calculated based on the received message or on other mechanisms, see Section 16.

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

1. RREQ.metric-type := used-metric-type (as set in Section 11.2)

2. RREQ.hop-count := RREQ.hop-count + 1

3. RREQ.route-metric := received-route-metric + link-metric

   Where link-metric is calculated according to the specification of RREQ.metric-type.

An RREQ MIST be forwarded according to the flooding operation, specified for the network. This MAY be by way of classic flooding, a reduced relay set mechanism such as [RFC6621], 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 traverse 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. The source address of the RREQ MUST be an address of the LOADng Interface over which the RREQ is sent.

When an RREQ is transmitted, all receiving LOADng Routers will process the RREQ message and as a consequence 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], in order to avoid such losses.

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 from RREP.destination. RREPs are sent, hop by hop, in unicast towards the originator of the RREQ, in response to which the RREP was generated, along the Reverse Route installed by that RREQ. A LOADng Router, upon forwarding an RREP, installs the Forward Route towards the RREP.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 RREQ.originator and RREQ.destination.

13.1. RREP Generation

At least one RREP MUST be generated in response to a (set of) received RREQ messages with identical (RREP.originator, RREP.seq-num). An RREP MAY be generated immediately as a response to each RREQ processed, in order to provide shortest possible route establishment delays, or MAY be generated after a certain delay after the arrival of the first RREQ, in order to use the "best" received RREQ (e.g., received over the lowest-cost route) but at the expense of longer route establishment delays. A LOADng Router MAY generate further RREPs for subsequent RREQs received with the same (RREP.originator, RREP.seq-num) pairs, if these indicate a better route, at the expense of additional control traffic being generated. In all cases, however, the content of an RREP is as follows:

- o RREP.addr-length set to the length of the address, as specified in Section 6;

- o RREP.seq-num set to the next unused sequence number, maintained by this LOADng Router;

- o RREP.metric-type set to the same value as the RREQ.metric-type in the corresponding RREQ;

- o RREP.hop-count := 0;

o RREP.destination := the address to which this RREP message is to be sent; this corresponds to the RREQ.originator from the RREQ message, in response to which this RREP message is generated;

o RREP.originator := the address of the LOADng Router, generating the RREP. If the LOADng Router is generating an RREP on behalf of the hosts connected to it, or on behalf of one of the addresses contained in the LOADng Routers Destination Address Set, the host address is used.

o RREP.route-metric := 0

The RREP so generated is transmitted according to Section 13.4.

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 RREP.ackrequired is set, an RREP_ACK message MUST be sent to the previous-hop, according to Section 15.1.

4. If the RREP.hop-count is equal to MAX_HOP_COUNT, the message is not considered for forwarding.

5. If RREP.destination 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

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

received-route-metric is a variable, representing the route metric, as included in the received RREP message, see Section 16.

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

link-metric is a variable, representing the link metric between this LOADng Router and the LOADng Router, from which the RREP message was received, as calculated based on the received message or on other mechanisms, see Section 16.

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

1. RREP.metric-type := used-metric-type (as set in Section 11.2)

2. RREP.hop-count := RREP.hop-count + 1

3. RREP.route-metric := received-route-metric + link-metric

   Where link-metric is calculated according to the specification of RREP.metric-type.

4. The RREP is transmitted, according to Section 13.4.

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 which has the address listed in the RREP.destination field in either of its Local Interface Set, or in its Destination Address Set. The RREP is forwarded in unicast towards that 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; AND
- Permit that RREP.hop-count be updated to reflect the route.

RREP Transmission is accomplished by the following procedure:

1. Find the Routing Tuple (henceforth, the "Matching Routing Tuple") in the Routing Set, where:

   * R_dest_addr = RREP.destination

   * R_metric_type = RREP.metric-type

   * R_metric is minimum

2. Find the Local Interface Tuple (henceforth, "Matching Interface Tuple), where:

   * I_local_iface_addr_list contains R_local_iface_addr from the Matching Routing Tuple

3. If RREP_ACK_REQUIRED is set for the LOADng Interface, identified by the Matching Interface Tuple:

   * Create a new Pending Acknowledgment Tuple with:

     + P_next_hop := R_next_addr from the Matching Routing Tuple

     + P_originator := RREP.originator

     + P_seq_num := RREP.seq-num

     + P_ack_timeout := current_time + RREP_ACK_TIMEOUT

   * Set RREP.ackrequired to true

4. Otherwise:

   * Set RREP.ackrequired to false.

