

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: January 5, 2017

T. Clausen  
A. Colin de Verdiere  
J. Yi  
Ecole Polytechnique  
A. Niktash  
Maxim Integrated Products  
Y. Igarashi  
H. Satoh  
Hitachi, Ltd., Yokohama Research  
Laboratory  
U. Herberg  
  
C. Lavenu  
EDF R&D  
T. Lys  
ERDF  
J. Dean  
Naval Research Laboratory  
July 4, 2016

The Lightweight On-demand Ad hoc Distance-vector Routing Protocol - Next  
Generation (LOADng)  
draft-clausen-lln-loadng-15

#### Abstract

This document describes the Lightweight Ad hoc On-Demand - Next Generation (LOADng) distance vector routing protocol, a reactive routing protocol intended for use in Mobile Ad hoc NETWORKS (MANETs).

#### Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79. This document may not be modified, and derivative works of it may not be created, except to format it for publication as an RFC or to translate it into languages other than English.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on January 5, 2017.

#### Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

#### Table of Contents

1. Introduction . . . . .	5
2. Terminology and Notation . . . . .	6
2.1. Message and Message Field Notation . . . . .	6
2.2. Variable Notation . . . . .	7
2.3. Other Notation . . . . .	7
2.4. Terminology . . . . .	7
3. Applicability Statement . . . . .	8
4. Protocol Overview and Functioning . . . . .	9
4.1. Overview . . . . .	9
4.2. LOADng Routers and LOADng Interfaces . . . . .	11
4.3. Information Base Overview . . . . .	11
4.4. Signaling Overview . . . . .	12
5. Protocol Parameters . . . . .	13
5.1. Protocol and Port Numbers . . . . .	13
5.2. Router Parameters . . . . .	13
5.3. Interface Parameters . . . . .	14
5.4. Constants . . . . .	15
6. Protocol Message Content . . . . .	15
6.1. Route Request (RREQ) Messages . . . . .	15
6.2. Route Reply (RREP) Messages . . . . .	16
6.3. Route Reply Acknowledgement (RREP_ACK) Messages . . . . .	17
6.4. Route Error (RERR) Messages . . . . .	18
7. Information Base . . . . .	19
7.1. Routing Set . . . . .	19
7.2. Local Interface Set . . . . .	20
7.3. Blacklisted Neighbor Set . . . . .	20
7.4. Destination Address Set . . . . .	21
7.5. Pending Acknowledgment Set . . . . .	21
8. LOADng Router Sequence Numbers . . . . .	22

9. Route Maintenance . . . . .	22
10. Unidirectional Link Handling . . . . .	24
10.1. Blacklist Usage . . . . .	24
11. Common Rules for RREQ and RREP Messages . . . . .	25
11.1. Identifying Invalid RREQ or RREP Messages . . . . .	26
11.2. RREQ and RREP Message Processing . . . . .	27
12. Route Requests (RREQs) . . . . .	30
12.1. RREQ Generation . . . . .	30
12.2. RREQ Processing . . . . .	31
12.3. RREQ Forwarding . . . . .	31
12.4. RREQ Transmission . . . . .	32
13. Route Replies (RREPs) . . . . .	32
13.1. RREP Generation . . . . .	32
13.2. RREP Processing . . . . .	33
13.3. RREP Forwarding . . . . .	34
13.4. RREP Transmission . . . . .	34
14. Route Errors (RERRs) . . . . .	35
14.1. Identifying Invalid RERR Messages . . . . .	36
14.2. RERR Generation . . . . .	36
14.3. RERR Processing . . . . .	37
14.4. RERR Forwarding . . . . .	38
14.5. RERR Transmission . . . . .	38
15. Route Reply Acknowledgments (RREP_ACKs) . . . . .	39
15.1. Identifying Invalid RREP_ACK Messages . . . . .	39
15.2. RREP_ACK Generation . . . . .	39
15.3. RREP_ACK Processing . . . . .	40
15.4. RREP_ACK Forwarding . . . . .	41
15.5. RREP_ACK Transmission . . . . .	41
16. Metrics . . . . .	41
16.1. Specifying New Metrics . . . . .	41
17. Implementation Status . . . . .	41
17.1. Implementation of Ecole Polytechnique . . . . .	42
17.2. Implementation of Fujitsu Laboratories of America . . . . .	42
17.3. Implementation of Hitachi Yokohama Research Laboratory - 1 . . . . .	43
17.4. Implementation of Hitachi Yokohama Research Laboratory - 2 . . . . .	43
18. Security Considerations . . . . .	43
18.1. Security Threats . . . . .	44
18.1.1. Confidentiality . . . . .	44
18.1.2. Integrity . . . . .	45
18.1.3. Channel Jamming and State Explosion . . . . .	46
18.1.4. Interaction with External Routing Domains . . . . .	47
18.2. Integrity Protection . . . . .	47
18.2.1. Overview . . . . .	48
18.2.2. Message Generation and Processing . . . . .	50
19. LOADng Specific IANA Considerations . . . . .	52
19.1. Error Codes . . . . .	52

20. Contributors . . . . .	53
21. Acknowledgments . . . . .	53
22. References . . . . .	54
22.1. Normative References . . . . .	54
22.2. Informative References . . . . .	54
Appendix A. Gateway Considerations . . . . .	56
Appendix B. LOADng Control Messages using RFC5444 . . . . .	56
B.1. RREQ-Specific Message Encoding Considerations . . . . .	56
B.2. RREP-Specific Message Encoding Considerations . . . . .	58
B.3. RREP_ACK Message Encoding . . . . .	60
B.4. RERR Message Encoding . . . . .	61
B.5. RFC5444-Specific IANA Considerations . . . . .	62
B.5.1. Expert Review: Evaluation Guidelines . . . . .	62
B.5.2. Message Types . . . . .	63
B.6. RREQ Message-Type-Specific TLV Type Registries . . . . .	63
B.7. RREP Message-Type-Specific TLV Type Registries . . . . .	64
B.8. RREP_ACK Message-Type-Specific TLV Type Registries . . . . .	67
B.9. RERR Message-Type-Specific TLV Type Registries . . . . .	67
Appendix C. LOADng Control Packet Illustrations . . . . .	68
C.1. RREQ . . . . .	68
C.2. RREP . . . . .	70
C.3. RREP_ACK . . . . .	71
C.4. RERR . . . . .	72

## 1. Introduction

The Lightweight On-demand Ad hoc Distance-vector Routing Protocol - Next Generation (LOADng) is a routing protocol, derived from AODV [RFC3561] and extended for use in Mobile Ad hoc NETWORKS (MANETs). 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 to be broken, e.g., 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 to inform the originator about the route breakage.

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 RREQ/RREP messages 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 addresses [EUI64], 6 octet MAC addresses and 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.
- o 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., originator).

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 B, and Appendix C illustrates the bit layout 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 a field in the message `MsgType`, then `MsgType.field` is also used to represent the value of that field.

`bar` - A variable (not prepended by `MsgType`), 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.

Network Address - A layer 3 IP address plus an associated prefix length. This may be an address with an associated maximum prefix length or an address prefix including a prefix length. A network address thus represents a range of IP addresses.

### 3. Applicability Statement

LOADng is a reactive MANET protocol, i.e., routes are discovered only when a data packet is sent by a router (e.g., on behalf of an attached host), and when the router has no route for this destination. In that case, the router floods Route Requests (RREQ) throughout the network for discovering the destination. Reactive protocols require state only for the routes currently in use, contrary to proactive protocols, which periodically send control traffic and store routes to all destinations in the network. As MANETs are often operated on wireless channels, flooding RREQs may lead to frame collisions and therefore data loss. Moreover, each transmission on a network interface consumes energy, reducing the life-time of battery-driven routers. Consequently, in order to reduce the amount of control traffic, LOADng (and in general reactive protocols) are most suitable under the following constraints:

- o Few concurrent traffic flows in the network (i.e., traffic flows only between few sources and destinations);
- o Little data traffic overall, and therefore the traffic load from periodic signaling (for proactive protocols) is greater than the traffic load from flooding RREQs (for reactive protocols);
- o State requirements on the router are very stringent, i.e., it is beneficial to store only few routes on a router.

In these specific use cases, reactive MANET protocols have shown to be beneficial, and may be preferable over the more general use case of proactive MANET protocols.

Specifically, the applicability of LOADng is determined by its characteristics, which are that this protocol:

- o Is a reactive routing protocol for Mobile Ad hoc NETWORKS (MANETs).
- o Is designed to work in networks with dynamic topology in which the links may be lossy due to collisions, channel instability, or movement of routers.
- o Supports the use of optimized flooding for RREQs.



- o 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.
- o Supports addresses of any length with integral number of octets, from 16 octets to a single octet.
- o 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.
- o 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:

- o Discover a bi-directional route to any destination in the network.
- o Establish a route only when there is data traffic to be sent along that route.
- o Maintain a route only for as long as there is data traffic being sent along that route.
- o 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:

- o When having a data packet to deliver to a destination, for which no tuple in the routing set 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.
- o 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).

- o 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.
- o When an RREQ is considered a candidate for forwarding, retransmit it according to the flooding operation, specified for the network.
- o 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).
- o 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.
- o Has a number of interface parameters.

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 metric 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 Interface addresses of a LOADng Router 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 (destination) address 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 (originator) address for which a Reverse Route is to be installed 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 (originator) address to which a Forward Route is to be installed when forwarding the RREP.

- \* The (destination) address towards which the RREP is to be sent. More precisely, the destination address determines 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 and constants 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 RREQ message is expected to take when traversing from one end of the network to the other, with the consideration of RREQ\_MAX\_JITTER.

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\_MIN\_INTERVAL - is the minimal interval (in milliseconds) of RREQs that a particular LOADng Router is allowed to send.

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 representing the maximum possible metric (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 during that time (B\_HOLD\_TIME). B\_HOLD\_TIME should be greater than  $2 \times \text{NET\_TRAVERSAL\_TIME} \times \text{RREQ\_RETRIES}$ , to ensure that subsequent RREQs will reach the destination via a route, excluding the link to the blacklisted neighbor.

MAX\_HOP\_LIMIT - is the maximum limit of the number of hops that LOADng routing messages are allowed to traverse.

### 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.

RREQ\_MAX\_JITTER - is the default value of MAXJITTER used in [RFC5148] for RREQ messages forwarded by this LOADng Router on this interface.

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 for this interface.

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

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.

#### 5.4. Constants

MAX\_HOP\_COUNT - is the maximum number of hops as representable by the encoding that is used (e.g., 255 when using [RFC5444]). It SHOULD NOT be used to limit the scope of a message; the router parameter MAX\_HOP\_LIMIT can be used to limit the scope of a LOADng routing message.

### 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 B, and Appendix C 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 that 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.hop-limit is an unsigned integer field and specifies the number of hops that the message is allowed to traverse.

RREQ.originator is an identifier of RREQ.addr-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, RREQ.originator 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 RREQ.addr-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, RREQ.hop-limit, 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 RREP.originator to RREP.destination.

RREP.hop-limit is an unsigned integer field and specifies the number of hops that the message is allowed to traverse.

RREP.originator is an identifier of RREP.addr-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, RREP.originator 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 RREP.addr-length + 1 octets, specifying the address to which the RREP should be sent. (I.e., this address is equivalent to RREQ.originator 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, RREP.hop-limit, 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 RREP\_ACK.addr-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.originator is an identifier of RERR.addr-length + 1 octets, specifying the address of the LOADng Interface over which this RERR was generated by a LOADng Router.

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.

RERR.hop-limit is an unsigned integer field and specifies the number of hops that the message is allowed to traverse.

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, RERR.originator and RERR.destination.

The following fields of an RERR message are mutable, i.e., they will be changed by intermediate routers during processing or forwarding, as specified in Section 14.3 and Section 14.4: RERR.hop-limit.

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\_bidirectional, R\_local\_iface\_addr, R\_valid\_time)

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\_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 of the interface with R\_local\_iface\_address.

R\_local\_iface\_addr - is an address of the local LOADng Interface, through which the destination can be reached.

R\_valid\_time - specifies the time until which the information recorded in this Routing Tuple is considered valid.

## 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\_received, 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\_received - is a boolean flag, which specifies the tuple has been acknowledged by a corresponding RREP\_ACK message. The default value is FALSE.

P\_ack\_timeout - is the time after which the tuple MUST be expired.

## 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 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) can 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:

$$S2 < S1 \text{ AND } S1 - S2 \leq \text{MAXVALUE}/2 \text{ OR}$$
$$S1 < S2 \text{ AND } S2 - S1 > \text{MAXVALUE}/2$$

## 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 needs 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:

- o 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.
- o 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 a LOADng Router fails to deliver a data packet to a next-hop, it MUST generate an RERR message, as specified in 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):

- o 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).
- o If a mechanism such as NDP [RFC4861] is available, the LOADng Router MAY use it.
- o A LOADng Router MAY use a neighborhood discovery mechanism with bidirectionality verification, such as NHDP [RFC6130].
- o RREP\_ACK message exchange, as described in Section 15.
- o 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 an interface of 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:

- o B\_neighbor\_address := the address of the blacklisted neighbor LOADng Router interface



- o `B_valid_time := current_time + B_HOLD_TIME`

When a neighbor LOADng Router interface 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 LOADng Router interface 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 an 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.

hop-count is a variable, representing the hop-count, as included in the received RREQ or RREP message.

hop-limit is a variable, representing the hop-limit, as included in the received RREQ or RREP message.

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 MSG.originator 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 mechanism as specified in Section 18.2 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/updated according to their specification.
2. Set the variable hop-count to MSG.hop-count + 1.
3. Set the variable hop-limit to MSG.hop-limit - 1.
4. If MSG.metric-type is known to this LOADng Router, and if MSG.metric-type is not HOP\_COUNT, 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.
5. Otherwise:
  - \* Set the variable used-metric-type to HOP\_COUNT.
  - \* Set the variable route-metric to MAX\_DIST, see Section 16.
  - \* Set the variable link-metric to MAX\_DIST.
6. Find the Routing Tuple (henceforth, Matching Routing Tuple) where:
  - \* R\_dest\_addr = MSG.originator
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 := MSG.originator

- \* R\_next\_addr := previous-hop
  - \* R\_metric\_type := used-metric-type
  - \* R\_metric := MAX\_DIST
  - \* R\_hop\_count := 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.
8. 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\_type = used-metric-type; AND
    - + R\_metric > route-metricOR
    - + R\_seq\_num = MSG.seq-num; AND
    - + R\_metric\_type = used-metric-type; AND
    - + R\_metric = route-metric; AND
    - + R\_hop\_count > hop-countOR
    - + R\_seq\_num = MSG.seq-num; AND
    - + R\_metric\_type does not equal to used-metric-type; AND
    - + R\_metric\_type = HOP\_COUNTOR

+ R\_seq\_num < MSG.seq-num

Then:

1. The message is used for updating the Routing Set. The Routing Tuple, where:
  - R\_dest\_addr = MSG.originator;is updated thus:
  - R\_next\_addr := previous-hop
  - R\_metric\_type = used-metric-type
  - R\_metric := route-metric
  - R\_hop\_count := 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.
2. 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
3. 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, if the message is an RREQ, it is not processed further and is not considered for forwarding. If it is an RREP and if RREP.ackrequired is set, an RREP\_ACK message MUST be sent to the previous-hop, according to Section 15.2. The RREP 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. Furthermore, if there is a matching tuple in the Routing Set with the R\_bidirectional set to FALSE, and the parameter USE\_BIDIRECTIONAL\_LINK\_ONLY of the interface with R\_local\_iface\_address equals TRUE, an RREQ MUST be generated.

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. Two consequent RREQs generated on an interface of a LOADng Router SHOULD be separated at least RREQ\_MIN\_INTERVAL.

### 12.1. RREQ Generation

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

- o 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.route-metric := 0.
- o RREQ.seq-num set to the next unused sequence number, maintained by this LOADng Router;
- o RREQ.hop-count := 0;

- o RREQ.hop-limit := MAX\_HOP\_LIMIT;
- 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;

## 12.2. RREQ Processing

The variables hop-count and hop-limit have been updated in Section 11.2 (when processing the message) and are used in this section. 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.destination is listed in I\_local\_iface\_addr\_list of any Local Interface Tuple, or corresponds to D\_address of any Destination Address Tuple of this LOADng Router, the RREP generation process in Section 13.1 MUST be applied. The RREQ is not considered for forwarding.
4. Otherwise, if hop-count is less than MAX\_HOP\_COUNT and hop-limit is greater than 0, the message is considered for forwarding according to Section 12.3.

## 12.3. RREQ Forwarding

The variables used-metric type, hop-count, hop-limit and route-metric have been updated in Section 11.2 (when processing the message) and are used in this section to update the content of the message to be forwarded. 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.route-metric := route-metric (as set in Section 11.2)
3. RREQ.hop-count := hop-count (as set in Section 11.2)

#### 4. RREQ.hop-limit := hop-limit (as set in Section 11.2)

An RREQ MUST 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, whether initially generated or forwarded, are sent to all neighbor LOADng Routers through all interfaces in the Local Interface Set.

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], using RREQ\_MAX\_JITTER, in order to avoid such losses.

#### 13. Route Replies (RREPs)

Route Replies (RREPs) are generated by a LOADng Router in response to an RREQ (henceforth denoted "corresponding RREQ"), and are sent by the LOADng Router which has, in either its Destination Address Set or in its Local Interface Set, the address from RREQ.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 (RREQ.originator, RREQ.seqnum). 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 (RREQ.originator, RREQ.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 if the metric type is known to the router. Otherwise, RREP.metric-type is set to HOP\_COUNT;
- o RREP.route-metric := 0
- o RREP.hop-count := 0;
- o RREP.hop-limit := MAX\_HOP\_LIMIT;
- 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.

The RREP that is generated is transmitted according to Section 13.4.

### 13.2. RREP Processing

The variables hop-count and hop-limit have been updated in Section 11.2 (when processing the message) and are used in this section. 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 15.2.
4. If hop-count is equal to MAX\_HOP\_COUNT or hop-limit is equal to 0, the message is not considered for forwarding.
5. Otherwise, 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

The variables used-metric type, hop-count, hop-limit and route-metric have been updated in Section 11.2 (when processing the message) and are used in this section to update the content of the message to be forwarded. 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.route-metric := route-metric (as set in Section 11.2)
3. RREP.hop-count := hop-count (as set in Section 11.2)
4. RREP.hop-limit := hop-limit (as set in Section 11.2)
5. The RREP is transmitted, according to Section 13.4.

The RREP message is then unicast to the next hop towards RREP.destination.

### 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:

- o Install proper forward routes; AND
- o 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
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\_received := FALSE
    - + P\_ack\_timeout := current\_time + RREP\_ACK\_TIMEOUT
  - \* RREP.ackrequired := TRUE
4. Otherwise:
  - \* RREP.ackrequired := 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.

When a Pending Acknowledgement Tuple expires, if P\_ack\_received = FALSE, the P\_next\_hop address MUST be blacklisted by creating a Blacklisted Neighbor Tuple according to Section 7.3

#### 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 belongs to 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:

- o "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 received RERR 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., RERR.addr-length + 1) differs from the length of the address(es) of this LOADng Router.
- o The address contained in RERR.originator is an address of this LOADng Router.