5. The RREP is transmitted over the LOADng Interface, identified by the Matching Interface Tuple to the neighbor LOADng Router, identified by R_next_addr from the Matching Routing Tuple.

14. Route Errors (RERRs)

If a LOADng Router fails to deliver a data packet to a next hop or a destination, and if neither the source nor destination address of that data packet is not in the Destination Address Set of that LOADng Router, it MUST generate a Route Error (RERR). This RERR MUST be sent along the Reverse Route towards the source of the data packet for which delivery was unsuccessful (to the last LOADng Router along the Reverse Route, if the data packet was originated by a host behind that LOADng Router).

The following definition is used in this section:

- "EXPIRED" indicates that a timer is set to a value clearly preceding the current time (e.g., current time - 1).

14.1. Identifying Invalid RERR Messages

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

14.2. RERR Generation

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

- RERR.error-code := the error code corresponding to the event causing the RERR to be generated, from among those recorded in Table 1;

- RERR.addr-length := the length of the address, as specified in Section 6;

- RERR.destination := the source address from the unsuccessfully delivered data packet, towards which the RERR is to be sent.

o RERR.unreachableAddress := the destination address from the unsuccessfully delivered data packet.

14.3. 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 = RERR.unreachableAddress

   * 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.

14.4. RERR Forwarding

An RERR is, ultimately, destined for the LOADng Router which has, in either its Destination Address Set or in its Local Interface Set, the address from RERR.originator.

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 = RERR.destination

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 RERR.destination.

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 14.5.

14.5. RERR Transmission

An RERR is, ultimately, destined for the LOADng Router which has the address listed in the RERR.destination field in either of its Local Interface Set, or in its Destination Address Set. The RERR is forwarded in unicast towards that LOADng Router. The RERR MUST, however, be transmitted so as to allow it to be processed in each intermediate LOADng Router to:

o Allow intermediate LOADng Routers to update their Routing Sets, i.e., remove tuples for this destination.

RERR Transmission is accomplished by the following procedure:

1. Find the Routing Tuple (henceforth, the "Matching Routing Tuple") in the Routing Set, where:

   * R_dest_addr = RERR.destination

2. Find the Local Interface Tuple (henceforth, "Matching Interface Tuple), where:

   * I_local_iface_addr_list contains R_local_iface_addr from the Matching Routing Tuple

3. The RERR is transmitted over the LOADng Interface, identified by the Matching Interface Tuple to the neighbor LOADng Router, identified by R_next_addr from the Matching Routing Tuple.

15. Route Reply Acknowledgments (RREP_ACKs)

A LOADng Router MUST signal in a transmitted RREP that it is expecting an RREP_ACK, by setting RREP.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 Acknowledgment Set, and set P_ack_timeout to RREP_ACK_TIMEOUT, as described in Section 13.4.

The following definition is used in this section:

- "EXPIRED" indicates that a timer is set to a value clearly preceding the current time (e.g., current time - 1).

15.1. RREP_ACK Generation

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

An RREP_ACK message is generated by a LOADng Router with the following content:

- RREP_ACK.addr-length := the length of the address, as specified in Section 6;

- RREP_ACK.seq-num := the value of the RREP.seq-num field of the received RREP;

- RREP_ACK.destination := RREP.originator of the received RREP.

15.2. RREP_ACK Processing

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

1. Find the Routing Tuple (henceforth, Matching Routing Tuple) where:

   * R_dest_addr = previous-hop;

   The Matching Routing Tuple is updated as follows:

   * R_bidirectional := TRUE

2. If a Pending Acknowledgement Tuple (henceforth, Matching Pending Acknowledgement Tuple) exists, where:

   * P_next_hop is the address of the LOADng neighbor LOADng Router from which the RREP_ACK was received.

   * P_originator = RREP_ACK.destination

   * P_seq_num = RREP_ACK.seq-num

   Then the RREP has been correctly acknowledged. The Matching Pending Acknowledgement Tuple is updated as follows:

   * P_ack_timeout := EXPIRED

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 metrics.

Metrics as defined in LOADng are additive, and the routes that are to be created are those with the minimum sum of the metrics along that route.

16.1. Specifying New Metrics

When defining a metric, the following considerations SHOULD be taken

into consideration:

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

17. Security Considerations

Currently, this protocol does not specify any special security measures. As a reactive routing protocol, this protocol is a potential target for various attacks. Various possible vulnerabilities are discussed in this section.

By way of (i) enabling inclusion of TLVs and (ii) permitting that LOADng recognizes external reasons for rejecting RREQ, RREP, RREP_ACK and RERR messages, development of security measures, appropriate for a given deployment, is however supported. This architecture is a result of the observation that with respect to security in LOADng routed networks, "one size rarely fits all". This, as LOADng deployment domains have varying security requirements ranging from "unbreakable" to "virtually none", depending on, e.g., physical access to the network, or on security available on other layers. The virtue of this approach is that LOADng routing protocol specifications (and implementations) can remain "generic", with extensions providing proper deployment-domain specific security mechanisms.

17.1. Confidentiality

This protocol floods Route Requests (RREQs) to all the LOADng Routers in the network, when there is traffic to deliver to a given destination. Hence, if used in an unprotected network (such as an unprotected wireless network):

- o Part of the network topology is revealed to anyone who listens, specifically (i) the identity (and existence) of the source LOADng Router; (ii) the identity of the destination; and (iii) the fact that a path exists between the source LOADng Router and the LOADng Router from which the RREQ was received.

- o The network traffic patterns are revealed to anyone who listens to the LOADng control traffic, specifically which pairs of devices

communicate. If, for example, a majority of traffic originates from or terminates in a specific LOADng Router, this may indicate that this LOADng Router has a central role in the network.

This protocol also unicasts Route Replies (RREPs) from the destination of an RREQ to the originator of that same RREQ. Hence, if used in an unprotected network (such as an unprotected wireless network):

- o Part of the network topology is revealed to anyone who is near or on the unicast path of the RREP (such as within radio range of LOADng Routers on the unicast path in an unprotected wireless network), specifically that a path from the originator (of the RREP) to the destination (of the RREP) exists.

Finally, this protocol unicasts Route Errors (RERRs) when an intermediate LOADng Router detects that the path from a source to a destination is no longer available. Hence, if used in an unprotected network (such as an unprotected wireless network):

- o A disruption of the network topology is revealed to anyone who is near or on the unicast path of the RERR (such as within radio range of LOADng Routers on the unicast path in an unprotected wireless network), specifically that a path from the originator (of the RERR) to the destination (of the RERR) has been disrupted.

This protocol signaling behavior enables, for example, an attacker to identify central devices in the network (by monitoring RREQs) so as to target an attack, and (by way of monitoring RERRs) to measure the success of an attack.

17.2. Integrity

A LOADng Router injects topological information into the network by way of transmitting RREQ and RREP messages, and removes installed topological information by way of transmitting RERR messages. If some LOADng Routers for some reason, malice or malfunction, inject invalid control traffic, network integrity may be compromised. Therefore, message authentication is recommended.

Different such situations may occur, for instance:

1. A LOADng Router generates RREQ messages, pretending to be another LOADng Router;

2. A LOADng Router generates RREP messages, pretending to be another LOADng Router;

3. A LOADng Router generates RERR messages, pretending to be another LOADng Router;

4. A LOADng Router generates RERR messages, indicating that a link on a path to a destination is broken;

5. A LOADng Router forwards altered control messages;

6. A LOADng Router does not forward control messages;

7. A LOADng Router forwards RREPs and RREQs, but does not forward unicast data traffic;

8. A LOADng Router "replays" previously recorded control messages from another LOADng Router.

Authentication of the originator LOADng Router for control messages (for situations 1, 2 and 3) and on individual links announced in the control message (for situation 2 and 4) may be used as a countermeasure. However, to prevent routers from repeating old (and correctly authenticated) information (situation 8), temporal information is required, requiring a router to positively identify such a delayed message.

In general, integrity check values and other required security information may be transmitted as a separate Message Type, or signatures and security information may be transmitted within the control messages, using the TLV mechanism. Either option permits that "secured" and "unsecured" routers can coexist in the same network, if desired.

Specifically, if LOADng is used on the IP layer, the authenticity of entire control messages can be established through employing IPsec

authentication headers, whereas authenticity of individual links (situations 2 and 4) require additional security information to be distributed.