A LOADng Router MAY recognize additional reasons, external to this specification, for identifying that an RERR message is invalid for processing, e.g., to allow a security mechanism as specified in Section 18.2 to perform verification of integrity check values 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:

- o RERR.error-code := the error code corresponding to the event causing the RERR to be generated, from among those recorded in Table 1;
- o RERR.addr-length := the length of the address, as specified in Section 6;
- o RERR.unreachableAddress := the destination address from the unsuccessfully delivered data packet.
- o RERR.originator := one address of the LOADng Interface of the LOADng Router that generates the RERR.
- o RERR.destination := the source address from the unsuccessfully delivered data packet, towards which the RERR is to be sent.

- o `RERR.hop-limit := MAX_HOP_LIMIT;`

#### 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.

`hop-limit` is a variable, representing the hop-limit, as included in the received RERR message.

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

1. If the RERR is invalid for processing, as defined in Section 14.1, the RERR MUST be discarded without further processing. The message is not considered for forwarding.
2. Included TLVs are processed/updated according to their specification.
3. Set the variable `hop-limit` to `RERR.hop-limit - 1`.
4. Find the Routing Tuple (henceforth "matching Routing Tuple") in the Routing Set where:
  - \* `R_dest_addr = RERR.unreachableAddress`
  - \* `R_next_addr = previous-hop`
5. If no matching Routing Tuple is found, the RERR is not processed further, but is considered for forwarding, as specified in Section 14.4.
6. Otherwise, if one matching Routing Tuple is found:
  1. If `RERR.errorcode` is 0 ("No available route", as specified in Section 19.1), this matching Routing Tuple is updated as follows:
    - + `R_valid_time := EXPIRED`

Extensions to this specification MAY define additional error codes in the Error Code IANA registry, and MAY insert processing rules here for RERRs with that error code.

2. If hop-limit is greater than 0, the RERR message is considered for forwarding, as specified in Section 14.4

#### 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. RERR.hop-limit := hop-limit (as set in Section 14.3)
2. Find the Destination Address Tuple (henceforth, matching Destination Address Tuple) in the Destination Address Set where:
  - \* D\_address = RERR.destination
3. 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.
4. 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.
5. If a matching Local Interface Tuple is found, the RERR message is discarded and not retransmitted, as it has reached the final destination.
6. 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 `current_time + RREP_ACK_TIMEOUT`, as described in Section 13.4.

The following definition is used in this section:

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

##### 15.1. Identifying Invalid RREP\_ACK Messages

A received RREP\_ACK 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., `RREP_ACK.addr-length + 1`) differs from the length of the address(es) of this LOADng Router.

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

##### 15.2. 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:

- o RREP\_ACK.addr-length := the length of the address, as specified in Section 6;
- o RREP\_ACK.seq-num := the value of the RREP.seq-num field of the received RREP;
- o RREP\_ACK.destination := RREP.originator of the received RREP.

### 15.3. RREP\_ACK Processing

On receiving an RREP\_ACK from a LOADng neighbor LOADng Router, a LOADng Router MUST do the following:

1. If the RREP\_ACK is invalid for processing, as defined in Section 15.1, the RREP\_ACK MUST be discarded without further processing.

2. 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

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

- \* P\_next\_hop is the address of the 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 acknowledged. The Matching Pending Acknowledgement Tuple is updated as follows:

- \* P\_ack\_received := TRUE

- \* P\_ack\_timeout := EXPIRED



#### 15.4. RREP\_ACK Forwarding

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

#### 15.5. 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. Implementation Status

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and based on a proposal described in [RFC6982]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC6982], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

In the following subsections, each publicly-known implementation of LOADng is listed. There are currently four publicly-known implementations of LOADng. These have been tested for interoperability in at least three interop events, as described in [I-D.loadng-interop-report].

#### 17.1. Implementation of Ecole Polytechnique

This implementation is developed by the Networking Group at Ecole Polytechnique. It can run over real network interfaces, and can also be integrated with the network simulator NS2. It is a Java implementation, and can be used on any platform that includes a Java virtual machine.

The implementation has been maintained since the 00 revision of LOADng, and is quite mature. It has been tested in interoperability events with other implementations (as described in [I-D.loadng-interop-report]), and in large-scale network simulations with up to 1000 routers. There have been several scientific publications based on this implementation, such as [IEEE\_VTC2012] [IEEE\_WiCom2012] [IEEE\_ICWITS2012].

All the protocol functions of this revision (-08) of the specification, including RREQ/RREP/RREP-ACK/RERR generation, processing, forwarding and transmission, as well as blacklisting, are implemented.

The latest implementation conforms to the LOADng-07 revision as documented in this specification. This software is currently closed source.

#### 17.2. Implementation of Fujitsu Laboratories of America

This implementation is developed by Fujitsu Laboratories of America. It is a Java implementation, structured in multiple separate modules, notably a [RFC5444] generator and parser, and integration module in the network simulator Ns-2, a kernel module for integrating the implementation in a Linux kernel (not yet completed), and the protocol core.

The implementation is mature and has been tested both in interoperability tests with other implementations [I-D.loadng-interop-report], as well as large-scale simulations with hundreds of routers. The implementation is not currently used in deployments. The implementation supports all LOADng functions (RREQ, RREP, RREP-ACK generation, processing, forwarding and transmission), and conforms to the LOADng-06 specification. The software is currently closed source.

### 17.3. Implementation of Hitachi Yokohama Research Laboratory - 1

This implementation is developed by Hitachi, Ltd. Yokohama Research Laboratory. It can run over real embedded devices. It is a C implementation. The implementation is maintained since the 00 revision of LOADng. It was tested in the first interoperability event with other implementations, as described in [I-D.loadng-interop-report].

This implementation is alpha version, mainly for performance test and evaluations. All the functions of the protocol, including RREQ/RREP/RREP-ACK/RERR generation, processing, forwarding and transmission, blacklisting, have been implemented. Also a RFC5444 generator and parser have been implemented. The latest implementation conforms to LOADng-06 revision. This software is currently closed source.

### 17.4. Implementation of Hitachi Yokohama Research Laboratory -2

This implementation is developed by Hitachi, Ltd. Yokohama Research Laboratory. It can run over real network interface, and can also be integrated with network simulator NS2. It is a C++ implementation.

The implementation is mature and maintained since the 00 revision of LOADng. It was tested in large-scale network simulations up to 500 routers.

All the functions of the protocol, including RREQ/RREP/RREP-ACK/RERR generation, processing, forwarding and transmission, blacklisting, have been implemented. The latest implementation conforms to the LOADng-05 revision. This software is currently closed source.

## 18. Security Considerations

This section analyzes security threats of LOADng, and specifies mandatory-to-implement security mechanisms of LOADng for integrity and replay protection. A deployment of LOADng protocol may choose to employ an alternative(s) to these mechanisms. For example, it may choose to use an alternative Integrity Check Value (ICV) with preferred properties, and/or it may use an alternative timestamp. A deployment may choose to use no such security mechanisms, but this is not recommended.

Section 18.1 illustrates the security threats of the protocol assuming if no security measure is applied. Section 18.2 specifies how to use Integrity Check Value (ICV) and timestamps to protect the protocol messages, and how the security mechanisms can alleviate the threats.

### 18.1. Security Threats

As a reactive routing protocol, this protocol is a potential target for various attacks. Various possible vulnerabilities are discussed in this section. For each kind of threats, the vulnerabilities of the protocol is firstly analyzed, with the assumption that no security measure is applied. Then how the integrity protection specified in Section 18.2 can alleviate the threats are discussed, with the analyses of limitations.

#### 18.1.1. 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.

This protocol does not specify mechanism to protect the confidentiality of network topology. Unless the information about the network topology itself is confidential, integrity of control messages is sufficient to admit only trusted routers (i.e., routers with valid credentials) to the network.

In situations where the confidentiality of the network topology is of importance, regular cryptographic techniques, can be applied to ensure that control traffic can be read and interpreted by only those authorized to do so.

#### 18.1.2. 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, are able to inject invalid control traffic, network integrity may be compromised. Therefore, message authentication is recommended.

Different such situations may occur if not integrity protection mechanism is applied, 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.

#### 18.1.3. 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 the receipt of an RREQ corresponding to a destination owned by the LOADng Router. A router that forwards an RREQ records the reverse route state. A router that forwards an RREP records the forward route state. 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).

#### 18.1.4. 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".

#### 18.2. Integrity Protection

The mechanisms specified are the use of an ICV for protection of the protocols' control messages and the use of timestamps in those messages to prevent replay attacks. Both use the TLV mechanism specified in [RFC5444] to add this information to the messages. These ICV and **TIMESTAMP** TLVs are defined in [RFC7182].

## 18.2.1. Overview

When an RREQ/RREP/RREP\_ACK/RERR message is being processed, LOADng specifies that it MAY recognize additional reasons for identifying invalid messages (Section 11.1 and Section 14.1 ). This section specifies a mechanism that provides this functionality. Implementations of this protocol MUST include this mechanism, and deployments of LOADng SHOULD use this mechanism, except when a different security mechanism is more appropriate.

The integrity protection mechanism specified in this section is placed in the control packet/message processing flow as indicated in Figure 1. It exists between the control packet parsing/generation function of [RFC5444] and the message processing/generation function of LOADng.

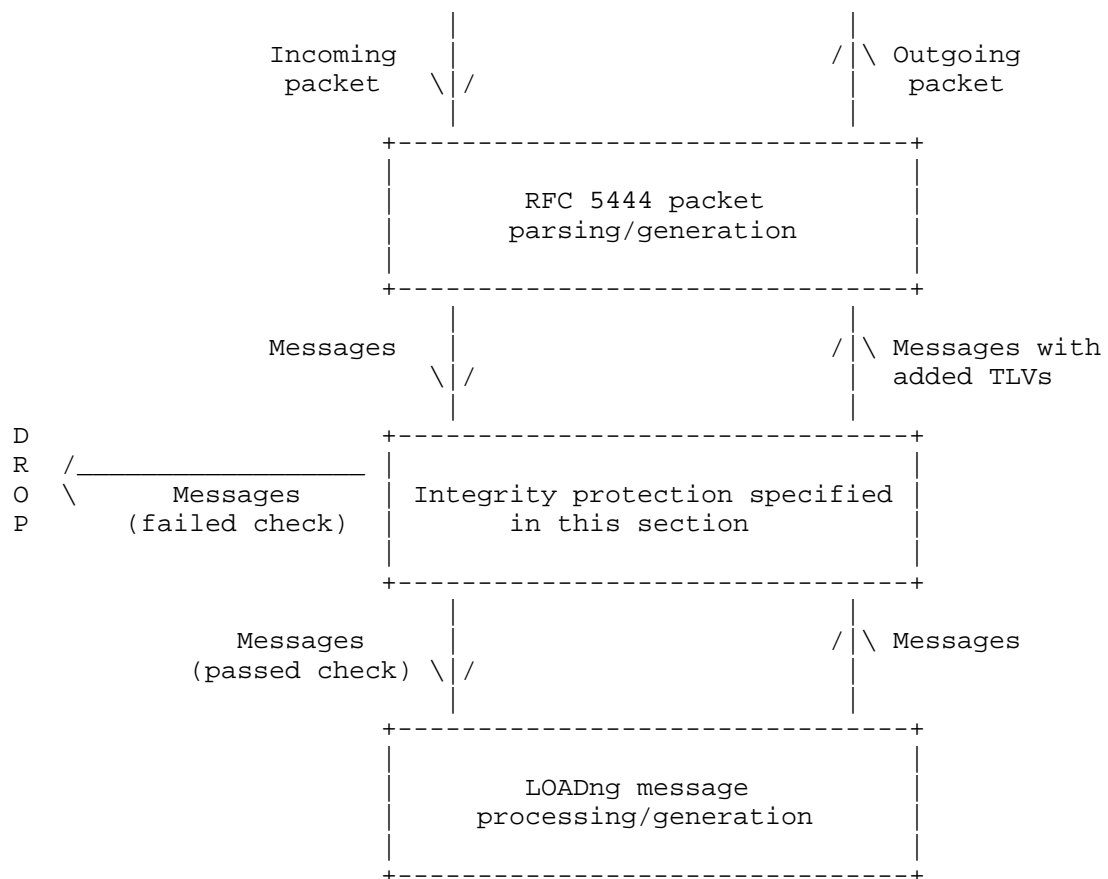


Figure 1: Relationship with RFC5444 and LOADng



The integrity projection mechanism:

- o Specifies an association of ICVs with protocol messages, and specifies how to use a missing or invalid ICV as a reason to reject a message as invalid for processing.
- o Specifies the implementation of an ICV Message TLV, defined in [RFC7182], using a SHA-256-based Hashed Message Authentication Code (HMAC) applied to the appropriate message contents. Implementations of this protocol MUST support an HMAC-SHA-256 ICV TLV, and deployments SHOULD use it except when use of a different algorithm is more appropriate. An implementation MAY use more than one ICV TLV in a message, as long as they each use a different algorithm or key to calculate the ICV.
- o Specifies the implementation of a TIMESTAMP Message TLV, defined in [RFC7182], to provide message replay protection. Implementations of LOADng using this mechanism MUST support a timestamp based on POSIX time, and deployments SHOULD use it if the clocks in all routers in the network can be synchronized with sufficient precision.
- o Assumes that a router that is able to generate correct integrity check values is considered trusted.

This mechanism does not:

- o Specify which key identifiers are to be used in a MANET in which the routers share more than one secret key. (Such keys will be differentiated using the <key-id> field defined in an ICV TLV in [RFC7182].)
- o Specify how to distribute cryptographic material (shared secret key(s)).
- o Specify how to detect compromised routers with valid keys.
- o Specify how to handle (revoke) compromised routers with valid keys.

No key management mechanism is specified in this scenario because given the various application scenarios of LOADng, it is hard to identify a basic mechanism that fits for all. Depending on the applications, either automated key management or manual key management [RFC4107] can be used. For example, in a controlled industrial environments but with very limited data rate and high delay, a manual key management is probably preferred. In a home applications, simple pairing mechanisms for key exchange can be

applied. However, in a malicious environments, such as battlefield, a much more sophisticated key management is desired, by taking consideration of compromised routers, etc.

#### 18.2.2. Message Generation and Processing

##### 18.2.2.1. Message Content

Messages MUST have the content specified in Section 6. In addition, messages that conform to the integration protection mechanism MUST contain:

- o At least one ICV Message TLV (as specified in [RFC7182]), generated according to Section 18.2.2.2. Implementations of LOADng MUST support the following version of the ICV TLV, but other versions MAY be used instead, or in addition, in a deployment, if more appropriate:
  - \* For RREQ/RREP/RREP\_ACK/RERR messages, type-extension := 1
  - \* hash-function := 3 (SHA-256)
  - \* cryptographic-function := 3 (HMAC)
- o At least one TIMESTAMP Message TLV (as specified in [RFC7182]), generated according to Section 18.2.2.2. Implementations of LOADng using this mechanism MUST support the following version of the TIMESTAMP TLV, but other versions MAY be used instead, or in addition, in a deployment, if more appropriate:
  - \* type-extension := 1

##### 18.2.2.2. Message Generation

For each RREQ/RREP/RREP\_ACK/RERR message, after message generation and before message transmission, the additional TLVs specified in Section 18.2.2.1 MUST (unless already present) be added to the outgoing message when using this mechanism.

The following processing steps (when using a single timestamp version and a single ICV algorithm) MUST be performed for a cryptographic algorithm that is used for generating an ICV for a message:

1. All ICV TLVs (if any) are temporarily removed from the message. Any temporarily removed ICV TLVs MUST be stored, in order to be reinserted into the message in step 5. The message size and Message TLV Block size are updated accordingly.

2. All the mutable fields of the message (specified in Section 6) are temporarily set to 0.
3. A TLV of type `TIMESTAMP`, as specified in Section 18.2.2.1, is added to the Message TLV Block. The message size and Message TLV Block size are updated accordingly.
4. A TLV of type `ICV`, as specified in Section 18.2.2.1, is added to the Message TLV Block. The message size and Message TLV Block size are updated accordingly.
5. All `ICV` TLVs that were temporarily removed in step 1, are restored. The message size and Message TLV Block size are updated accordingly.
6. All mutable fields that are temporarily set to 0 are restored to their previous values.

An implementation MAY add either alternative `TIMESTAMP` and/or `ICV` TLVs or more than one `TIMESTAMP` and/or `ICV` TLVs. All `TIMESTAMP` TLVs MUST be inserted before adding `ICV` TLVs.

#### 18.2.2.3. Message Processing

LOADng gives a number of conditions that will lead to a rejection of the message as "invalid". When using a single timestamp version, and a single `ICV` algorithm, add the following conditions to that list, each of which, if true, MUST cause LOADng to consider the message as invalid for processing when using this integrity protection mechanism:

1. The Message TLV Block of the message does not contain exactly one `TIMESTAMP` TLV of the selected version. This version specification includes the type extension. (The Message TLV Block may also contain `TIMESTAMP` TLVs of other versions.)
2. The Message TLV Block does not contain exactly one `ICV` TLV using the selected algorithm and key identifier. This algorithm specification includes the type extension, and for type extensions 1, the hash function and cryptographic function. (The Message TLV Block may also contain `ICV` TLVs using other algorithms and key identifiers.)
3. Validation of the identified (in step 1) `TIMESTAMP` TLV in the Message TLV Block of the message fails, as according to the timestamp validation:

1. If the current POSIX time minus the value of that `TIMESTAMP TLV` is greater than `NET_TRAVERSAL_TIME`, then the message validation fails.
2. Otherwise, the message validation succeeds.
4. Validation of the identified (in step 2) `ICV TLV` in the Message TLV Block of the message fails, as according to the `ICV` validation:
  1. All `ICV Message TLVs` (including the identified `ICV Message TLV`) are temporarily removed from the message, and the message size and Message TLV Block size are updated accordingly.
  2. All the mutable fields of the message (specified in Section 6) are temporarily set to 0.
  3. Calculate the `ICV` for the parameters specified in the identified `ICV Message TLV`, as specified in [RFC7182].
  4. If this `ICV` differs from the value of `<ICV-data>` in the `ICV Message TLV`, then the message validation fails. If the `<ICV-data>` has been truncated (as specified in [RFC7182]), the `ICV` calculated in the previous step MUST be truncated to the TLV length of the `ICV Message TLV` before comparing it with the `<ICV-data>`.
  5. Otherwise, the message validation succeeds. The message's mutable fields are restored to their previous value, and the `ICV Message TLVs` are returned to the message, whose size is updated accordingly.

## 19. LOADng Specific IANA Considerations

### 19.1. 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

## 20. 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>

## 21. 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.

## 22. References

## 22.1. 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.
- [RFC7182] Herberg, U., Clausen, T., and C. Dearlove, "Integrity Check Value and Timestamp TLV Definitions for Mobile Ad Hoc Networks (MANETs)", RFC 7182, April 2014.
- [RFC5498] Chakeres, I., "IANA Allocations for Mobile Ad Hoc Network (MANET) Protocols", RFC 5498, March 2009.