17.3. Channel Jamming and State Explosion

A reactive protocol, LOADng control messages are generated in response to network events. For RREQs, such an event is that a data packet is present in a router which does not have a route to the destination of the data packet, or that the router receives an RERR message, invalidating a route. For RREPs, such an event is receipt of an RREQ corresponding to a destination owned by the LOADng Router. A router, forwarding an RREQ or an RREP records state, for the reverse and forward routes, respectively. If some routers for some reason, malice or malfunction, generates excessive RREQ, RREP or RERRs, otherwise correctly functioning LOADng Routers may fall victim to either "indirect jamming" (being "tricked" into generating excessive control traffic) or an explosion in the state necessary for maintaining protocol state (potentially, exhausting the available memory resources).

Different such situations may occur, for instance:

1. A router, within a short time, generates RREQs to an excessive amount of destinations in the network (possibly all destinations, possibly even destinations not present in the network), causing intermediate routers to allocate state for the forward routes.

2. A router generates excessively frequent RREQs to the same (existing) destination, causing the corresponding LOADng Router to generate excessive RREPs.

3. A router generates RERRs for a destination to the source LOADng Router for traffic to that destination, causing that LOADng Router to flood renewed RREQs.

For situation 1, the state required for recording forward and/or reverse routes may exceed the memory available in the intermediate LOADng Routers - to the detriment of being able of recording state for other routes. This, in particular, if a LOADng Router generates RREQs for destinations "not present in the network".

A router which, within a short time, generates RREPs to an excessive amount of destinations in the network (possibly all destinations, possibly even destinations not present in the network), will not have the same network-wide effect: an intermediate router receiving an RREP for a destination for which no reverse route exists will neither attempt forwarding the RREP nor allocate state for the forward route.

For situations 1, 2, and 3, a possible countermeasure is to rate-limit the number of control messages that a LOADng Router forwards on behalf of another LOADng Router. Such a rate limit should take into consideration the expected normal traffic for a given LOADng deployment. Authentication may furthermore be used so as to prohibit a LOADng Router from forwarding control traffic from any non-authenticated router (with the assumption being that an authenticated router is not expected to exhibit such rogue behavior).

17.4. Interaction with External Routing Domains

This protocol does provide a basic mechanism for a LOADng Router to be able to discover routes to external routing domains: a LOADng Router configured to "own" a given set of addresses will respond to RREQs for destinations with these addresses, and can - by whatever protocols governing the routing domain wherein these addresses exist - provide paths to these addresses.

When operating routers connecting a LOADng domain to an external routing domain, destinations inside the LOADng domain can be injected into the external domain, if the routing protocol governing that domain so permits. Care MUST be taken to not allow potentially

insecure and untrustworthy information to be injected into the external domain.

In case LOADng is used on the IP layer, a RECOMMENDED way of extending connectivity from an external routing domain to a LOADng routed domain is to assign an IP prefix (under the authority of the routers/gateways connecting the LOADng routing domain with the external routing domain) exclusively to that LOADng routing domain, and to statically configure gateways to advertise routes for that prefix into the external domain. Within the LOADng domain, gateways SHOULD only generate RREPs for destinations which are not part of

that prefix; this is in particularly important if a gateway otherwise provides connectivity to "a default route".

18. LOADng Specific IANA Considerations

18.1. Error Codes

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

Code	Description	Allocation Policy
0	No available route	
1-251	Unassigned	Expert Review
252-255	Unassigned	Experimental Use

Table 1: Error Codes

19. Contributors

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

- o Alberto Camacho, LIX, France, <alberto@albertocamacho.com>

- o Thomas Heide Clausen, LIX, France, <T.Clausen@computer.org>

- o Axel Colin de Verdiere, LIX, France, <axel@axelcdv.com>

- o Kenneth Garey, Maxim Integrated Products, USA, <kenneth.garey@maxim-ic.com>

- o Ulrich Herberg, Fujitsu Laboratories of America, USA <ulrich.herberg@us.fujitsu.com>

- o Yuichi Igarashi, Hitachi Ltd, Yokohama Research Laboratory, Japan, <yuichi.igarashi.hb@hitachi.com>

- o Cedric Lavenu, EDF R&D, France, <cedric-2.lavenu@edf.fr>

o Afshin Niktash, Maxim Integrated Products, USA, <afshin.niktash@maxim-ic.com>

o Charles E. Perkins, Futurewei Inc, USA, <charliep@computer.org>

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

o Thierry Lys, ERDF, France, <thierry.lys@erdfdistribution.fr>

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

20. Acknowledgments

The authors would like to acknowledge the team behind AODV [RFC3561]. 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.

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

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

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.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman,

"Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

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

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

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

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

[I-D.DFF] Herberg, U., Ed., Cardenas, A., Iwao, T., Dow, M., and S. Cespedes, "Simplified Multicast Forwarding", draft-cardenas-dff-06 (work in progress), June 2012.

[EUI64] IEEE, "Guidelines for 64-bit Global Identifier (EUI-64) Registration Authority".

Appendix A. LOADng Control Messages using RFC5444

This section presents how the abstract LOADng messages, used throughout this specification, are mapped into RFC5444 messages.

A.1. RREQ-Specific Message Encoding Considerations

This protocol defines, and hence owns, the RREQ Message Type. Thus, as specified in [RFC5444], this protocol generates and transmits all RREQ messages, receives all RREQ messages and is responsible for determining whether and how each RREQ message is to be processed (updating the Information Base) and/or forwarded, according to this specification. Table 2 specifies how RREQ messages are mapped into [RFC5444]-elements.

RREQ Element	RFC5444-Element	Considerations
RREQ.addr-length	<msg-addr-length>	Supports addresses from 1-16 octets
RREQ.seq-num	<msg-seq-num>	16 bits, hence MAXVALUE (Section 8) is 65535. MUST be included
RREQ.metric-type	METRIC Message TLV	Encoded by way of the Type-Extension of a Message-Type-specific Message TLV of type METRIC, defined in Table 12. Exactly one METRIC TLV MUST be included in each RREQ message.
RREQ.route-metric	METRIC Message TLV value	Encoded as the value field of the METRIC TLV.
RREQ.hop-count	<msg-hop-count>	8 bits, hence MAX_HOP_COUNT is 255. MUST be included in a RREQ message.
RREQ.originator	<msg-orig-addr>	MUST be included in an RREQ message.
RREQ.destination	Address in Address-Block w/TLV	Encoded by way of an address in an address block, with which a Message-Type-specific Address Block TLV of type ADDR-TYPE and with Type-Extension DESTINATION is associated, defined in Table 9. A RREQ MUST contain exactly one address with a TLV of type ADDR-TYPE and with Type-Extension DESTINATION associated.

Table 2: RREQ Message Elements

A.2. RREP-Specific Message Encoding Considerations

This protocol defines, and hence owns, the RREP Message Type. Thus, as specified in [RFC5444], this protocol generates and transmits all RREP messages, receives all RREP messages and is responsible for determining whether and how each RREP message is to be processed (updating the Information Base) and/or forwarded, according to this specification. Table 3 describes how RREP messages are mapped into [RFC5444]-elements.

RREP Element	RFC5444-Element	Considerations
RREP.addr-length	<msg-addr-length>	Supports addresses from 1-16 octets
RREP.seq-num	<msg-seq-num>	16 bits, hence MAXVALUE (Section 8) is 65535. MUST be included
RREP.metric-type	METRIC Message TLV	Encoded by way of the Type-Extension of a Message-Type-specific Message TLV of type METRIC, defined in Table 12. Exactly one METRIC TLV MUST be included in each RREP message.
RREP.route-metric	METRIC Message TLV value	Encoded as the value field of the METRIC TLV.
RREP.ackrequired	ACKREQUIRED Message TLV	Encoded by way of a Message-Type-specific Message TLV of type ACKREQUIRED. If

		RREP.ackrequired is set, then a TLV of type ACKREQUIRED MUST be included in the RREP message. If RREP.ackrequired is cleared, then a TLV of type ACKREQUIRED MUST NOT be included in the RREP message.
RREP.hop-count	<msg-hop-count>	8 bits, hence MAX_HOP_COUNT is 255. MUST be included in a RREP message.

RREP.originator	<msg-orig-addr>	MUST be included in an RREP message.
RREP.destination	Address in Address-Block w/TLV	Encoded by way of an address in an address block, with which a Message-Type-specific Address Block TLV of type ADDR-TYPE and with Type-Extension DESTINATION is associated, defined in Table 14. A RREP MUST contain exactly one address with a TLV of type ADDR-TYPE and with Type-Extension DESTINATION associated.