## 22.2. Informative References

- [RFC6982] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", RFC 6982, July 2013.
- [I-D.loadng-interop-report] Clausen, T., Camacho, A., Yi, J., Colin de Verdiere, A., Igarashi, Y., Satoh, H., Morii, Y., Heropberg, U., and C. Lavenu, "Interoperability Report for the Lightweight On-demand Ad hoc Distance-vector Routing Protocol - Next Generation (LOADng)", draft-lavenu-lln-loadng-interopability-report-04 (work in progress), December 2012.
- [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.
- [RFC6130] Clausen, T., Dean, J., and C. Dearlove, "MANET Neighborhood Discovery Protocol (NHDP)", RFC 6130, April 2011.
- [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.
- [RFC4107] Bellovin, S. and R. Housley, "Guidelines for Cryptographic Key Management", BCP 107, RFC 4107, June 2005.
- [EUI64] IEEE, "Guidelines for 64-bit Global Identifier (EUI-64) Registration Authority".
- [IEEE754-2008] IEEE, "IEEE 754-2008: IEEE Standard for Floating-Point Arithmetic", 2008.
- [IEEE\_VTC2012] Clausen, T., Yi, J., and A. Coline de Verdiere, "Towards AODV Version 2", Proceedings of IEEE VTC 2012 Fall, IEEE 76th Vehicular Technology Conference, 2012.
- [IEEE\_WiCom2012] Yi, J., Clausen, T., and A. Coline de Verdiere, "Efficient Data Acquisition in Sensor Networks: Introducing (the) LOADng Collection Tree Protocol", Proceedings of IEEE WiCom 2012, The 8th IEEE International Conference on Wireless Communications, Networking and Mobile Computing., 2012.
- [IEEE\_ICWITS2012] Yi, J., Clausen, T., and A. Bas, "Smart Route Request for On-demand Route

Discovery in Constrained Environments",  
 Proceedings of IEEE ICWITS 2012, IEEE  
 International Conference on Wireless  
 Information Technology and Systems.,  
 2012.

## Appendix A. Gateway Considerations

For the LOADng implementations running in network layer, sometimes gateways are desired to connect to other networks. This section specifies how to enable gateways for LOADng routing domains.

A LOADng router in a network, which is a gateway to 'the larger Internet' MAY answer to RREQs for any destination except for destinations within the network itself for which it MUST NOT answer to RREQs. Consequently, a LOADng router, intended to act as a gateway, MUST be configured with the addresses which can occur within the network (ideally, the network is configured such that all devices share a common prefix).

## Appendix B. LOADng Control Messages using RFC5444

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

### B.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 8. A LOADng Router generating an RREQ (as specified in Section 12.1) when using the HOP_COUNT metric, MUST NOT add the METRIC Message TLV to the RREQ (in order to reduce overhead, as the hop count value is already encoded in RREQ.hop-count). LOADng Routers receiving an RREQ without METRIC Message TLV assume that RREQ.metric-type is HOP_COUNT, and MUST not add the METRIC Message TLV when forwarding the message. Otherwise, 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. (LOADng Routers generating RREQs when using the HOP_COUNT metric do not need need to add the METRIC Message TLV, as specified above for the RREQ.metric-type field.)
RREQ.hop-limit	<msg-hop-limit>	8 bits. MUST be included in an RREQ message
RREQ.hop-count	<msg-hop-count>	8 bits, hence MAX_HOP_COUNT is 255. MUST be included in an 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. An RREQ MUST contain exactly one address with a TLV of type ADDR-TYPE and with Type-Extension DESTINATION associated.
------------------	--------------------------------	---

Table 2: RREQ Message Elements

## B.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. A LOADng Router generating an RREP (as specified in Section 13.1) when using the HOP_COUNT metric, MUST NOT add the METRIC Message TLV to the RREP (in order to reduce overhead, as the hop count value is already encoded in RREP.hop-count). LOADng Routers receiving an RREP without METRIC Message TLV assume that RREP.metric-type is HOP_COUNT, and MUST not add the METRIC Message TLV when forwarding the message. Otherwise, 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. (LOADng Routers generating RREPs when using the HOP_COUNT metric do not need need to add the METRIC Message TLV, as specified above for the RREP.metric-type field.)
RREP.ackrequired	FLAGS Message TLV	Encoded by way of a Message-Type-specific Message TLV of type FLAGS, defined in Table 13. A TLV of type FLAGS MUST always be included in an RREP message.
RREP.hop-limit	<msg-hop-limit>	8 bits. MUST be included in an RREQ message

RREP.hop-count	<msg-hop-count>	8 bits, hence MAX_HOP_COUNT is 255. MUST be included in an 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. An RREP MUST contain exactly one address with a TLV of type ADDR-TYPE and with Type-Extension DESTINATION associated.

Table 3: RREP Message Elements

## B.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. An 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

## B.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.hop-limit	<msg-hop-limit>	8 bits. MUST be included in an RREQ message
RERR.errorcode	Address Block TLV Value	According to Section 19.1.

RERR.unreachableAddresses	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.originator	<msg-orig-addr>	MUST be included in an RERR message.
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. An RERR MUST contain exactly one address with a TLV of type ADDR-TYPE and with Type-Extension DESTINATION associated.

Table 5: RERR Message Elements

#### B.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].

##### B.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].

## B.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

## B.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	
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	HOP_COUNT: MSG.hop-count is used instead of the METRIC TLV Value. MAX_DIST is 255.	

METRIC	128	1	DIMENSIONLESS: A 32-bit, dimensionless, additive metric, single precision float, formatted according to [IEEE754-2008].	
METRIC	128	2-251	Unassigned	Expert Review
METRIC	128	252-255	Unassigned	Experimental

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

#### B.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	
129	FLAGS	
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	HOP_COUNT: MSG.hop-count is used instead of the METRIC TLV Value. MAX_DIST is 255.	
METRIC	128	1	DIMENSIONLESS: A 32-bit, dimensionless, additive metric, single precision float, formatted according to [IEEE754-2008].	
METRIC	128	2-251	Unassigned	Expert Review
METRIC	128	252-255	Unassigned	Experimental

Table 12: Message TLV Type assignment: METRIC

Allocation of the FLAGS 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
FLAGS	129	0	Bit 0 represents the ackrequired flag (i.e., ackrequired is TRUE when bit 0 is set to 1 and FALSE when bit 0 is 0.). All other bits are reserved for future use.	
FLAGS	129	1-255	Unassigned	Expert Review

Table 13: Message TLV Type assignment: FLAGS

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

## B.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

## B.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\_ACK 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 C. LOADng Control Packet Illustrations

This section presents example packets following this specification.

### C.1. RREQ

RREQ messages are instances of [RFC5444] messages. This specification requires that RREQ messages contain RREQ.msg-seq-num, RREQ.msg-hop-limit, RREQ.msg-hop-count and RREQ.msg-orig-addr fields.

It supports RREQ messages with any combination of remaining message header options and address encodings, enabled by [RFC5444] that convey the required information. As a consequence, there is no single way to represent how all RREQ messages look. This section illustrates an RREQ message, the exact values and content included are explained in the following text.

The RREQ message's four bit Message Flags (MF) field has value 15 indicating that the message header contains originator address, hop limit, hop count, and message sequence number fields. Its four bit Message Address Length (MAL) field has value 3, indicating addresses in the message have a length of four octets, here being IPv4 addresses. The overall message length is 30 octets.

The message has a Message TLV Block with content length 6 octets containing one TLV. The TLVs is of type METRIC and has a Flags octet (MTLVF) value 144, indicating that it has a Value, a type extension, but no start and stop indexes. The Value Length of the METRIC TLV is 2 octets.

The message has one Address Block. The Address Block contains 1 address, with Flags octet (ATLVF) value 0, hence with no Head or Tail sections, and hence with a Mid section with length four octets. The following TLV Block (content length 2 octets) contains one TLV. The TLV is an ADDR\_TYPE TLV with Flags octet (ATLVF) value 0, indicating no Value and no indexes. Thus, the address is associated with the Type ADDR\_TYPE, i.e., it is the destination address of the RREQ.

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
RREQ										MF=15										MAL=3										Message Length = 30									
Originator Address																																							
Hop Limit										Hop Count										Message Sequence Number																			
Message TLV Block Length = 6																				METRIC										MTLVF = 144									
Type Ext.										Value Len = 2										Value (metric)																			
Num Addrs = 1										ABF = 0										Mid																			
Mid																				Address TLV Block Length = 2																			
ADDR-TYPE										ATLVF = 0																													

## C.2. RREP

RREP messages are instances of [RFC5444] messages. This specification requires that RREP messages contain RREP.msg-seq-num, RREP.msg-hop-limit, RREP.msg-hop-count and RREP.msg-orig-addr fields. It supports RREP messages with any combination of remaining message header options and address encodings, enabled by [RFC5444] that convey the required information. As a consequence, there is no single way to represent how all RREP messages look. This section illustrates an RREP message, the exact values and content included are explained in the following text.

The RREP message's four bit Message Flags (MF) field has value 15 indicating that the message header contains originator address, hop limit, hop count, and message sequence number fields. Its four bit Message Address Length (MAL) field has value 3, indicating addresses in the message have a length of four octets, here being IPv4 addresses. The overall message length is 34 octets.

The message has a Message TLV Block with content length 10 octets containing two TLVs. The first TLV is of type METRIC and has a Flags octet (MTLVF) value 144, indicating that it has a Value, a type extension, but no start and stop indexes. The Value Length of the METRIC TLV is 2 octets. The second TLV is of type FLAGS and has a Flags octet (MTLVF) value of 16, indicating that it has a Value, but no type extension or start and stop indexes. The Value Length of the FLAGS TLV is 1 octet. The TLV value is 0x80 indicating that the ackrequired flag is set.

The message has one Address Block. The Address Block contains 1 address, with Flags octet (ATLVF) value 0, hence with no Head or Tail sections, and hence with a Mid section with length four octets. The following TLV Block (content length 2 octets) contains one TLV. The TLV is an ADDR\_TYPE TLV with Flags octet (ATLVF) value 0, indicating no Value and no indexes. Thus, the address is associated with the Type ADDR\_TYPE, i.e., it is the destination address of the RREP.

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|      RREP      | MF=15 | MAL=3 |      Message Length = 34      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Originator Address          |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Hop Limit      | Hop Count |      Message Sequence Number      |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Message TLV Block Length = 10 |      METRIC      | MTLVF = 144 |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Type Ext.      | Value Len = 2 |      Value (metric)      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      FLAGS      | MTLVF = 16 | Value Len = 1 | Value (0x80) |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Num Addrs = 1 | ABF = 0 |      Mid      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Mid      | Address TLV Block Length = 2 |
+-----+-----+-----+-----+-----+-----+-----+-----+
| ADDR-TYPE      | ATLVF = 0 |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

### C.3. RREP\_ACK

RREP\_ACK messages are instances of [RFC5444] messages. This specification requires that RREP\_ACK messages contains RREP\_ACK.msg-seq-num. It supports RREP\_ACK messages with any combination of remaining message header options and address encodings, enabled by [RFC5444] that convey the required information. As a consequence, there is no single way to represent how all RREP\_ACK messages look. This section illustrates an RREP\_ACK message, the exact values and content included are explained in the following text.

The RREP\_ACK message's four bit Message Flags (MF) field has value 1 indicating that the message header contains the message sequence number field. Its four bit Message Address Length (MAL) field has value 3, indicating addresses in the message have a length of four octets, here being IPv4 addresses. The overall message length is 18 octets.

The message has a Message TLV Block with content length 0 octets containing zero TLVs.

The message has one Address Block. The Address Block contains 1 address, with Flags octet (ATLVF) value 0, hence with no Head or Tail sections, and hence with a Mid section with length four octets. The following TLV Block (content length 2 octets) contains one TLV. The TLV is an ADDR\_TYPE TLV with Flags octet (ATLVF) value 0, indicating

no Value and no indexes. Thus, the address is associated with the Type ADDR\_TYPE, i.e., it is the destination address of the RREP\_ACK.

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|  RREP_ACK    | MF=1  | MAL=3  |      Message Length = 18      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Message Sequence Number      | Message TLV Block Length = 0 |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Num Addrs = 1 |      ABF = 0      |      Mid      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Mid      | Address TLV Block Length = 2 |
+-----+-----+-----+-----+-----+-----+-----+-----+
|  ADDR-TYPE   |      ATLVF = 0      |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

#### C.4. RERR

RERR messages are instances of [RFC5444] messages. This specification supports RERR messages with any combination of message header options and address encodings, enabled by [RFC5444] that convey the required information. As a consequence, there is no single way to represent how all RERR messages look. This section illustrates an RERR message, the exact values and content included are explained in the following text.

The RERR message's four bit Message Flags (MF) field has value 12 indicating that the message header contains RERR.msg-orig-addr field and RERR.msg-hop-limit field. Its four bit Message Address Length (MAL) field has value 3, indicating addresses in the message have a length of four octets, here being IPv4 addresses. The overall message length is 30 octets.

The message has a Message TLV Block with content length 0 octets containing zero TLVs.

The message has one Address Block. The Address Block contains 2 addresses, with Flags octet (ATLVF) value 128, hence with a Head section (with length 3 octets), but no Tail section, and hence with Mid sections with length one octet. The following TLV Block (content length 9 octets) contains two TLVs. The first TLV is an ADDR\_TYPE TLV with Flags octet (ATLVF) value 64, indicating a single index of 0, but no Value. Thus, the first address is associated with the Type ADDR\_TYPE and Type Extension DESTINATION, i.e., it is the destination address of the RERR. The second TLV is an ADDR\_TYPE TLV with Flags octet (ATLVF) value 208, indicating Type Extension, Value, and single index. Thus, the second address is associated with the Type



ADDR\_TYPE, Type Extension ERRORCODE, and Value 0, i.e., it is associated with error code 0.

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|      RERR      | MF=12 | MAL=3 |      Message Length = 30      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Originator Address                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Hop Limit      | Message TLV Block Length = 0 | Num Addrs = 2 |
+-----+-----+-----+-----+-----+-----+-----+-----+
| ABF = 128      | Head Len = 3 |      Head      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|      Head      |      Mid      |      Address TLV      |
+-----+-----+-----+-----+-----+-----+-----+-----+
|Block Length= 9| ADDR-TYPE | ATLVF = 64 | Index = 0 |
+-----+-----+-----+-----+-----+-----+-----+-----+
| ADDR-TYPE      | ATLVF = 208 |TypeEx=ERRORCODE| Index = 1 |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Value Len = 1 | Value = 0 |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

#### Authors' Addresses

Thomas Heide Clausen  
Ecole Polytechnique

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

Axel Colin de Verdiere  
Ecole Polytechnique

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

Jiazi Yi  
Ecole Polytechnique

Phone: +33 1 6933 4031  
EMail: [jiazi@jiaziyi.com](mailto:jiazi@jiaziyi.com)  
URI: <http://www.jiaziyi.com/>

Afshin Niktash  
Maxim Integrated Products

Phone: +1 94 9450 1692  
EMail: [afshin.niktash@maxim-ic.com](mailto: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](mailto: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](mailto:hiroki.satoh.yj@hitachi.com)  
URI: <http://www.hitachi.com/>

Ulrich Herberg

EMail: [ulrich@herberg.name](mailto:ulrich@herberg.name)  
URI: <http://www.herberg.name/>

Cedric Lavenu  
EDF R&D

Phone: +33 1 4765 2729  
EMail: [cedric-2.lavenu@edf.fr](mailto:cedric-2.lavenu@edf.fr)  
URI: <http://www.edf.fr/>

Thierry Lys  
ERDF

Phone: +33 1 8197 6777  
EMail: [thierry.lys@erdfdistribution.fr](mailto:thierry.lys@erdfdistribution.fr)  
URI: <http://www.erdfdistribution.fr/>

Justin Dean  
Naval Research Laboratory

EMail: [jdean@itd.nrl.navy.mil](mailto:jdean@itd.nrl.navy.mil)  
URI: <http://cs.itd.nrl.navy.mil/>



Mobile Ad hoc Networks Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: September 29, 2017

S. Ratliff  
VT iDirect  
S. Jury  
Cisco Systems  
D. Satterwhite  
Broadcom  
R. Taylor  
Airbus Defence & Space  
B. Berry  
March 28, 2017

Dynamic Link Exchange Protocol (DLEP)  
draft-ietf-manet-dlep-29

Abstract

When routing devices rely on modems to effect communications over wireless links, they need timely and accurate knowledge of the characteristics of the link (speed, state, etc.) in order to make routing decisions. In mobile or other environments where these characteristics change frequently, manual configurations or the inference of state through routing or transport protocols does not allow the router to make the best decisions. DLEP describes a new protocol for a bidirectional, event-driven communication channel between the router and the modem to facilitate communication of changing link characteristics.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on September 29, 2017.

## Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	4
2. Protocol Overview . . . . .	7
2.1. Destinations . . . . .	8
2.2. Conventions and Terminology . . . . .	9
3. Requirements . . . . .	9
4. Implementation Scenarios . . . . .	9
5. Assumptions . . . . .	10
6. Metrics . . . . .	11
7. DLEP Session Flow . . . . .	12
7.1. Peer Discovery State . . . . .	12
7.2. Session Initialization State . . . . .	13
7.3. In-Session State . . . . .	14
7.3.1. Heartbeats . . . . .	14
7.4. Session Termination State . . . . .	15
7.5. Session Reset state . . . . .	15
7.5.1. Unexpected TCP connection termination . . . . .	16
8. Transaction Model . . . . .	16
9. Extensions . . . . .	17
9.1. Experiments . . . . .	17
10. Scalability . . . . .	18
11. DLEP Signal and Message Structure . . . . .	18
11.1. DLEP Signal Header . . . . .	18
11.2. DLEP Message Header . . . . .	19
11.3. DLEP Generic Data Item . . . . .	20
12. DLEP Signals and Messages . . . . .	20
12.1. General Processing Rules . . . . .	20
12.2. Status code processing . . . . .	21
12.3. Peer Discovery Signal . . . . .	22
12.4. Peer Offer Signal . . . . .	22
12.5. Session Initialization Message . . . . .	23
12.6. Session Initialization Response Message . . . . .	24

12.7.	Session Update Message . . . . .	25
12.8.	Session Update Response Message . . . . .	27
12.9.	Session Termination Message . . . . .	27
12.10.	Session Termination Response Message . . . . .	27
12.11.	Destination Up Message . . . . .	28
12.12.	Destination Up Response Message . . . . .	29
12.13.	Destination Announce Message . . . . .	30
12.14.	Destination Announce Response Message . . . . .	30
12.15.	Destination Down Message . . . . .	32
12.16.	Destination Down Response Message . . . . .	32
12.17.	Destination Update Message . . . . .	32
12.18.	Link Characteristics Request Message . . . . .	34
12.19.	Link Characteristics Response Message . . . . .	34
12.20.	Heartbeat Message . . . . .	35
13.	DLEP Data Items . . . . .	36
13.1.	Status . . . . .	37
13.2.	IPv4 Connection Point . . . . .	39
13.3.	IPv6 Connection Point . . . . .	40
13.4.	Peer Type . . . . .	41
13.5.	Heartbeat Interval . . . . .	42
13.6.	Extensions Supported . . . . .	43
13.7.	MAC Address . . . . .	44
13.8.	IPv4 Address . . . . .	44
13.8.1.	IPv4 Address Processing . . . . .	45
13.9.	IPv6 Address . . . . .	46
13.9.1.	IPv6 Address Processing . . . . .	47
13.10.	IPv4 Attached Subnet . . . . .	48
13.10.1.	IPv4 Attached Subnet Processing . . . . .	49
13.11.	IPv6 Attached Subnet . . . . .	50
13.11.1.	IPv6 Attached Subnet Processing . . . . .	51
13.12.	Maximum Data Rate (Receive) . . . . .	52
13.13.	Maximum Data Rate (Transmit) . . . . .	53
13.14.	Current Data Rate (Receive) . . . . .	54
13.15.	Current Data Rate (Transmit) . . . . .	54
13.16.	Latency . . . . .	55
13.17.	Resources . . . . .	56
13.18.	Relative Link Quality (Receive) . . . . .	57
13.19.	Relative Link Quality (Transmit) . . . . .	57
13.20.	Maximum Transmission Unit (MTU) . . . . .	58
14.	Security Considerations . . . . .	59
15.	IANA Considerations . . . . .	60
15.1.	Registrations . . . . .	60
15.2.	Signal Type Registration . . . . .	60
15.3.	Message Type Registration . . . . .	61
15.4.	DLEP Data Item Registrations . . . . .	61
15.5.	DLEP Status Code Registrations . . . . .	62
15.6.	DLEP Extensions Registrations . . . . .	63
15.7.	DLEP IPv4 Connection Point Flags . . . . .	63

15.8.	DLEP IPv6 Connection Point Flags . . . . .	64
15.9.	DLEP Peer Type Flag . . . . .	64
15.10.	DLEP IPv4 Address Flag . . . . .	64
15.11.	DLEP IPv6 Address Flag . . . . .	65
15.12.	DLEP IPv4 Attached Subnet Flag . . . . .	65
15.13.	DLEP IPv6 Attached Subnet Flag . . . . .	65
15.14.	DLEP Well-known Port . . . . .	66
15.15.	DLEP IPv4 Link-local Multicast Address . . . . .	66
15.16.	DLEP IPv6 Link-local Multicast Address . . . . .	66
16.	Acknowledgments . . . . .	66
17.	References . . . . .	66
17.1.	Normative References . . . . .	66
17.2.	Informative References . . . . .	67
Appendix A.	Discovery Signal Flows . . . . .	68
Appendix B.	Peer Level Message Flows . . . . .	68
B.1.	Session Initialization . . . . .	68
B.2.	Session Initialization - Refused . . . . .	69
B.3.	Router Changes IP Addresses . . . . .	70
B.4.	Modem Changes Session-wide Metrics . . . . .	70
B.5.	Router Terminates Session . . . . .	70
B.6.	Modem Terminates Session . . . . .	71
B.7.	Session Heartbeats . . . . .	71
B.8.	Router Detects a Heartbeat timeout . . . . .	72
B.9.	Modem Detects a Heartbeat timeout . . . . .	73
Appendix C.	Destination Specific Message Flows . . . . .	73
C.1.	Common Destination Notification . . . . .	73
C.2.	Multicast Destination Notification . . . . .	74
C.3.	Link Characteristics Request . . . . .	75
Authors' Addresses	. . . . .	76

## 1. Introduction

There exist today a collection of modem devices that control links of variable datarate and quality. Examples of these types of links include line-of-sight (LOS) terrestrial radios, satellite terminals, and broadband modems. Fluctuations in speed and quality of these links can occur due to configuration, or on a moment-to-moment basis, due to physical phenomena like multipath interference, obstructions, rain fade, etc. It is also quite possible that link quality and datarate vary with respect to individual destinations on a link, and with the type of traffic being sent. As an example, consider the case of an IEEE 802.11 access point, serving two associated laptop computers. In this environment, the answer to the question "What is the datarate on the 802.11 link?" is "It depends on which associated laptop we're talking about, and on what kind of traffic is being sent." While the first laptop, being physically close to the access point, may have a datarate of 54Mbps for unicast traffic, the other laptop, being relatively far away, or obstructed by some object, can



simultaneously have a datarate of only 32Mbps for unicast. However, for multicast traffic sent from the access point, all traffic is sent at the base transmission rate (which is configurable, but depending on the model of the access point, is usually 24Mbps or less).

In addition to utilizing variable datarate links, mobile networks are challenged by the notion that link connectivity will come and go over time, without an effect on a router's interface state (Up or Down). Effectively utilizing a relatively short-lived connection is problematic in IP routed networks, as IP routing protocols tend to rely on interface state and independent timers to maintain network convergence (e.g., HELLO messages and/or recognition of DEAD routing adjacencies). These dynamic connections can be better utilized with an event-driven paradigm, where acquisition of a new neighbor (or loss of an existing one) is signaled, as opposed to a paradigm driven by timers and/or interface state. DLEP not only implements such an event-driven paradigm, but does so over a local (1 hop) TCP session, which guarantees delivery of the event messages.

Another complicating factor for mobile networks are the different methods of physically connecting the modem devices to the router. Modems can be deployed as an interface card in a router's chassis, or as a standalone device connected to the router via Ethernet or serial link. In the case of Ethernet attachment, with existing protocols and techniques, routing software cannot be aware of convergence events occurring on the radio link (e.g., acquisition or loss of a potential routing neighbor), nor can the router be aware of the actual capacity of the link. This lack of awareness, along with the variability in datarate, leads to a situation where finding the (current) best route through the network to a given node is difficult to establish and properly maintain. This is especially true of demand-based access schemes such as Demand Assigned Multiple Access (DAMA) implementations used on some satellite systems. With a DAMA-based system, additional datarate may be available, but will not be used unless the network devices emit traffic at a rate higher than the currently established rate. Increasing the traffic rate does not guarantee additional datarate will be allocated; rather, it may result in data loss and additional retransmissions on the link.

Addressing the challenges listed above, the Dynamic Link Exchange Protocol, or DLEP, has been developed. The DLEP protocol runs between a router and its attached modem devices, allowing the modem to communicate link characteristics as they change, and convergence events (acquisition and loss of potential routing next-hops). The following diagrams are used to illustrate the scope of DLEP packets.

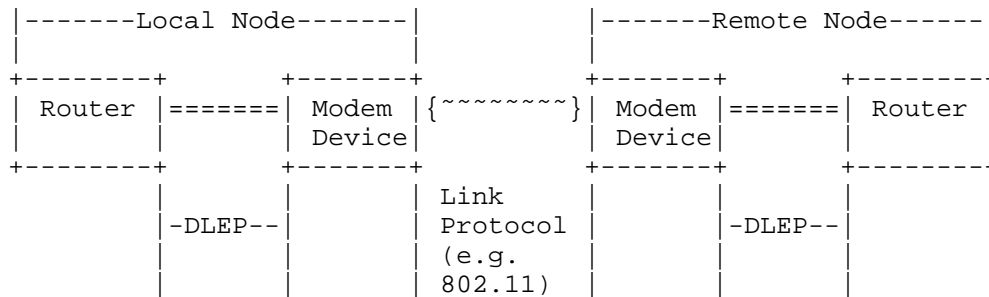


Figure 1: DLEP Network

In Figure 1, when the local modem detects the presence of a remote node, it (the local modem) sends a message to its router via the DLEP protocol. The message consists of an indication of what change has occurred on the link (e.g., presence of a remote node detected), along with a collection of DLEP-defined data items that further describe the change. Upon receipt of the message, the local router may take whatever action it deems appropriate, such as initiating discovery protocols, and/or issuing HELLO messages to converge the network. On a continuing, as-needed basis, the modem devices use DLEP to report any characteristics of the link (datarate, latency, etc.) that have changed. DLEP is independent of the link type and topology supported by the modem. Note that the DLEP protocol is specified to run only on the local link between router and modem. Some over the air signaling may be necessary between the local and remote modem in order to provide some parameters in DLEP messages between the local modem and local router, but DLEP does not specify how such over the air signaling is carried out. Over the air signaling is purely a matter for the modem implementer.

Figure 2 shows how DLEP can support a configuration where routers are connected with different link types. In this example, Modem A implements a point-to-point link, and Modem B is connected via a shared medium. In both cases, the DLEP protocol is used to report the characteristics of the link (datarate, latency, etc.) to routers. The modem is also able to use the DLEP session to notify the router when the remote node is lost, shortening the time required to re-converge the network.

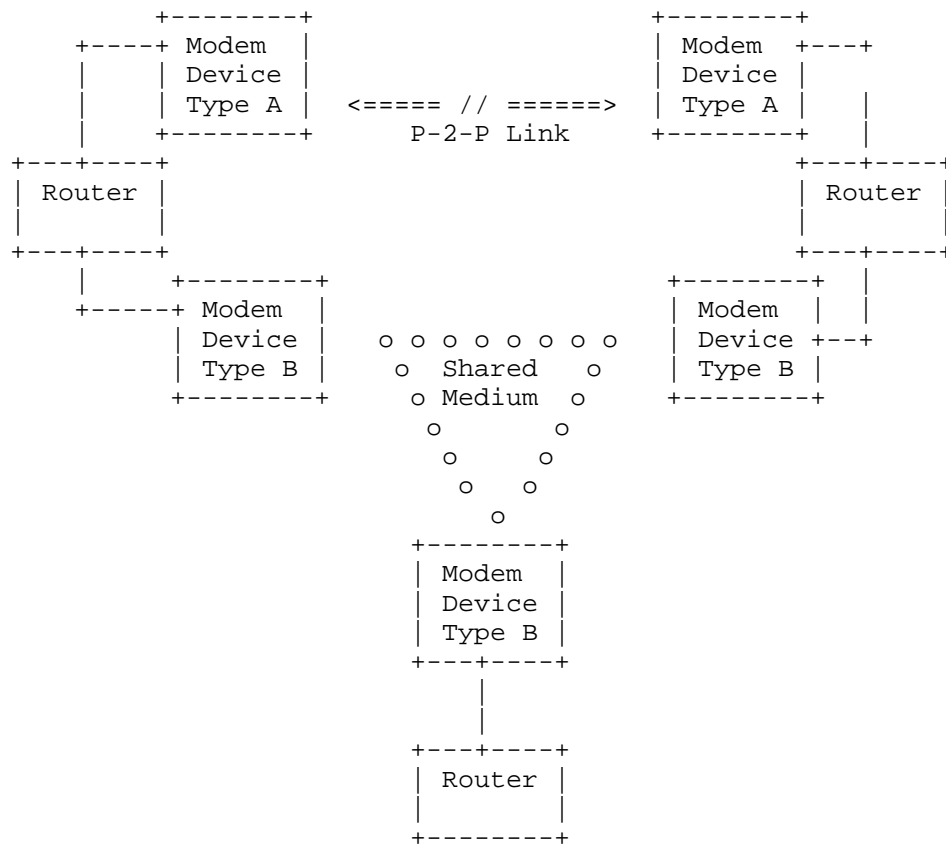


Figure 2: DLEP Network with Multiple Modem Devices

## 2. Protocol Overview

DLEP defines a set of Messages used by modems and their attached routers to communicate events that occur on the physical link(s) managed by the modem: for example, a remote node entering or leaving the network, or that the link has changed. Associated with these Messages are a set of Data Items - information that describes the remote node (e.g., address information), and/or the characteristics of the link to the remote node. Throughout this document, we refer to a modems/routers participating in a DLEP session as 'DLEP Participants', unless a specific distinction (e.g. modem or router) is required.

DLEP uses a session-oriented paradigm between the modem device and its associated router. If multiple modem devices are attached to a router (as in Figure 2), or the modem supports multiple connections

(via multiple logical or physical interfaces), then separate DLEP sessions exist for each modem or connection. A router and modem form a session by completing the discovery and initialization process. This router-modem session persists unless or until it either (1) times out, based on the absence of DLEP traffic (including heartbeats), or (2) is explicitly torn down by one of the DLEP participants.

While this document represents the best efforts of the working group to be functionally complete, it is recognized that extensions to DLEP will in all likelihood be necessary as more link types are used. Such extensions are defined as additional Messages, Data Items and/or status codes, and associated rules of behavior, that are not defined in this document. DLEP contains a standard mechanism for router and modem implementations to negotiate the available extensions to use on a per-session basis.

## 2.1. Destinations

The router/modem session provides a carrier for information exchange concerning 'destinations' that are available via the modem device. Destinations can be identified by either the router or the modem, and represent a specific, addressable location that can be reached via the link(s) managed by the modem.

The DLEP Messages concerning destinations thus become the way for routers and modems to maintain, and notify each other about, an information base representing the physical and logical destinations accessible via the modem device, as well as the link characteristics to those destinations.

A destination can be either physical or logical. The example of a physical destination would be that of a remote, far-end router attached via the variable-quality network. It should be noted that for physical destinations the MAC address is the address of the far-end router, not the modem.

The example of a logical destination is Multicast. Multicast traffic destined for the variable-quality network (the network accessed via the modem) is handled in IP networks by deriving a Layer 2 MAC address based on the Layer 3 address. Leveraging on this scheme, multicast traffic is supported in DLEP simply by treating the derived MAC address as any other destination in the network.

To support these logical destinations, one of the DLEP participants (typically, the router) informs the other as to the existence of the logical destination. The modem, once it is aware of the existence of this logical destination, reports link characteristics just as it

would for any other destination in the network. The specific algorithms a modem would use to derive metrics on logical destinations are outside the scope of this specification, and is left to specific implementations to decide.

In all cases, when this specification uses the term destination, it refers to the addressable locations, either logical or physical, that are accessible by the radio link(s).

## 2.2. Conventions and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14, RFC 2119 [RFC2119].

## 3. Requirements

DLEP MUST be implemented on a single Layer 2 domain. The protocol identifies next-hop destinations by using the MAC address for delivering data traffic. No manipulation or substitution is performed; the MAC address supplied in all DLEP Messages is used as the Destination MAC address for frames emitted by the participating router. MAC addresses MUST be unique within the context of router-modem session.

To enforce the single Layer 2 domain, implementations MUST support The Generalized TTL Security Mechanism [RFC5082], and implementations MUST adhere to this specification for all DLEP Messages.

DLEP specifies UDP multicast for single-hop discovery signaling, and TCP for transport of the Messages. Modems and routers participating in DLEP sessions MUST have topologically consistent IP addresses assigned. It is RECOMMENDED that DLEP implementations utilize IPv6 link-local addresses to reduce the administrative burden of address assignment.

DLEP relies on the guaranteed delivery of its Messages between router and modem, once the 1 hop discovery process is complete, hence, the specification of TCP to carry the Messages. Other reliable transports for the protocol are possible, but are outside the scope of this document.

## 4. Implementation Scenarios

During development of this specification, two types of deployments were discussed.

The first can be viewed as a "dedicated deployment". In this mode, DLEP routers and modems are either directly connected (e.g., using cross-over cables to connect interfaces), or are connected to a dedicated switch. An example of this type of deployment would be a router with a line-of-sight radio connected into one interface, with a satellite modem connected into another interface. In mobile environments, the router and the connected modem(s) are placed into a mobile platform (e.g., a vehicle, boat, or airplane). In this mode, when a switch is used, it is possible that a small number of ancillary devices (e.g., a laptop) are also plugged into the switch. But in either event, the resulting network segment is constrained to a small number of devices, and is not generally accessible from anywhere else in the network.

The other type of deployment envisioned can be viewed as a "networked deployment". In this type of scenario, the DLEP router and modem(s) are placed on a segment that is accessible from other points in the network. In this scenario, not only are the DLEP router and modem(s) accessible from other points in the network; the router and a given modem could be multiple physical hops away from each other. This scenario necessitates the use of Layer 2 tunneling technology to enforce the single-hop requirement of DLEP.

## 5. Assumptions

DLEP assumes that a signaling protocol exists between modems participating in a network. The specification does not define the character or behavior of this over-the-air signaling, but does expect some information to be carried (or derived) by the signaling, such as the arrival and departure of modems from this network, and the variation of the link characteristics between modems. This information is then assumed to be used by the modem to implement the DLEP protocol.

The specification assumes that the link between router and modem is static with respect to data rate and latency, and that this link is not likely to be the cause of a performance bottleneck. In deployments where the router and modem are physically separated by multiple network hops, served by Layer 2 tunneling technology, DLEP statistics on the RF links could be insufficient for routing protocols to make appropriate routing decisions. This would especially become an issue in cases where the Layer 2 tunnel between router and modem is itself served in part (or in total) with a wireless back-haul link.

## 6. Metrics

DLEP includes the ability for the router and modem to communicate metrics that reflect the characteristics (e.g., data rate, latency) of the variable-quality link in use. DLEP does not specify how a given metric value is to be calculated, rather, the protocol assumes that metrics have been calculated by a 'best effort', incorporating all pertinent data that is available to the modem device. Metrics based on large enough sample sizes will preclude short traffic bursts from adversely skewing reported values.

DLEP allows for metrics to be sent within two contexts - metrics for a specific destination within the network (e.g., a specific router), and per-session (those that apply to all destinations accessed via the modem). Most metrics can be further subdivided into transmit and receive metrics. In cases where metrics are provided at session level, the router propagates the metrics to all entries in its information base for destinations that are accessed via the modem.

DLEP modems announce all metric Data Items that will be reported during the session, and provide default values for those metrics, in the Session Initialization Response Message (Section 12.6). In order to use a metric type that was not included in the Session Initialization Response Message, modem implementations terminate the session with the router (via the Session Terminate Message (Section 12.9)), and establish a new session.

A DLEP modem can send metrics both in a session context, via the Session Update Message (Section 12.7), and a specific destination context, via the Destination Update Message (Section 12.17), at any time. The most recently received metric value takes precedence over any earlier value, regardless of context - that is:

1. If the router receives metrics in a specific destination context (via the Destination Update Message), then the specific destination is updated with the new metric.
2. If the router receives metrics in a session-wide context (via the Session Update Message), then the metrics for all destinations accessed via the modem are updated with the new metric.

It is left to implementations to choose sensible default values based on their specific characteristics. Modems having static (non-changing) link metric characteristics can report metrics only once for a given destination (or once on a session-wide basis, if all connections via the modem are of this static nature).

In addition to communicating existing metrics about the link, DLEP provides a Message allowing a router to request a different datarate or latency from the modem. This Message is the Link Characteristics Request Message (Section 12.18), and gives the router the ability to deal with requisite increases (or decreases) of allocated datarate/latency in demand-based schemes in a more deterministic manner.

## 7. DLEP Session Flow

All DLEP participants of a session transition through a number of distinct states during the lifetime of a DLEP session:

- o Peer Discovery
- o Session Initialization
- o In-Session
- o Session Termination
- o Session Reset

Modems, and routers supporting DLEP discovery, transition through all five (5) of the above states. Routers that rely on preconfigured TCP address/port information start in the Session Initialization state.

Modems **MUST** support the Peer Discovery state.