Table 3: RREP Message Elements

A.3. RREP_ACK Message Encoding

This protocol defines, and hence owns, the RREP_ACK Message Type. Thus, as specified in [RFC5444], this protocol generates and transmits all RREP_ACK messages, receives all RREP_ACK messages and

is responsible for determining whether and how each RREP_ACK message is to be processed (updating the Information Base), according to this specification. Table 4 describes how RREP_ACK Messages are mapped into [RFC5444]-elements.

RREP_ACK Element	RFC5444-Element	Considerations
RREP_ACK.addr-length	<msg-addr-length>	Supports addresses from 1-16 octets
RREP_ACK.seq-num	<msg-seq-num>	16 bits, hence MAXVALUE (Section 8) is 65535. MUST be included
RREP_ACK.destination	Address in Address-Block w/TLV	Encoded by way of an address in an address block, with which a Message-Type-specific Address Block TLV of type ADDR-TYPE and with Type-Extension DESTINATION is associated, defined in Table 17. A RREP_ACK MUST contain exactly one address with a TLV of type ADDR-TYPE and with Type-Extension DESTINATION associated.

Table 4: RREP_ACK Message Elements

A.4. RERR Message Encoding

This protocol defines, and hence owns, the RERR Message Type. Thus, as specified in [RFC5444], this protocol generates and transmits all RERR messages, receives all RERR messages and is responsible for determining whether and how each RERR message is to be processed (updating the Information Base) and/or forwarded, according to this specification. Table 5 describes how RERR Messages are mapped into [RFC5444]-elements.

RERR Element	RFC5444-Element	Considerations
RERR.addr-length	<msg-addr-length>	Supports addresses from 1-16 octets
RERR.unreachableAddress	Address in Address-Block w/TLV	Encoded by way of an address in an address block, with which a Message-Type-specific Address Block TLV of type ADDR-TYPE and with Type-Extension ERRORCODE is associated, defined in Table 20.
RERR.errorcode	Address Block TLV Value	According to Section 18.1.

RERR.destination	Address in Address-Block w/TLV	Encoded by way of an address in an address block, with which a Message-Type-specific Address Block TLV of type ADDR-TYPE and with Type-Extension DESTINATION is associated, defined in Table 20. A RERR MUST contain exactly one address with a

		TLV of type ADDR-TYPE and with Type-Extension DESTINATION associated.

Table 5: RERR Message Elements

A.5. RFC5444-Specific IANA Considerations

This specification defines four Message Types, which must be allocated from the "Message Types" repository of [RFC5444], two Message TLV Types, which must be allocated from the "Message TLV Types" repository of [RFC5444], and four Address Block TLV Types, which must be allocated from the "Address Block TLV Types" repository of [RFC5444].

A.5.1. Expert Review: Evaluation Guidelines

For the registries where an Expert Review is required, the designated expert should take the same general recommendations into consideration as are specified by [RFC5444].

A.5.2. Message Types

This specification defines four Message Type, to be allocated from the 0-223 range of the "Message Types" namespace defined in [RFC5444], as specified in Table 6.

Type	Description
TBD1	RREQ: Route Request Message
TBD1	RREP: Route Reply Message

TBD1	RREP_ACK: Route Reply Acknowledgement Message
TBD1	RERR: Route Error Message

Table 6: Message Type assignment

A.6. RREQ Message-Type-Specific TLV Type Registries

IANA is requested to create a registry for Message-Type-specific Message TLVs for RREQ messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 7.

Type	Description	Allocation Policy
128	METRIC	Assigned
129-223	Unassigned	Expert Review

Table 7: RREQ Message-Type-specific Message TLV Types

Allocation of the METRIC TLV from the RREQ Message-Type-specific Message TLV Types in Table 7 will create a new Type Extension registry, with assignments as specified in Table 8.

Name	Type	Type Extension	Description	Allocation Policy
METRIC	128	0-255	Unassigned	Expert Review

Table 8: Message TLV Type assignment: METRIC

IANA is requested to create a registry for Message-Type-specific Address Block TLVs for RREQ messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 9.

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

Table 9: RREQ Message-Type-specific Address Block TLV Types

Allocation of the ADDR-TYPE TLV from the RREQ Message-Type-specific Address Block TLV Types in Table 9 will create a new Type Extension registry, with assignments as specified in Table 10.