### 7.1. Peer Discovery State

Modems **MUST** support DLEP Peer Discovery; routers **MAY** support the discovery signals, or rely on a priori configuration to locate modems. If a router chooses to support DLEP discovery, all signals **MUST** be supported.

In the Peer Discovery state, routers that support DLEP discovery **MUST** send Peer Discovery Signals (Section 12.3) to initiate modem discovery.

The router implementation then waits for a Peer Offer Signal (Section 12.4) response from a potential DLEP modem. While in the Peer Discovery state, Peer Discovery Signals **MUST** be sent repeatedly by a DLEP router, at regular intervals. It is **RECOMMENDED** that this interval be set to 60 seconds. The interval **MUST** be a minimum of one second; it **SHOULD** be a configurable parameter. Note that this operation (sending Peer Discovery and waiting for Peer Offer) is outside the DLEP Transaction Model (Section 8), as the Transaction Model only describes Messages on a TCP session.



Routers receiving a Peer Offer Signal MUST use one of the modem address/port combinations from the Peer Offer Signal to establish a TCP connection to the modem, even if a priori configuration exists. If multiple connection point Data Items exist in the received Peer Offer Signal, routers SHOULD prioritize IPv6 connection points over IPv4 connection points. If multiple connection points exist with the same transport (e.g. IPv6 or IPv4), implementations MAY use their own heuristics to determine the order in which they are tried. If a TCP connection cannot be achieved using any of the address/port combinations and the Discovery mechanism is in use, then the router SHOULD resume issuing Peer Discovery Signals. If no Connection Point Data Items are included in the Peer Offer Signal, the router MUST use the source address of the UDP packet containing the Peer Offer Signal as the IP address, and the DLEP well-known port number.

In the Peer Discovery state, the modem implementation MUST listen for incoming Peer Discovery Signals on the DLEP well-known IPv6 and/or IPv4 link-local multicast address and port. On receipt of a valid Peer Discovery Signal, it MUST reply with a Peer Offer Signal.

Modems MUST be prepared to accept a TCP connection from a router that is not using the Discovery mechanism, i.e. a connection attempt that occurs without a preceding Peer Discovery Signal.

Implementations of DLEP SHOULD implement, and use, TLS [RFC5246] to protect the TCP session. The "dedicated deployments" discussed in Implementation Scenarios (Section 4) MAY consider use of DLEP without TLS. For all "networked deployments" (again, discussed in Implementation Scenarios), implementation and use of TLS is STRONGLY RECOMMENDED. If TLS is to be used then the TLS session MUST be established before any Messages are passed between peers. Routers supporting TLS MUST prioritize connection points using TLS over those that do not.

Upon establishment of a TCP connection, and TLS session if TLS is in use, both modem and router enter the Session Initialization state. It is up to the router implementation if Peer Discovery Signals continue to be sent after the device has transitioned to the Session Initialization state. Modem implementations MUST silently ignore Peer Discovery Signals from a router with which it already has a TCP connection.

## 7.2. Session Initialization State

On entering the Session Initialization state, the router MUST send a Session Initialization Message (Section 12.5) to the modem. The router MUST then wait for receipt of a Session Initialization Response Message (Section 12.6) from the modem. Receipt of the

Session Initialization Response Message containing a Status Data Item (Section 13.1) with status code set to 0 'Success', see Table 2, indicates that the modem has received and processed the Session Initialization Message, and the router MUST transition to the In-Session state.

On entering the Session Initialization state, the modem MUST wait for receipt of a Session Initialization Message from the router. Upon receipt of a Session Initialization Message, the modem MUST send a Session Initialization Response Message, and the session MUST transition to the In-Session state. If the modem receives any Message other than Session Initialization, or it fails to parse the received Message, it MUST NOT send any Message, and MUST terminate the TCP connection and transition to the Session Reset state.

DLEP provides an extension negotiation capability to be used in the Session Initialization state, see Section 9. Extensions supported by an implementation MUST be declared to potential DLEP participants using the Extensions Supported Data Item (Section 13.6). Once both DLEP participants have exchanged initialization Messages, an implementation MUST NOT emit any Message, Signal, Data Item or status code associated with an extension that was not specified in the received initialization Message from its peer.

### 7.3. In-Session State

In the In-Session state, Messages can flow in both directions between DLEP participants, indicating changes to the session state, the arrival or departure of reachable destinations, or changes of the state of the links to the destinations.

The In-Session state is maintained until one of the following conditions occur:

- o The implementation terminates the session by sending a Session Termination Message (Section 12.9), or,
- o Its peer terminates the session, indicated by receiving a Session Termination Message.

The implementation MUST then transition to the Session Termination state.

#### 7.3.1. Heartbeats

In order to maintain the In-Session state, periodic Heartbeat Messages (Section 12.20) MUST be exchanged between router and modem. These Messages are intended to keep the session alive, and to verify

bidirectional connectivity between the two DLEP participants. It is RECOMMENDED that the interval timer between heartbeat messages be set to 60 seconds. The interval MUST be a minimum of one second; it SHOULD be a configurable parameter.

Each DLEP participant is responsible for the creation of Heartbeat Messages.

Receipt of any valid DLEP Message MUST reset the heartbeat interval timer (i.e., valid DLEP Messages take the place of, and obviate the need for, additional Heartbeat Messages).

Implementations MUST allow a minimum of two (2) heartbeat intervals to expire with no Messages from its peer before terminating the session. When terminating the session, a Session Termination Message containing a Status Data Item (Section 13.1) with status code set to 132 'Timed Out', see Table 2, MUST be sent, and then the implementation MUST transition to the Session Termination state.

#### 7.4. Session Termination State

When an implementation enters the Session Termination state after sending a Session Termination Message (Section 12.9) as the result of an invalid Message or error, it MUST wait for a Session Termination Response Message (Section 12.10) from its peer. Senders SHOULD allow four (4) heartbeat intervals to expire before assuming that its peer is unresponsive, and continuing with session termination. Any other Message received while waiting MUST be silently ignored.

When the sender of the Session Termination Message receives a Session Termination Response Message from its peer, or times out, it MUST transition to the Session Reset state.

When an implementation receives a Session Termination Message from its peer, it enters the Session Termination state and then it MUST immediately send a Session Termination Response and transition to the Session Reset state.

#### 7.5. Session Reset state

In the Session Reset state the implementation MUST perform the following actions:

- o Release all resources allocated for the session.
- o Eliminate all destinations in the information base represented by the session. Destination Down Messages (Section 12.15) MUST NOT be sent.

- o Terminate the TCP connection.

Having completed these actions the implementation SHOULD return to the relevant initial state: Peer Discovery for modems; either Peer Discovery or Session Initialization for routers, depending on configuration.

#### 7.5.1. Unexpected TCP connection termination

If the TCP connection between DLEP participants is terminated when an implementation is not in the Session Reset state, the implementation MUST immediately transition to the Session Reset state.

### 8. Transaction Model

DLEP defines a simple Message transaction model: Only one request per destination may be in progress at a time per session. A Message transaction is considered complete when a response matching a previously issued request is received. If a DLEP participant receives a request for a destination for which there is already an outstanding request, the implementation MUST terminate the session by issuing a Session Termination Message (Section 12.9) containing a Status Data Item (Section 13.1) with status code set to 129 'Unexpected Message', see Table 2, and transition to the Session Termination state. There is no restriction to the total number of Message transactions in progress at a time, as long as each transaction refers to a different destination.

It should be noted that some requests may take a considerable amount of time for some DLEP participants to complete, for example, a modem handling a multicast destination up request may have to perform a complex network reconfiguration. A sending implementation MUST be able to handle such long running transactions gracefully.

Additionally, only one session request, e.g. a Session Initialization Message (Section 12.5), may be in progress at a time per session. As above, a session transaction is considered complete when a response matching a previously issued request is received. If a DLEP participant receives a session request while there is already a session request in progress, it MUST terminate the session by issuing a Session Termination Message containing a Status Data Item with status code set to 129 'Unexpected Message', and transition to the Session Termination state. Only the Session Termination Message may be issued when a session transaction is in progress. Heartbeat Messages (Section 12.20) MUST NOT be considered part of a session transaction.

DLEP transactions do not time out and are not cancellable, except for transactions in-flight when the DLEP session is reset. If the session is terminated, canceling transactions in progress MUST be performed as part of resetting the state machine. An implementation can detect if its peer has failed in some way by use of the session heartbeat mechanism during the In-Session state, see Section 7.3.

## 9. Extensions

Extensions MUST be negotiated on a per-session basis during session initialization via the Extensions Supported mechanism. Implementations are not required to support any extension in order to be considered DLEP compliant.

If interoperable protocol extensions are required, they will need to be standardized either as an update to this document, or as an additional stand-alone specification. The requests for IANA-controlled registries in this document contain sufficient Reserved space for DLEP Signals, Messages, Data Items and status codes to accommodate future extensions to the protocol.

As multiple protocol extensions MAY be announced during session initialization, authors of protocol extensions need to consider the interaction of their extension with other published extensions, and specify any incompatibilities.

### 9.1. Experiments

This document requests Private Use numbering space in the DLEP Signal, Message, Data Item and status code registries for experimental extensions. The intent is to allow for experimentation with new Signals, Messages, Data Items, and/or status codes, while still retaining the documented DLEP behavior.

Use of the Private Use Signals, Messages, Data Items, status codes, or behaviors MUST be announced as DLEP Extensions, during session initialization, using extension identifiers from the Private Use space in the Extensions Supported registry (Table 3), with a value agreed upon (a priori) between the participants. DLEP extensions using the Private Use numbering space are commonly referred to as Experiments.

Multiple experiments MAY be announced in the Session Initialization Messages. However, use of multiple experiments in a single session could lead to interoperability issues or unexpected results (e.g., clashes of experimental Signals, Messages, Data Items and/or status code types), and is therefore discouraged. It is left to implementations to determine the correct processing path (e.g., a

decision on whether to terminate the session, or to establish a precedence of the conflicting definitions) if such conflicts arise.

## 10. Scalability

The protocol is intended to support thousands of destinations on a given modem/router pair. At large scale, implementations should consider employing techniques to prevent flooding its peer with a large number of Messages in a short time. For example, a dampening algorithm could be employed to prevent a flapping device from generating a large number of Destination Up/Destination Down Messages.

Also, use of techniques such as a hysteresis can lessen the impact of rapid, minor fluctuations in link quality. The specific algorithms for handling flapping destinations and minor changes in link quality are outside the scope of this specification.

## 11. DLEP Signal and Message Structure

DLEP defines two protocol units used in two different ways: Signals and Messages. Signals are only used in the Discovery mechanism and are carried in UDP datagrams. Messages are used bidirectionally over a TCP connection between the participants, in the Session Initialization, In-Session and Session Termination states.

Both Signals and Messages consist of a Header followed by an unordered list of Data Items. Headers consist of Type and Length information, while Data Items are encoded as TLV (Type-Length-Value) structures. In this document, the Data Items following a Signal or Message Header are described as being 'contained in' the Signal or Message.

There is no restriction on the order of Data Items following a Header, and the acceptability of duplicate Data Items is defined by the definition of the Signal or Message declared by the type in the Header.

All integers in Header fields and values MUST be in network byte-order.

### 11.1. DLEP Signal Header

The DLEP Signal Header contains the following fields:

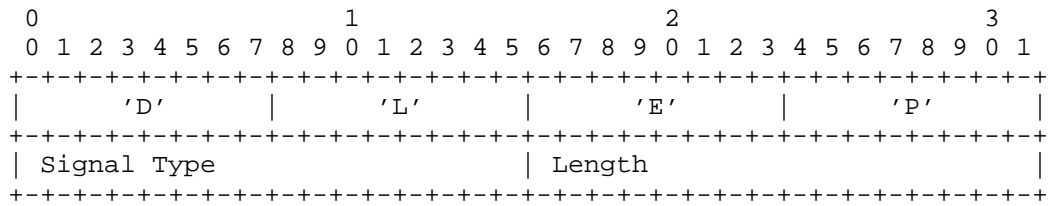


Figure 3: DLEP Signal Header

"DLEP": Every Signal MUST start with the characters: U+0044, U+004C, U+0045, U+0050.

Signal Type: A 16-bit unsigned integer containing one of the DLEP Signal Type values defined in this document.

Length: The length in octets, expressed as a 16-bit unsigned integer, of all of the DLEP Data Items contained in this Signal. This length MUST NOT include the length of the Signal Header itself.

The DLEP Signal Header is immediately followed by zero or more DLEP Data Items, encoded in TLVs, as defined in this document.

## 11.2. DLEP Message Header

The DLEP Message Header contains the following fields:

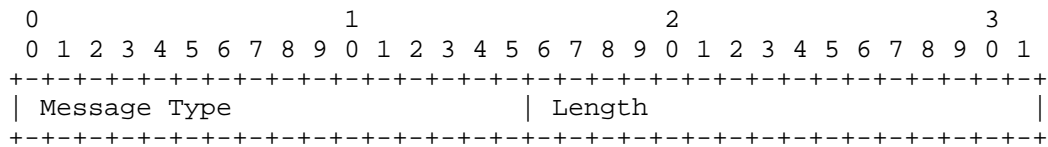


Figure 4: DLEP Message Header

Message Type: A 16-bit unsigned integer containing one of the DLEP Message Type values defined in this document.

Length: The length in octets, expressed as a 16-bit unsigned integer, of all of the DLEP Data Items contained in this Message. This length MUST NOT include the length of the Message Header itself.

The DLEP Message Header is immediately followed by zero or more DLEP Data Items, encoded in TLVs, as defined in this document.

### 11.3. DLEP Generic Data Item

All DLEP Data Items contain the following fields:

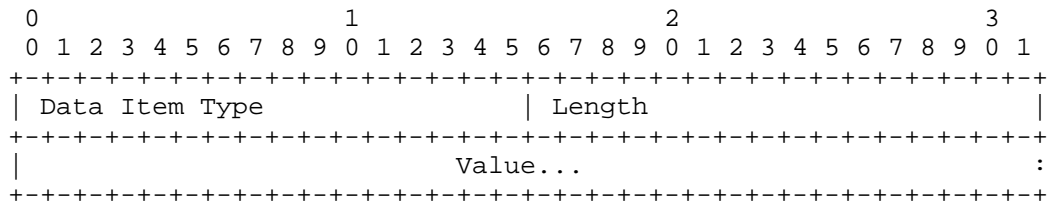


Figure 5: DLEP Generic Data Item

**Data Item Type:** A 16-bit unsigned integer field specifying the type of Data Item being sent.

**Length:** The length in octets, expressed as a 16-bit unsigned integer, of the Value field of the Data Item. This length **MUST** NOT include the length of the Data Item Type and Length fields.

**Value:** A field of <Length> octets, which contains data specific to a particular Data Item.

## 12. DLEP Signals and Messages

### 12.1. General Processing Rules

If an unrecognized, or unexpected Signal is received, or a received Signal contains unrecognized, invalid, or disallowed duplicate Data Items, the receiving implementation **MUST** ignore the Signal.

If a Signal is received with a TTL value that is NOT equal to 255, the receiving implementation **MUST** ignore the Signal.

If an unrecognized Message is received, the receiving implementation **MUST** issue a Session Termination Message (Section 12.9) containing a Status Data Item (Section 13.1) with status code set to 128 'Unknown Message', see Table 2, and transition to the Session Termination state.

If an unexpected Message is received, the receiving implementation **MUST** issue a Session Termination Message containing a Status Data Item with status code set to 129 'Unexpected Message', and transition to the Session Termination state.

If a received Message contains unrecognized, invalid, or disallowed duplicate Data Items, the receiving implementation **MUST** issue a



Session Termination Message containing a Status Data Item with status code set to 130 'Invalid Data', and transition to the Session Termination state.

If a packet in the TCP stream is received with a TTL value other than 255, the receiving implementation MUST immediately transition to the Session Reset state.

Prior to the exchange of Destination Up (Section 12.11) and Destination Up Response (Section 12.12) Messages, or Destination Announce (Section 12.13) and Destination Announce Response (Section 12.14) Messages, no Messages concerning a destination may be sent. An implementation receiving any Message with such an unannounced destination MUST terminate the session by issuing a Session Termination Message containing a Status Data Item with status code set to 131 'Invalid Destination', and transition to the Session Termination state.

After exchanging Destination Down (Section 12.15) and Destination Down Response (Section 12.16) Messages, no Messages concerning a destination may be sent until a new Destination Up or Destination Announce Message is sent. An implementation receiving a Message about a destination previously announced as 'down' MUST terminate the session by issuing a Session Termination Message containing a Status Data Item with status code set to 131 'Invalid Destination', and transition to the Session Termination state.

#### 12.2. Status code processing

The behavior of a DLEP participant receiving a Message containing a Status Data Item (Section 13.1) is defined by the failure mode associated with the value of the status code field, see Table 2. All status code values less than 100 have a failure mode of 'Continue', all other status codes have a failure mode of 'Terminate'.

A DLEP participant receiving any Message apart from Session Termination Message (Section 12.9) containing a Status Data Item with a status code value with failure mode 'Terminate' MUST immediately issue a Session Termination Message echoing the received Status Data Item, and then transition to the Session Termination state.

A DLEP participant receiving a Message containing a Status Data Item with a status code value with failure mode 'Continue' can continue normal operation of the session.

### 12.3. Peer Discovery Signal

A Peer Discovery Signal SHOULD be sent by a DLEP router to discover DLEP modems in the network, see Section 7.1.

A Peer Discovery Signal MUST be encoded within a UDP packet. The destination MUST be set to the DLEP well-known address and port number. For routers supporting both IPv4 and IPv6 DLEP operation, it is RECOMMENDED that IPv6 be selected as the transport. The source IP address MUST be set to the router IP address associated with the DLEP interface. There is no DLEP-specific restriction on source port.

To construct a Peer Discovery Signal, the Signal Type value in the Signal Header is set to 1 (see Signal Type Registration (Section 15.2)).

The Peer Discovery Signal MAY contain a Peer Type Data Item (Section 13.4).

### 12.4. Peer Offer Signal

A Peer Offer Signal MUST be sent by a DLEP modem in response to a properly formatted and addressed Peer Discovery Signal (Section 12.3).

A Peer Offer Signal MUST be encoded within a UDP packet. The IP source and destination fields in the packet MUST be set by swapping the values received in the Peer Discovery Signal. The Peer Offer Signal completes the discovery process, see Section 7.1.

To construct a Peer Offer Signal, the Signal Type value in the Signal Header is set to 2 (see Signal Type Registration (Section 15.2)).

The Peer Offer Signal MAY contain a Peer Type Data Item (Section 13.4).

The Peer Offer Signal MAY contain one or more of any of the following Data Items, with different values:

- o IPv4 Connection Point (Section 13.2)
- o IPv6 Connection Point (Section 13.3)

The IP Connection Point Data Items indicate the unicast address the router MUST use when connecting the DLEP TCP session.

## 12.5. Session Initialization Message

A Session Initialization Message MUST be sent by a DLEP router as the first Message of the DLEP TCP session. It is sent by the router after a TCP connect to an address/port combination that was obtained either via receipt of a Peer Offer, or from a priori configuration.