Name	Type	Type Extension	Description	Allocation Policy
ADDR-TYPE	128	0	Destination	
ADDR-TYPE	128	2-255	Unassigned	Expert Review

Table 10: Address Block TLV Type assignment: ADDR-TYPE

A.7. RREP Message-Type-Specific TLV Type Registries

IANA is requested to create a registry for Message-Type-specific Message TLVs for RREP messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 11.

Type	Description	Allocation Policy
128	METRIC	Assigned
129	ACKREQUIRED	Assigned
130-223	Unassigned	Expert Review

Table 11: RREP Message-Type-specific Message TLV Types

Allocation of the METRIC TLV from the RREP Message-Type-specific Message TLV Types in Table 11 will create a new Type Extension registry, with assignments as specified in Table 12.

Name	Type	Type Extension	Description	Allocation Policy
METRIC	128	0-255	Unassigned	Expert Review

Table 12: Message TLV Type assignment: METRIC

Allocation of the ACKREQUIRED TLV from the RREP Message-Type-specific Message TLV Types in Table 11 will create a new Type Extension registry, with assignments as specified in Table 13.

Name	Type	Type Extension	Description	Allocation Policy
ACKREQUIRED	129	0-255	Unassigned	Expert Review

Table 13: Message TLV Type assignment: ACKREQUIRED

IANA is requested to create a registry for Message-Type-specific Address Block TLVs for RREP messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 14.

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

Table 14: RREP Message-Type-specific Address Block TLV Types

Allocation of the ADDR-TYPE TLV from the RREP Message-Type-specific Address Block TLV Types in Table 14 will create a new Type Extension registry, with assignments as specified in Table 15.

Name	Type	Type Extension	Description	Allocation Policy

ADDR-TYPE	128	0	Destination	
ADDR-TYPE	128	1-255	Unassigned	Expert Review

Table 15: Address Block TLV Type assignment: ADDR-TYPE

A.8. RREP_ACK Message-Type-Specific TLV Type Registries

IANA is requested to create a registry for Message-Type-specific Message TLVs for RREP_ACK messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 16.

Type	Description	Allocation Policy
128-223	Unassigned	Expert Review

Table 16: RREP_ACK Message-Type-specific Message TLV Types

IANA is requested to create a registry for Message-Type-specific Address Block TLVs for RREP_ACK messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 17.

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

Table 17: RREP_ACK Message-Type-specific Address Block TLV Types

Allocation of the ADDR-TYPE TLV from the RREP_ACK Message-Type-specific Address Block TLV Types in Table 17 will create a new Type Extension registry, with assignments as specified in Table 18.

Name	Type	Type Extension	Description	Allocation

				Policy
ADDR-TYPE	128	0	Destination	
ADDR-TYPE	128	2-255	Unassigned	Expert Review

Table 18: Address Block TLV Type assignment: ADDR-TYPE

A.9. RERR Message-Type-Specific TLV Type Registries

IANA is requested to create a registry for Message-Type-specific Message TLVs for RERR messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as

specified in Table 19.

Type	Description	Allocation Policy
128-223	Unassigned	Expert Review

Table 19: RERR Message-Type-specific Message TLV Types

IANA is requested to create a registry for Message-Type-specific Address Block TLVs for RERR messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 20.

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

Table 20: RREP_ACL Message-Type-specific Address Block TLV Types

Allocation of the ADDR-TYPE TLV from the RERR Message-Type-specific Address Block TLV Types in Table 20 will create a new Type Extension registry, with assignments as specified in Table 21.

Name	Type	Type Extension	Description	Allocation Policy
ADDR-TYPE	128	0	Destination	
ADDR-TYPE	128	1	ERRORCODE	
ADDR-TYPE	128	2-255	Unassigned	Expert Review

Table 21: Address Block TLV Type assignment: ADDR-TYPE

Appendix B. LOADng Control Packet Illustrations

This section presents example packets following this specification.

TO BE REDRAWN WHEN WE'VE FINISHED QUIBBLING OVER THE ENCODING

B.1. RREQ

B.2. RREP

B.3. RREP_ACK

B.4. RERR

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/

Charles E. Perkins
Futurewei Inc.

Phone: +1-408-421-1172
EMail: charliep@computer.org
URI: http://www.huawei.com/na/en/