To construct a Session Initialization Message, the Message Type value in the Message Header is set to 1 (see Message Type Registration (Section 15.3)).

The Session Initialization Message MUST contain one of each of the following Data Items:

- o Heartbeat Interval Data Item (Section 13.5)
- o Peer Type (Section 13.4)

The Session Initialization Message MUST contain an Extensions Supported Data Item (Section 13.6), if DLEP extensions are supported.

The Session Initialization Message MAY contain one or more of each of the following Data Items, with different values, and the data item Add flag set to 1:

- o IPv4 Address (Section 13.8)
- o IPv6 Address (Section 13.9)
- o IPv4 Attached Subnet (Section 13.10)
- o IPv6 Attached Subnet (Section 13.11)

If any optional extensions are supported by the implementation, they MUST be enumerated in the Extensions Supported Data Item. If an Extensions Supported Data Item does not exist in a Session Initialization Message, the modem MUST conclude that there is no support for extensions in the router.

DLEP Heartbeats are not started until receipt of the Session Initialization Response Message (Section 12.6), and therefore implementations MUST use their own timeout heuristics for this Message.

As an exception to the general rule governing an implementation receiving an unrecognized Data Item in a Message, see Section 12.1, if a Session Initialization Message contains one or more Extension Supported Data Items announcing support for extensions that the

implementation does not recognize, then the implementation MAY ignore Data Items it does not recognize.

## 12.6. Session Initialization Response Message

A Session Initialization Response Message MUST be sent by a DLEP modem in response to a received Session Initialization Message (Section 12.5).

To construct a Session Initialization Response Message, the Message Type value in the Message Header is set to 2 (see Message Type Registration (Section 15.3)).

The Session Initialization Response Message MUST contain one of each of the following Data Items:

- o Status (Section 13.1)
- o Peer Type (Section 13.4)
- o Heartbeat Interval (Section 13.5)
- o Maximum Data Rate (Receive) (Section 13.12)
- o Maximum Data Rate (Transmit) (Section 13.13)
- o Current Data Rate (Receive) (Section 13.14)
- o Current Data Rate (Transmit) (Section 13.15)
- o Latency (Section 13.16)

The Session Initialization Response Message MUST contain one of each of the following Data Items, if the Data Item will be used during the lifetime of the session:

- o Resources (Section 13.17)
- o Relative Link Quality (Receive) (Section 13.18)
- o Relative Link Quality (Transmit) (Section 13.19)
- o Maximum Transmission Unit (MTU) (Section 13.20)

The Session Initialization Response Message MUST contain an Extensions Supported Data Item (Section 13.6), if DLEP extensions are supported.

The Session Initialization Response Message MAY contain one or more of each of the following Data Items, with different values, and the data item Add flag set to 1:

- o IPv4 Address (Section 13.8)
- o IPv6 Address (Section 13.9)
- o IPv4 Attached Subnet (Section 13.10)
- o IPv6 Attached Subnet (Section 13.11)

The Session Initialization Response Message completes the DLEP session establishment; the modem should transition to the In-Session state when the Message is sent, and the router should transition to the In-Session state upon receipt of an acceptable Session Initialization Response Message.

All supported metric Data Items MUST be included in the Session Initialization Response Message, with default values to be used on a session-wide basis. This can be viewed as the modem 'declaring' all supported metrics at DLEP session initialization. Receipt of any further DLEP Message containing a metric Data Item not included in the Session Initialization Response Message MUST be treated as an error, resulting in the termination of the DLEP session between router and modem.

If any optional extensions are supported by the modem, they MUST be enumerated in the Extensions Supported Data Item. If an Extensions Supported Data Item does not exist in a Session Initialization Response Message, the router MUST conclude that there is no support for extensions in the modem.

After the Session Initialization/Session Initialization Response Messages have been successfully exchanged, implementations MUST only use extensions that are supported by both DLEP participants, see Section 7.2.

## 12.7. Session Update Message

A Session Update Message MAY be sent by a DLEP participant to indicate local Layer 3 address changes, or metric changes on a session-wide basis.

To construct a Session Update Message, the Message Type value in the Message Header is set to 3 (see Message Type Registration (Section 15.3)).

The Session Update Message MAY contain one or more of each of the following Data Items, with different values:

- o IPv4 Address (Section 13.8)
- o IPv6 Address (Section 13.9)
- o IPv4 Attached Subnet (Section 13.10)
- o IPv6 Attached Subnet (Section 13.11)

When sent by a modem, the Session Update Message MAY contain one of each of the following Data Items:

- o Maximum Data Rate (Receive) (Section 13.12)
- o Maximum Data Rate (Transmit) (Section 13.13)
- o Current Data Rate (Receive) (Section 13.14)
- o Current Data Rate (Transmit) (Section 13.15)
- o Latency (Section 13.16)

When sent by a modem, the Session Update Message MAY contain one of each of the following Data Items, if the Data Item is in use by the session:

- o Resources (Section 13.17)
- o Relative Link Quality (Receive) (Section 13.18)
- o Relative Link Quality (Transmit) (Section 13.19)
- o Maximum Transmission Unit (MTU) (Section 13.20)

If metrics are supplied with the Session Update Message (e.g., Maximum Data Rate), these metrics are considered to be session-wide, and therefore MUST be applied to all destinations in the information base associated with the DLEP session. This includes destinations for which metrics may have been stored based on received Destination Update messages.

It should be noted that Session Update Messages can be sent by both routers and modems. For example, addition of an IPv4 address on the router MAY prompt a Session Update Message to its attached modems. Also, for example, a modem that changes its Maximum Data Rate

(Receive) for all destinations MAY reflect that change via a Session Update Message to its attached router(s).

Concerning Layer 3 addresses and subnets: If the modem is capable of understanding and forwarding this information (via mechanisms not defined by DLEP), the update would prompt any remote DLEP-enabled modems to issue a Destination Update Message (Section 12.17) to their local routers with the new (or deleted) addresses and subnets.

#### 12.8. Session Update Response Message

A Session Update Response Message MUST be sent by a DLEP participant when a Session Update Message (Section 12.7) is received.

To construct a Session Update Response Message, the Message Type value in the Message Header is set to 4 (see Message Type Registration (Section 15.3)).

The Session Update Response Message MUST contain a Status Data Item (Section 13.1).

#### 12.9. Session Termination Message

When a DLEP participant determines the DLEP session needs to be terminated, the participant MUST send (or attempt to send) a Session Termination Message.

To construct a Session Termination Message, the Message Type value in the Message Header is set to 5 (see Message Type Registration (Section 15.3)).

The Session Termination Message MUST contain Status Data Item (Section 13.1).

It should be noted that Session Termination Messages can be sent by both routers and modems.

#### 12.10. Session Termination Response Message

A Session Termination Response Message MUST be sent by a DLEP participant when a Session Termination Message (Section 12.9) is received.

To construct a Session Termination Response Message, the Message Type value in the Message Header is set to 6 (see Message Type Registration (Section 15.3)).

There are no valid Data Items for the Session Termination Response Message.

Receipt of a Session Termination Response Message completes the tear-down of the DLEP session, see Section 7.4.

#### 12.11. Destination Up Message

Destination Up Messages MAY be sent by a modem to inform its attached router of the presence of a new reachable destination.

To construct a Destination Up Message, the Message Type value in the Message Header is set to 7 (see Message Type Registration (Section 15.3)).

The Destination Up Message MUST contain a MAC Address Data Item (Section 13.7).

The Destination Up Message SHOULD contain one or more of each of the following Data Items, with different values:

- o IPv4 Address (Section 13.8)
- o IPv6 Address (Section 13.9)

The Destination Up Message MAY contain one of each of the following Data Items:

- o Maximum Data Rate (Receive) (Section 13.12)
- o Maximum Data Rate (Transmit) (Section 13.13)
- o Current Data Rate (Receive) (Section 13.14)
- o Current Data Rate (Transmit) (Section 13.15)
- o Latency (Section 13.16)

The Destination Up Message MAY contain one of each of the following Data Items, if the Data Item is in use by the session:

- o Resources (Section 13.17)
- o Relative Link Quality (Receive) (Section 13.18)
- o Relative Link Quality (Transmit) (Section 13.19)
- o Maximum Transmission Unit (MTU) (Section 13.20)



The Destination Up Message MAY contain one or more of each of the following Data Items, with different values:

- o IPv4 Attached Subnet (Section 13.10)
- o IPv6 Attached Subnet (Section 13.11)

A router receiving a Destination Up Message allocates the necessary resources, creating an entry in the information base with the specifics (MAC Address, Latency, Data Rate, etc.) of the destination. The information about this destination will persist in the router's information base until a Destination Down Message (Section 12.15) is received, indicating that the modem has lost contact with the remote node, or the implementation transitions to the Session Termination state.

#### 12.12. Destination Up Response Message

A router MUST send a Destination Up Response Message when a Destination Up Message (Section 12.11) is received.

To construct a Destination Up Response Message, the Message Type value in the Message Header is set to 8 (see Message Type Registration (Section 15.3)).

The Destination Up Response Message MUST contain one of each of the following Data Items:

- o MAC Address (Section 13.7)
- o Status (Section 13.1)

A router that wishes to receive further information concerning the destination identified in the corresponding Destination Up Message MUST set the status code of the included Status Data Item to 0 'Success', see Table 2.

If the router has no interest in the destination identified in the corresponding Destination Up Message, then it MAY set the status code of the included Status Data Item to 1 'Not Interested'.

A modem receiving a Destination Up Response Message containing a Status Data Item with status code of any value other than 0 'Success' MUST NOT send further Destination messages about the destination, e.g. Destination Down (Section 12.15) or Destination Update (Section 12.17) with the same MAC address.

### 12.13. Destination Announce Message

Usually a modem will discover the presence of one or more remote router/modem pairs and announce each destination's arrival by sending a corresponding Destination Up Message (Section 12.11) to the router. However, there may be times when a router wishes to express an interest in a destination that has yet to be announced, typically a multicast destination. Destination Announce Messages MAY be sent by a router to announce such an interest.

A Destination Announce Message MAY also be sent by a router to request information concerning a destination in which it has previously declined interest, via the 1 'Not Interested' status code in a Destination Up Response Message (Section 12.12), see Table 2, or declared as 'down', via the Destination Down Message (Section 12.15).

To construct a Destination Announce Message, the Message Type value in the Message Header is set to 9 (see Message Type Registration (Section 15.3)).

The Destination Announce Message MUST contain a MAC Address Data Item (Section 13.7).

The Destination Announce Message MAY contain zero or more of the following Data Items, with different values:

- o IPv4 Address (Section 13.8)
- o IPv6 Address (Section 13.9)

One of the advantages of implementing DLEP is to leverage the modem's knowledge of the links between remote destinations allowing routers to avoid using probed neighbor discovery techniques, therefore modem implementations SHOULD announce available destinations via the Destination Up Message, rather than relying on Destination Announce Messages.

### 12.14. Destination Announce Response Message

A modem MUST send a Destination Announce Response Message when a Destination Announce Message (Section 12.13) is received.

To construct a Destination Announce Response Message, the Message Type value in the Message Header is set to 10 (see Message Type Registration (Section 15.3)).

The Destination Announce Response Message MUST contain one of each of the following Data Items:

- o MAC Address (Section 13.7)
- o Status (Section 13.1)

The Destination Announce Response Message MAY contain one or more of each of the following Data Items, with different values:

- o IPv4 Address (Section 13.8)
- o IPv6 Address (Section 13.9)
- o IPv4 Attached Subnet (Section 13.10)
- o IPv6 Attached Subnet (Section 13.11)

The Destination Announce Response Message MAY contain one of each of the following Data Items:

- o Maximum Data Rate (Receive) (Section 13.12)
- o Maximum Data Rate (Transmit) (Section 13.13)
- o Current Data Rate (Receive) (Section 13.14)
- o Current Data Rate (Transmit) (Section 13.15)
- o Latency (Section 13.16)

The Destination Announce Response Message MAY contain one of each of the following Data Items, if the Data Item is in use by the session:

- o Resources (Section 13.17)
- o Relative Link Quality (Receive) (Section 13.18)
- o Relative Link Quality (Transmit) (Section 13.19)
- o Maximum Transmission Unit (MTU) (Section 13.20)

If a modem is unable to report information immediately about the requested information, if the destination is not currently reachable, for example, the status code in the Status Data Item MUST be set to 2 'Request Denied', see Table 2.

After sending a Destination Announce Response Message containing a Status Data Item with status code of 0 'Success', a modem then announces changes to the link to the destination via Destination Update Messages.

When a successful Destination Announce Response Message is received, the router should add knowledge of the available destination to its information base.

#### 12.15. Destination Down Message

A modem **MUST** send a Destination Down Message to report when a destination (a remote node or a multicast group) is no longer reachable.

A router **MAY** send a Destination Down Message to report when it no longer requires information concerning a destination.

To construct a Destination Down Message, the Message Type value in the Message Header is set to 11 (see Message Type Registration (Section 15.3)).

The Destination Down Message **MUST** contain a MAC Address Data Item (Section 13.7).

It should be noted that both modem and router may send a Destination Down Message to their peer, regardless of which participant initially indicated the destination to be 'up'.

#### 12.16. Destination Down Response Message

A Destination Down Response **MUST** be sent by the recipient of a Destination Down Message (Section 12.15) to confirm that the relevant data concerning the destination has been removed from the information base.

To construct a Destination Down Response Message, the Message Type value in the Message Header is set to 12 (see Message Type Registration (Section 15.3)).

The Destination Down Response Message **MUST** contain one of each of the following Data Items:

- o MAC Address (Section 13.7)
- o Status (Section 13.1)

#### 12.17. Destination Update Message

A modem **SHOULD** send the Destination Update Message when it detects some change in the information base for a given destination (remote node or multicast group). Some examples of changes that would prompt a Destination Update Message are:

- o Change in link metrics (e.g., Data Rates)
- o Layer 3 addressing change

To construct a Destination Update Message, the Message Type value in the Message Header is set to 13 (see Message Type Registration (Section 15.3)).

The Destination Update Message **MUST** contain a MAC Address Data Item (Section 13.7).

The Destination Update Message **MAY** contain one of each of the following Data Items:

- o Maximum Data Rate (Receive) (Section 13.12)
- o Maximum Data Rate (Transmit) (Section 13.13)
- o Current Data Rate (Receive) (Section 13.14)
- o Current Data Rate (Transmit) (Section 13.15)
- o Latency (Section 13.16)

The Destination Update Message **MAY** contain one of each of the following Data Items, if the Data Item is in use by the session:

- o Resources (Section 13.17)
- o Relative Link Quality (Receive) (Section 13.18)
- o Relative Link Quality (Transmit) (Section 13.19)
- o Maximum Transmission Unit (MTU) (Section 13.20)

The Destination Update Message **MAY** contain one or more of each of the following Data Items, with different values:

- o IPv4 Address (Section 13.8)
- o IPv6 Address (Section 13.9)
- o IPv4 Attached Subnet (Section 13.10)
- o IPv6 Attached Subnet (Section 13.11)

Metrics supplied in this message overwrite metrics provided in a previously received Session or Destination Up Messages.

It should be noted that this Message has no corresponding response.

#### 12.18. Link Characteristics Request Message

The Link Characteristics Request Message MAY be sent by a router to request that the modem initiate changes for specific characteristics of the link. The request can reference either a real destination (e.g., a remote node), or a logical destination (e.g., a multicast group) within the network.

To construct a Link Characteristics Request Message, the Message Type value in the Message Header is set to 14 (see Message Type Registration (Section 15.3)).

The Link Characteristics Request Message MUST contain one of the following Data Items:

- o MAC Address (Section 13.7)

The Link Characteristics Request Message MUST contain at least one of each of the following Data Items:

- o Current Data Rate (Receive) (Section 13.14)
- o Current Data Rate (Transmit) (Section 13.15)
- o Latency (Section 13.16)

The Link Characteristics Request Message MAY contain either a Current Data Rate (CDRR or CDRT) Data Item to request a different datarate than is currently allocated, a Latency Data Item to request that traffic delay on the link not exceed the specified value, or both.

The router sending a Link Characteristics Request Message should be aware that a request may take an extended period of time to complete.

#### 12.19. Link Characteristics Response Message

A modem MUST send a Link Characteristics Response Message when a Link Characteristics Request Message (Section 12.18) is received.

To construct a Link Characteristics Response Message, the Message Type value in the Message Header is set to 15 (see Message Type Registration (Section 15.3)).

The Link Characteristics Response Message MUST contain one of each of the following Data Items:

- o MAC Address (Section 13.7)
- o Status (Section 13.1)

The Link Characteristics Response Message SHOULD contain one of each of the following Data Items:

- o Maximum Data Rate (Receive) (Section 13.12)
- o Maximum Data Rate (Transmit) (Section 13.13)
- o Current Data Rate (Receive) (Section 13.14)
- o Current Data Rate (Transmit) (Section 13.15)
- o Latency (Section 13.16)

The Link Characteristics Response Message MAY contain one of each of the following Data Items, if the Data Item is in use by the session:

- o Resources (Section 13.17)
- o Relative Link Quality (Receive) (Section 13.18)
- o Relative Link Quality (Transmit) (Section 13.19)
- o Maximum Transmission Unit (MTU) (Section 13.20)

The Link Characteristics Response Message MUST contain a complete set of metric Data Items, referencing all metrics declared in the Session Initialization Response Message (Section 12.6). The values in the metric Data Items in the Link Characteristics Response Message MUST reflect the link characteristics after the request has been processed.

If an implementation is not able to alter the characteristics of the link in the manner requested, then the status code of the Status Data Item MUST be set to 2 'Request Denied', see Table 2.

#### 12.20. Heartbeat Message

A Heartbeat Message MUST be sent by a DLEP participant every N milliseconds, where N is defined in the Heartbeat Interval Data Item (Section 13.5) of the Session Initialization Message (Section 12.5) or Session Initialization Response Message (Section 12.6).

To construct a Heartbeat Message, the Message Type value in the Message Header is set to 16 (see Message Type Registration (Section 15.3)).

There are no valid Data Items for the Heartbeat Message.

The Message is used by DLEP participants to detect when a DLEP session peer (either the modem or the router) is no longer communicating, see Section 7.3.1.

### 13. DLEP Data Items

The core DLEP Data Items are:

Type Code	Description
0	Reserved
1	Status (Section 13.1)
2	IPv4 Connection Point (Section 13.2)
3	IPv6 Connection Point (Section 13.3)
4	Peer Type (Section 13.4)
5	Heartbeat Interval (Section 13.5)
6	Extensions Supported (Section 13.6)
7	MAC Address (Section 13.7)
8	IPv4 Address (Section 13.8)
9	IPv6 Address (Section 13.9)
10	IPv4 Attached Subnet (Section 13.10)
11	IPv6 Attached Subnet (Section 13.11)
12	Maximum Data Rate (Receive) (MDRR) (Section 13.12)
13	Maximum Data Rate (Transmit) (MDRT) (Section 13.13)
14	Current Data Rate (Receive) (CDRR) (Section 13.14)
15	Current Data Rate (Transmit) (CDRT) (Section 13.15)
16	Latency (Section 13.16)
17	Resources (RES) (Section 13.17)
18	Relative Link Quality (Receive) (RLQR) (Section 13.18)
19	Relative Link Quality (Transmit) (RLQT) (Section 13.19)
20	Maximum Transmission Unit (MTU) (Section 13.20)
21-65407	Reserved for future extensions
65408-65534	Private Use. Available for experiments
65535	Reserved

Table 1: DLEP Data Item types



### 13.1. Status

For the Session Termination Message (Section 12.9), the Status Data Item indicates a reason for the termination. For all response Messages, the Status Data Item is used to indicate the success or failure of the previously received Message.

The Status Data Item includes an optional Text field that can be used to provide a textual description of the status. The use of the Text field is entirely up to the receiving implementation, e.g., it could be output to a log file or discarded. If no Text field is supplied with the Status Data Item, the Length field **MUST** be set to 1.

The Status Data Item contains the following fields:

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
| Data Item Type | Length |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Code          | Text... |
+-----+-----+-----+-----+-----+-----+-----+

```

Data Item Type: 1

Length: 1 + Length of text, in octets

Status Code: One of the codes defined in Table 2 below.

Text: UTF-8 encoded string of UNICODE [RFC3629] characters, describing the cause, used for implementation defined purposes. Since this field is used for description, implementations **SHOULD** limit characters in this field to printable characters.

An implementation **MUST NOT** assume the Text field is a NUL-terminated string of printable characters.

Status Code	Failure Mode	Description	Reason
0	Continue	Success	The Message was processed successfully.
1	Continue	Not Interested	The receiver is not interested in this Message subject, e.g. in a Destination Up

			Response Message (Section 12.12) to indicate no further Messages about the destination.
2	Continue	Request Denied	The receiver refuses to complete the request.
3	Continue	Inconsistent Data	One or more Data Items in the Message describe a logically inconsistent state in the network. For example, in the Destination Up Message (Section 12.11) when an announced subnet clashes with an existing destination subnet.
4-111	Continue	<Reserved>	Reserved for future extensions.
112-127	Continue	<Private Use>	Available for experiments.
128	Terminate	Unknown Message	The Message was not recognized by the implementation.
129	Terminate	Unexpected Message	The Message was not expected while the device was in the current state, e.g., a Session Initialization Message (Section 12.5) in the In-Session state.
130	Terminate	Invalid Data	One or more Data Items in the Message are invalid, unexpected or incorrectly duplicated.
131	Terminate	Invalid Destination	The destination included in the Message does not match a previously announced

			destination. For example, in the Link Characteristic Response Message (Section 12.19).
132	Terminate	Timed Out	The session has timed out.
133-239	Terminate	<Reserved>	Reserved for future extensions.
240-254	Terminate	<Private Use>	Available for experiments.
255	Terminate	<Reserved>	Reserved.

Table 2: DLEP Status Codes

### 13.2. IPv4 Connection Point

The IPv4 Connection Point Data Item indicates the IPv4 address and, optionally, the TCP port number on the modem available for connections. If provided, the router **MUST** use this information to initiate the TCP connection to the modem.

The IPv4 Connection Point Data Item contains the following fields:

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1								
Data Item Type																				Length																			
Flags										IPv4 Address...										:																			
: ...cont.										TCP Port Number (optional)																													

Data Item Type: 2

Length: 5 (or 7 if TCP Port included)

Flags: Flags field, defined below.

IPv4 Address: The IPv4 address listening on the modem.

TCP Port Number: TCP Port number on the modem.

If the Length field is 7, the port number specified **MUST** be used to establish the TCP session. If the TCP Port Number is omitted, i.e.

the Length field is 5, the router MUST use the DLEP well-known port number (Section 15.14) to establish the TCP connection.

The Flags field is defined as:

```

  0 1 2 3 4 5 6 7
+-----+
| Reserved |T|
+-----+
```

T: Use TLS flag, indicating whether the TCP connection to the given address and port requires the use of TLS [RFC5246] (1), or not (0).

Reserved: MUST be zero. Left for future assignment.

### 13.3. IPv6 Connection Point

The IPv6 Connection Point Data Item indicates the IPv6 address and, optionally, the TCP port number on the modem available for connections. If provided, the router MUST use this information to initiate the TCP connection to the modem.

The IPv6 Connection Point Data Item contains the following fields:

```

      0                      1                      2                      3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
| Data Item Type | Length |
+-----+-----+-----+-----+-----+-----+-----+-----+
|  Flags      | IPv6 Address |
+-----+-----+-----+-----+-----+-----+-----+-----+
: IPv6 Address :
+-----+-----+-----+-----+-----+-----+-----+-----+
: IPv6 Address :
+-----+-----+-----+-----+-----+-----+-----+-----+
: IPv6 Address :
+-----+-----+-----+-----+-----+-----+-----+-----+
: ...cont. | TCP Port Number (optional) |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

Data Item Type: 3

Length: 17 (or 19 if TCP Port included)

Flags: Flags field, defined below.

IPv6 Address: The IPv6 address listening on the modem.

TCP Port Number: TCP Port number on the modem.

If the Length field is 19, the port number specified MUST be used to establish the TCP session. If the TCP Port Number is omitted, i.e. the Length field is 17, the router MUST use the DLEP well-known port number (Section 15.14) to establish the TCP connection.

The Flags field is defined as:

```
  0 1 2 3 4 5 6 7
+---+---+---+---+---+
|   Reserved   |T|
+---+---+---+---+---+
```

T: Use TLS flag, indicating whether the TCP connection to the given address and port requires the use of TLS [RFC5246] (1), or not (0).

Reserved: MUST be zero. Left for future assignment.

#### 13.4. Peer Type

The Peer Type Data Item is used by the router and modem to give additional information as to its type and the properties of the over-the-air control-plane.

With some devices, access to the shared RF medium is strongly controlled. One example of this would be satellite modems - where protocols, proprietary in nature, have been developed to insure a given modem has authorization to connect to the shared medium. Another example of this class of modems is governmental/military devices, where elaborate mechanisms have been developed to ensure that only authorized devices can connect to the shared medium. Contrasting with the above, there are modems where no such access control is used. An example of this class of modem would be one that supports the 802.11 ad-hoc mode of operation. The Secured Medium flag is used to indicate if access control is in place.

The Peer Type Data Item includes a textual description of the peer that is envisioned to be used for informational purposes (e.g., as output in a display command).

The Peer Type Data Item contains the following fields:

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Data Item Type               | Length                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Flags           | Description...                                     :
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Data Item Type: 4

Length: 1 + Length of Peer Type string, in octets.

Flags: Flags field, defined below.

Description: UTF-8 encoded string of UNICODE [RFC3629] characters.

For example, a satellite modem might set this variable to "Satellite terminal". Since this Data Item is intended to provide additional information for display commands, sending implementations SHOULD limit the data to printable characters.

An implementation MUST NOT assume the Description field is a NUL-terminated string of printable characters.

The Flags field is defined as:

```

      0 1 2 3 4 5 6 7
+---+---+---+---+---+---+---+
| Reserved   |S|
+---+---+---+---+---+---+---+

```

S: Secured Medium flag, used by a modem to indicate if the shared RF medium implements access control (1), or not (0). The Secured Medium flag only has meaning in Signals and Messages sent by a modem.

Reserved: MUST be zero. Left for future assignment.

### 13.5. Heartbeat Interval

The Heartbeat Interval Data Item is used to specify a period in milliseconds for Heartbeat Messages (Section 12.20).

The Heartbeat Interval Data Item contains the following fields:

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
| Data Item Type                               | Length |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               Heartbeat Interval                               |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Data Item Type: 5

Length: 4

Heartbeat Interval: The interval in milliseconds, expressed as a 32-bit unsigned integer, for Heartbeat Messages. This value MUST NOT be 0.

As mentioned before, receipt of any valid DLEP Message MUST reset the heartbeat interval timer (e.g., valid DLEP Messages take the place of, and obviate the need for, additional Heartbeat Messages).

### 13.6. Extensions Supported

The Extensions Supported Data Item is used by the router and modem to negotiate additional optional functionality they are willing to support. The Extensions List is a concatenation of the types of each supported extension, found in the IANA DLEP Extensions repository. Each Extension Type definition includes which additional Signals and Data Items are supported.

The Extensions Supported Data Item contains the following fields:

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
| Data Item Type                               | Length |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Extensions List...                           :
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Data Item Type: 6

Length: Length of the extensions list in octets. This is twice (2x) the number of extensions.

Extension List: A list of extensions supported, identified by their 2-octet value as listed in the extensions registry.

### 13.7. MAC Address

The MAC Address Data Item contains the address of the destination on the remote node.

DLEP can support MAC addresses in either EUI-48 or EUI-64 format, with the restriction that all MAC addresses for a given DLEP session MUST be in the same format, and MUST be consistent with the MAC address format of the connected modem (e.g., if the modem is connected to the router with an EUI-48 MAC, all destination addresses via that modem MUST be expressed in EUI-48 format).

Examples of a virtual destination would be a multicast MAC address, or the broadcast MAC (FF:FF:FF:FF:FF:FF).

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
Data Item Type										Length																													
MAC Address																														:									
:	MAC Address														:	(if EUI-64 used)														:									

Data Item Type: 7

Length: 6 for EUI-48 format, or 8 for EUI-64 format

MAC Address: MAC Address of the destination.

### 13.8. IPv4 Address

When included in the Session Update Message, this Data Item contains the IPv4 address of the peer. When included in Destination Messages, this Data Item contains the IPv4 address of the destination. In either case, the Data Item also contains an indication of whether this is a new or existing address, or is a deletion of a previously known address.

The IPv4 Address Data Item contains the following fields:



```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Data Item Type                               | Length |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Flags           | IPv4 Address                :
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
:   ...cont.   |
+---+---+---+---+---+

```

Data Item Type: 8

Length: 5

Flags: Flags field, defined below.

IPv4 Address: The IPv4 address of the destination or peer.

The Flags field is defined as:

```

      0 1 2 3 4 5 6 7
+---+---+---+---+---+
| Reserved  | A |
+---+---+---+---+---+

```

A: Add/Drop flag, indicating whether this is a new or existing address (1), or a withdrawal of an address (0).

Reserved: MUST be zero. Reserved for future use.

#### 13.8.1. IPv4 Address Processing

Processing of the IPv4 Address Data Item MUST be done within the context of the DLEP Peer session on which it is presented.

The handling of erroneous or logically inconsistent conditions depends upon the type of the message that contains the data item:

If the containing message is a Session Message, e.g., Session Initialization Message (Section 12.5), or Session Update Message (Section 12.7), the receiver of inconsistent information MUST issue a Session Termination Message (Section 12.9) containing a Status Data Item (Section 13.1) with status code set to 130 'Invalid Data', and transition to the Session Termination state. Examples of such conditions are:

- o An address Drop operation referencing an address that is not associated with the peer in the current session.

- o An address Add operation referencing an address that has already been added to the peer in the current session.

If the containing message is a Destination Message, e.g., Destination Up Message (Section 12.11), or Destination Update Message (Section 12.17), the receiver of inconsistent information MAY issue the appropriate response message containing a Status Data Item, with status code set to 3 'Inconsistent Data', but MUST continue with session processing. Examples of such conditions are:

- o An address Add operation referencing an address that has already been added to the destination in the current session.
- o An address Add operation referencing an address that is associated with a different destination or the peer in the current session.
- o An address Add operation referencing an address that makes no sense, for example defined as not forwardable in [RFC6890].
- o An address Drop operation referencing an address that is not associated with the destination in the current session.

If no response message is appropriate, for example, the Destination Update Message, then the implementation MUST continue with session processing.

Modems that do not track IPv4 addresses MUST silently ignore IPv4 Address Data Items.

### 13.9. IPv6 Address

When included in the Session Update Message, this Data Item contains the IPv6 address of the peer. When included in Destination Messages, this Data Item contains the IPv6 address of the destination. In either case, the Data Item also contains an indication of whether this is a new or existing address, or is a deletion of a previously known address.

The IPv6 Address Data Item contains the following fields:

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Data Item Type                               | Length |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Flags           | IPv6 Address                :
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
:                               IPv6 Address                :
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
:                               IPv6 Address                :
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
:                               IPv6 Address                :
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
: IPv6 Address |
+---+---+---+---+---+

```

Data Item Type: 9

Length: 17

Flags: Flags field, defined below.

IPv6 Address: IPv6 Address of the destination or peer.

The Flags field is defined as:

```

      0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
| Reserved |A|
+---+---+---+---+---+---+

```

A: Add/Drop flag, indicating whether this is a new or existing address (1), or a withdrawal of an address (0).

Reserved: MUST be zero. Reserved for future use.

#### 13.9.1. IPv6 Address Processing

Processing of the IPv6 Address Data Item MUST be done within the context of the DLEP Peer session on which it is presented.

The handling of erroneous or logically inconsistent conditions depends upon the type of the message that contains the data item:

If the containing message is a Session Message, e.g., Session Initialization Message (Section 12.5), or Session Update Message (Section 12.7), the receiver of inconsistent information MUST issue a Session Termination Message (Section 12.9) containing a Status Data

Item (Section 13.1) with status code set to 130 'Invalid Data', and transition to the Session Termination state. Examples of such conditions are:

- o An address Drop operation referencing an address that is not associated with the peer in the current session.
- o An address Add operation referencing an address that has already been added to the peer in the current session.

If the containing message is a Destination Message, e.g., Destination Up Message (Section 12.11), or Destination Update Message (Section 12.17), the receiver of inconsistent information MAY issue the appropriate response message containing a Status Data Item, with status code set to 3 'Inconsistent Data', but MUST continue with session processing. Examples of such conditions are:

- o An address Add operation referencing an address that has already been added to the destination in the current session.
- o An address Add operation referencing an address that is associated with a different destination or the peer in the current session.
- o An address Add operation referencing an address that makes no sense, for example defined as not forwardable in [RFC6890].
- o An address Drop operation referencing an address that is not associated with the destination in the current session.

If no response message is appropriate, for example, the Destination Update Message, then the implementation MUST continue with session processing.

Modems that do not track IPv6 addresses MUST silently ignore IPv6 Address Data Items.

#### 13.10. IPv4 Attached Subnet

The DLEP IPv4 Attached Subnet allows a device to declare that it has an IPv4 subnet (e.g., a stub network) attached, that it has become aware of an IPv4 subnet being present at a remote destination, or that it has become aware of the loss of a subnet at the remote destination.

The DLEP IPv4 Attached Subnet Data Item contains the following fields:

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Data Item Type                                     | Length |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|  Flags                                     | IPv4 Attached Subnet |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
:  ...cont. | Prefix Length |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Data Item Type: 10

Length: 6

Flags: Flags field, defined below.

IPv4 Subnet: The IPv4 subnet reachable at the destination.

Prefix Length: Length of the prefix (0-32) for the IPv4 subnet. A prefix length outside the specified range MUST be considered as invalid.

The Flags field is defined as:

```

      0 1 2 3 4 5 6 7
+---+---+---+---+---+---+
|  Reserved  | A |
+---+---+---+---+---+---+

```

A: Add/Drop flag, indicating whether this is a new or existing subnet address (1), or a withdrawal of a subnet address (0).

Reserved: MUST be zero. Reserved for future use.

#### 13.10.1. IPv4 Attached Subnet Processing

Processing of the IPv4 Attached Subnet Data Item MUST be done within the context of the DLEP Peer session on which it is presented.

If the containing message is a Session Message, e.g., Session Initialization Message (Section 12.5), or Session Update Message (Section 12.7), the receiver of inconsistent information MUST issue a Session Termination Message (Section 12.9) containing a Status Data Item (Section 13.1) with status code set to 130 'Invalid Data', and transition to the Session Termination state. Examples of such conditions are:

- o A subnet Drop operation referencing a subnet that is not associated with the peer in the current session.
- o A subnet Add operation referencing a subnet that has already been added to the peer in the current session.

If the containing message is a Destination Message, e.g., Destination Up Message (Section 12.11), or Destination Update Message (Section 12.17), the receiver of inconsistent information MAY issue the appropriate response message containing a Status Data Item, with status code set to 3 'Inconsistent Data', but MUST continue with session processing. Examples of such conditions are:

- o A subnet Add operation referencing a subnet that has already been added to the destination in the current session.
- o A subnet Add operation referencing a subnet that is associated with a different destination in the current session.
- o An subnet Add operation referencing an subnet that makes no sense, for example defined as not forwardable in [RFC6890].
- o A subnet Drop operation referencing a subnet that is not associated with the destination in the current session.

If no response message is appropriate, for example, the Destination Update Message, then the implementation MUST continue with session processing.

Modems that do not track IPv4 subnets MUST silently ignore IPv4 Attached Subnet Data Items.

### 13.11. IPv6 Attached Subnet

The DLEP IPv6 Attached Subnet allows a device to declare that it has an IPv6 subnet (e.g., a stub network) attached, that it has become aware of an IPv6 subnet being present at a remote destination, or that it has become aware of the loss of a subnet at the remote destination.

The DLEP IPv6 Attached Subnet Data Item contains the following fields:

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
| Data Item Type                                     | Length |
+-----+-----+-----+-----+-----+-----+-----+-----+
|  Flags           |           IPv6 Attached Subnet           :
+-----+-----+-----+-----+-----+-----+-----+-----+
:                               IPv6 Attached Subnet           :
+-----+-----+-----+-----+-----+-----+-----+-----+
:                               IPv6 Attached Subnet           :
+-----+-----+-----+-----+-----+-----+-----+-----+
:                               IPv6 Attached Subnet           :
+-----+-----+-----+-----+-----+-----+-----+-----+
:   ...cont.   | Prefix Len.   |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Data Item Type: 11

Length: 18

Flags: Flags field, defined below.

IPv6 Attached Subnet: The IPv6 subnet reachable at the destination.

Prefix Length: Length of the prefix (0-128) for the IPv6 subnet. A prefix length outside the specified range MUST be considered as invalid.

The Flags field is defined as:

```

      0 1 2 3 4 5 6 7
+-----+-----+-----+
| Reserved | A |
+-----+-----+-----+

```

A: Add/Drop flag, indicating whether this is a new or existing subnet address (1), or a withdrawal of a subnet address (0).

Reserved: MUST be zero. Reserved for future use.

#### 13.11.1. IPv6 Attached Subnet Processing

Processing of the IPv6 Attached Subnet Data Item MUST be done within the context of the DLEP Peer session on which it is presented.

If the containing message is a Session Message, e.g., Session Initialization Message (Section 12.5), or Session Update Message (Section 12.7), the receiver of inconsistent information MUST issue a

Session Termination Message (Section 12.9) containing a Status Data Item (Section 13.1) with status code set to 130 'Invalid Data', and transition to the Session Termination state. Examples of such conditions are:

- o A subnet Drop operation referencing a subnet that is not associated with the peer in the current session.
- o A subnet Add operation referencing a subnet that has already been added to the peer in the current session.

If the containing message is a Destination Message, e.g., Destination Up Message (Section 12.11), or Destination Update Message (Section 12.17), the receiver of inconsistent information MAY issue the appropriate response message containing a Status Data Item, with status code set to 3 'Inconsistent Data', but MUST continue with session processing. Examples of such conditions are:

- o A subnet Add operation referencing a subnet that has already been added to the destination in the current session.
- o A subnet Add operation referencing a subnet that is associated with a different destination in the current session.
- o An subnet Add operation referencing an subnet that makes no sense, for example defined as not forwardable in [RFC6890].
- o A subnet Drop operation referencing a subnet that is not associated with the destination in the current session.

If no response message is appropriate, for example, the Destination Update Message, then the implementation MUST continue with session processing.

Modems that do not track IPv6 subnets MUST silently ignore IPv6 Attached Subnet Data Items.

#### 13.12. Maximum Data Rate (Receive)

The Maximum Data Rate (Receive) (MDRR) Data Item is used to indicate the maximum theoretical data rate, in bits per second, that can be achieved while receiving data on the link.

The Maximum Data Rate (Receive) Data Item contains the following fields:



```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Data Item Type                               | Length                |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               MDRR (bps)                               :
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
:                               MDRR (bps)                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Data Item Type: 12

Length: 8

Maximum Data Rate (Receive): A 64-bit unsigned integer, representing the maximum theoretical data rate, in bits per second (bps), that can be achieved while receiving on the link.

### 13.13. Maximum Data Rate (Transmit)

The Maximum Data Rate (Transmit) (MDRT) Data Item is used to indicate the maximum theoretical data rate, in bits per second, that can be achieved while transmitting data on the link.

The Maximum Data Rate (Transmit) Data Item contains the following fields:

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| Data Item Type                               | Length                |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                               MDRT (bps)                               :
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
:                               MDRT (bps)                               |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Data Item Type: 13

Length: 8

Maximum Data Rate (Transmit): A 64-bit unsigned integer, representing the maximum theoretical data rate, in bits per second (bps), that can be achieved while transmitting on the link.

## 13.14. Current Data Rate (Receive)

The Current Data Rate (Receive) (CDRR) Data Item is used to indicate the rate at which the link is currently operating for receiving traffic.

When used in the Link Characteristics Request Message (Section 12.18), Current Data Rate (Receive) represents the desired receive rate, in bits per second, on the link.

The Current Data Rate (Receive) Data Item contains the following fields:

0																1																2																3															
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1																																
Data Item Type																Length																																															
																CDRR (bps)																:																															
:																CDRR (bps)																																															

Data Item Type: 14

Length: 8

Current Data Rate (Receive): A 64-bit unsigned integer, representing the current data rate, in bits per second, that can currently be achieved while receiving traffic on the link.

If there is no distinction between Current Data Rate (Receive) and Maximum Data Rate (Receive) (Section 13.12), Current Data Rate (Receive) MUST be set equal to the Maximum Data Rate (Receive). The Current Data Rate (Receive) MUST NOT exceed the Maximum Data Rate (Receive).

## 13.15. Current Data Rate (Transmit)

The Current Data Rate (Transmit) (CDRT) Data Item is used to indicate the rate at which the link is currently operating for transmitting traffic.

When used in the Link Characteristics Request Message (Section 12.18), Current Data Rate (Transmit) represents the desired transmit rate, in bits per second, on the link.

The Current Data Rate (Transmit) Data Item contains the following fields:

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
| Data Item Type                               | Length                |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               CDRT (bps)      :
+-----+-----+-----+-----+-----+-----+-----+-----+
:                               CDRT (bps)      |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Data Item Type: 15

Length: 8

Current Data Rate (Transmit): A 64-bit unsigned integer, representing the current data rate, in bits per second, that can currently be achieved while transmitting traffic on the link.

If there is no distinction between Current Data Rate (Transmit) and Maximum Data Rate (Transmit) (Section 13.13), Current Data Rate (Transmit) MUST be set equal to the Maximum Data Rate (Transmit). The Current Data Rate (Transmit) MUST NOT exceed the Maximum Data Rate (Transmit).

### 13.16. Latency

The Latency Data Item is used to indicate the amount of latency, in microseconds, on the link.

The Latency value is reported as transmission delay. The calculation of latency is implementation dependent. For example, the latency may be a running average calculated from the internal queuing.

The Latency Data Item contains the following fields:

```

      0               1               2               3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
| Data Item Type                               | Length                |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               Latency        :
+-----+-----+-----+-----+-----+-----+-----+-----+
:                               Latency        |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Data Item Type: 16

Length: 8

Latency: A 64-bit unsigned integer, representing the transmission delay, in microseconds, that a packet encounters as it is transmitted over the link.

### 13.17. Resources

The Resources (RES) Data Item is used to indicate the amount of finite resources available for data transmission and reception at the destination as a percentage, with 0 meaning 'no resources remaining', and 100 meaning 'a full supply', assuming that when Resources reaches 0 data transmission and/or reception will cease.

An example of such resources might be battery life, but could equally be magic beans. The list of resources that might be considered is beyond the scope of this document, and is left to implementations to decide.

This Data Item is designed to be used as an indication of some capability of the modem and/or router at the destination.

The Resources Data Item contains the following fields:

0																1																2																3															
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9																								
Data Item Type																Length																																															
RES																																																															

Data Item Type: 17

Length: 1

Resources: An 8-bit unsigned integer percentage, 0-100, representing the amount of resources available. Any value greater than 100 MUST be considered as invalid.

If a device cannot calculate Resources, this Data Item MUST NOT be issued.

## 13.18. Relative Link Quality (Receive)

The Relative Link Quality (Receive) (RLQR) Data Item is used to indicate the quality of the link to a destination for receiving traffic, with 0 meaning 'worst quality', and 100 meaning 'best quality'.

Quality in this context is defined as an indication of the stability of a link for reception; a destination with high Relative Link Quality (Receive) is expected to have generally stable DLEP metrics, and the metrics of a destination with low Relative Link Quality (Receive) can be expected to rapidly fluctuate over a wide range.

The Relative Link Quality (Receive) Data Item contains the following fields:

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1								
Data Item Type										Length																													
RLQR																																							

Data Item Type: 18

Length: 1

Relative Link Quality (Receive): A non-dimensional unsigned 8-bit integer, 0-100, representing relative quality of the link for receiving traffic. Any value greater than 100 MUST be considered as invalid.

If a device cannot calculate the Relative Link Quality (Receive), this Data Item MUST NOT be issued.

## 13.19. Relative Link Quality (Transmit)

The Relative Link Quality (Transmit) (RLQT) Data Item is used to indicate the quality of the link to a destination for transmitting traffic, with 0 meaning 'worst quality', and 100 meaning 'best quality'.

Quality in this context is defined as an indication of the stability of a link for transmission; a destination with high Relative Link Quality (Transmit) is expected to have generally stable DLEP metrics, and the metrics of a destination with low Relative Link Quality (Transmit) can be expected to rapidly fluctuate over a wide range.

The Relative Link Quality (Transmit) Data Item contains the following fields:

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
| Data Item Type                               | Length                |
+-----+-----+-----+-----+-----+-----+-----+-----+
|           RLQT                               |
+-----+-----+-----+-----+-----+-----+-----+

```

Data Item Type: 19

Length: 1

Relative Link Quality (Transmit): A non-dimensional unsigned 8-bit integer, 0-100, representing relative quality of the link for transmitting traffic. Any value greater than 100 MUST be considered as invalid.

If a device cannot calculate the Relative Link Quality (Transmit), this Data Item MUST NOT be issued.

### 13.20. Maximum Transmission Unit (MTU)

The Maximum Transmission Unit (MTU) Data Item is used to indicate the maximum size, in octets, of an IP packet that can be transmitted without fragmentation, including headers, but excluding any lower layer headers.

The Maximum Transmission Unit Data Item contains the following fields:

```

      0               1               2               3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
| Data Item Type                               | Length                |
+-----+-----+-----+-----+-----+-----+-----+-----+
|           MTU                               |
+-----+-----+-----+-----+-----+-----+-----+

```

Data Item Type: 20

Length: 2

Maximum Transmission Unit: The maximum size, in octets, of an IP packet that can be transmitted without fragmentation, including headers, but excluding any lower layer headers.

If a device cannot calculate the Maximum Transmission Unit, this Data Item MUST NOT be issued.

#### 14. Security Considerations

The potential security concerns when using DLEP are:

1. An attacker might pretend to be a DLEP participant, either at DLEP session initialization, or by injection of DLEP Messages once a session has been established.
2. DLEP Data Items could be altered by an attacker, causing the receiving implementation to inappropriately alter its information base concerning network status.
3. An attacker could join an unsecured radio network and inject over-the-air signals that maliciously influence the information reported by a DLEP modem, causing a router to forward traffic to an inappropriate destination.

The implications of attacks on DLEP peers are directly proportional to the extent to which DLEP data is used within the control plane. While the use of DLEP data in other control plane components is out of scope for this document, as an example, if DLEP statistics are incorporated into route cost calculations, adversaries masquerading as a DLEP peer, and injecting malicious data via DLEP, could cause suboptimal route selection, adversely impacting network performance. Similar issues can arise if DLEP data is used as an input to policing algorithms - injection of malicious data via DLEP can cause those policing algorithms to make incorrect decisions, degrading network throughput.

For these reasons, security of the DLEP transport must be considered at both the transport layer, and at Layer 2.

At the transport layer, when TLS is in use, each peer SHOULD check the validity of credentials presented by the other peer during TLS session establishment. Implementations following the "dedicated deployments" model attempting to use TLS MAY need to consider use of pre-shared keys for credentials, and provide specialized techniques for peer identity validation, and MAY refer to [RFC5487] for additional details. Implementations following the "networked deployment" model described in Implementation Scenarios SHOULD refer to [RFC7525] for additional details.

At layer 2 - since DLEP is restricted to operation over a single (possibly logical) hop, implementations SHOULD also secure the Layer

2 link. Examples of technologies that can be deployed to secure the Layer 2 link include [IEEE-802.1AE] and [IEEE-802.1X].

By examining the Secured Medium flag in the Peer Type Data Item (Section 13.4), a router can decide if it is able to trust the information supplied via a DLEP modem. If this is not the case, then the router SHOULD consider restricting the size of attached subnets, announced in IPv4 Attached Subnet Data Items (Section 13.10) and/or IPv6 Attached Subnet Data Items (Section 13.11), that are considered for route selection.

To avoid potential denial of service attack, it is RECOMMENDED that implementations using the Peer Discovery mechanism maintain an information base of hosts that persistently fail Session Initialization having provided an acceptable Peer Discovery Signal, and ignore subsequent Peer Discovery Signals from such hosts.

This specification does not address security of the data plane, as it (the data plane) is not affected, and standard security procedures can be employed.

## 15. IANA Considerations

### 15.1. Registrations

Upon approval of this document, IANA is requested to create a new protocol registry for Dynamic Link Exchange Protocol (DLEP). The remainder of this section requests the creation of new DLEP specific registries.

### 15.2. Signal Type Registration

Upon approval of this document, IANA is requested to create a new DLEP registry, named "Signal Type Values".

The following table provides initial registry values and the [RFC5226] defined policies that should apply to the registry:

Type Code	Description/Policy
0	Reserved
1	Peer Discovery Signal
2	Peer Offer Signal
3-65519	Specification Required
65520-65534	Private Use
65535	Reserved



### 15.3. Message Type Registration

Upon approval of this document, IANA is requested to create a new DLEP registry, named "Message Type Values".

The following table provides initial registry values and the [RFC5226] defined policies that should apply to the registry:

Type Code	Description/Policy
0	Reserved
1	Session Initialization Message
2	Session Initialization Response Message
3	Session Update Message
4	Session Update Response Message
5	Session Termination Message
6	Session Termination Response Message
7	Destination Up Message
8	Destination Up Response Message
9	Destination Announce Message
10	Destination Announce Response Message
11	Destination Down Message
12	Destination Down Response Message
13	Destination Update Message
14	Link Characteristics Request Message
15	Link Characteristics Response Message
16	Heartbeat Message
17-65519	Specification Required
65520-65534	Private Use
65535	Reserved

### 15.4. DLEP Data Item Registrations

Upon approval of this document, IANA is requested to create a new DLEP registry, named "Data Item Type Values".

The following table provides initial registry values and the [RFC5226] defined policies that should apply to the registry:

Type Code	Description/Policy
0	Reserved
1	Status
2	IPv4 Connection Point
3	IPv6 Connection Point
4	Peer Type
5	Heartbeat Interval
6	Extensions Supported
7	MAC Address
8	IPv4 Address
9	IPv6 Address
10	IPv4 Attached Subnet
11	IPv6 Attached Subnet
12	Maximum Data Rate (Receive) (MDRR)
13	Maximum Data Rate (Transmit) (MDRT)
14	Current Data Rate (Receive) (CDRR)
15	Current Data Rate (Transmit) (CDRT)
16	Latency
17	Resources (RES)
18	Relative Link Quality (Receive) (RLQR)
19	Relative Link Quality (Transmit) (RLQT)
20	Maximum Transmission Unit (MTU)
21-65407	Specification Required
65408-65534	Private Use
65535	Reserved

#### 15.5. DLEP Status Code Registrations

Upon approval of this document, IANA is requested to create a new DLEP registry, named "Status Code Values".

The following table provides initial registry values and the [RFC5226] defined policies that should apply to the registry:

Status Code	Failure Mode	Description/Policy
0	Continue	Success
1	Continue	Not Interested
2	Continue	Request Denied
3	Continue	Inconsistent Data
4-111	Continue	Specification Required
112-127	Continue	Private Use
128	Terminate	Unknown Message
129	Terminate	Unexpected Message
130	Terminate	Invalid Data
131	Terminate	Invalid Destination
132	Terminate	Timed Out
133-239	Terminate	Specification Required
240-254	Terminate	Private Use
255	Terminate	Reserved

#### 15.6. DLEP Extensions Registrations

Upon approval of this document, IANA is requested to create a new DLEP registry, named "Extension Type Values".

The following table provides initial registry values and the [RFC5226] defined policies that should apply to the registry:

Code	Description/Policy
0	Reserved
1-65519	Specification Required
65520-65534	Private Use
65535	Reserved

Table 3: DLEP Extension types

#### 15.7. DLEP IPv4 Connection Point Flags

Upon approval of this document, IANA is requested to create a new DLEP registry, named "IPv4 Connection Point Flags".

The following table provides initial registry values and the [RFC5226] defined policies that should apply to the registry:

Bit	Description/Policy
0-6	Unassigned/Specification Required
7	Use TLS [RFC5246] indicator

#### 15.8. DLEP IPv6 Connection Point Flags

Upon approval of this document, IANA is requested to create a new DLEP registry, named "IPv6 Connection Point Flags".

The following table provides initial registry values and the [RFC5226] defined policies that should apply to the registry:

Bit	Description/Policy
0-6	Unassigned/Specification Required
7	Use TLS [RFC5246] indicator

#### 15.9. DLEP Peer Type Flag

Upon approval of this document, IANA is requested to create a new DLEP registry, named "Peer Type Flags".

The following table provides initial registry values and the [RFC5226] defined policies that should apply to the registry:

Bit	Description/Policy
0-6	Unassigned/Specification Required
7	Secured Medium indicator

#### 15.10. DLEP IPv4 Address Flag

Upon approval of this document, IANA is requested to create a new DLEP registry, named "IPv4 Address Flags".

The following table provides initial registry values and the [RFC5226] defined policies that should apply to the registry:

Bit	Description/Policy
0-6	Unassigned/Specification Required
7	Add/Drop indicator

#### 15.11. DLEP IPv6 Address Flag

Upon approval of this document, IANA is requested to create a new DLEP registry, named "IPv6 Address Flags".

The following table provides initial registry values and the [RFC5226] defined policies that should apply to the registry:

Bit	Description/Policy
0-6	Unassigned/Specification Required
7	Add/Drop indicator

#### 15.12. DLEP IPv4 Attached Subnet Flag

Upon approval of this document, IANA is requested to create a new DLEP registry, named "IPv4 Attached Subnet Flags".

The following table provides initial registry values and the [RFC5226] defined policies that should apply to the registry:

Bit	Description/Policy
0-6	Unassigned/Specification Required
7	Add/Drop indicator

#### 15.13. DLEP IPv6 Attached Subnet Flag

Upon approval of this document, IANA is requested to create a new DLEP registry, named "IPv6 Attached Subnet Flags".

The following table provides initial registry values and the [RFC5226] defined policies that should apply to the registry:

Bit	Description/Policy
0-6	Unassigned/Specification Required
7	Add/Drop indicator

#### 15.14. DLEP Well-known Port

Upon approval of this document, IANA is requested to assign a single value in the "Service Name and Transport Protocol Port Number Registry" found at <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml> for use by "DLEP", as defined in this document. This assignment should be valid for TCP and UDP.

#### 15.15. DLEP IPv4 Link-local Multicast Address

Upon approval of this document, IANA is requested to assign an IPv4 multicast address registry found at <http://www.iana.org/assignments/multicast-addresses> for use as the "IPv4 DLEP Discovery Address".

#### 15.16. DLEP IPv6 Link-local Multicast Address

Upon approval of this document, IANA is requested to assign an IPv6 multicast address registry found at <http://www.iana.org/assignments/multicast-addresses> for use as the "IPv6 DLEP Discovery Address".

### 16. Acknowledgments

We would like to acknowledge and thank the members of the DLEP design team, who have provided invaluable insight. The members of the design team are: Teco Boot, Bow-Nan Cheng, John Dowdell, and Henning Rogge.

We would also like to acknowledge the influence and contributions of Greg Harrison, Chris Olsen, Martin Duke, Subir Das, Jaewon Kang, Vikram Kaul, Nelson Powell, Lou Berger, and Victoria Pritchard.

### 17. References

#### 17.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.

- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <<http://www.rfc-editor.org/info/rfc3629>>.
- [RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., Ed., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, DOI 10.17487/RFC5082, October 2007, <<http://www.rfc-editor.org/info/rfc5082>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<http://www.rfc-editor.org/info/rfc5246>>.

## 17.2. Informative References

- [IEEE-802.1AE]  
"IEEE Standards for Local and Metropolitan Area Networks: Media Access Control (MAC) Security",  
DOI 10.1109/IEEESTD.2006.245590, August 2006.
- [IEEE-802.1X]  
"IEEE Standards for Local and Metropolitan Area Networks: Port based Network Access Control",  
DOI 10.1109/IEEESTD.2010.5409813, February 2010.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <<http://www.rfc-editor.org/info/rfc5226>>.
- [RFC5487] Badra, M., "Pre-Shared Key Cipher Suites for TLS with SHA-256/384 and AES Galois Counter Mode", RFC 5487, DOI 10.17487/RFC5487, March 2009, <<http://www.rfc-editor.org/info/rfc5487>>.
- [RFC5578] Berry, B., Ed., Ratliff, S., Paradise, E., Kaiser, T., and M. Adams, "PPP over Ethernet (PPPoE) Extensions for Credit Flow and Link Metrics", RFC 5578, DOI 10.17487/RFC5578, February 2010, <<http://www.rfc-editor.org/info/rfc5578>>.
- [RFC6890] Cotton, M., Vegoda, L., Bonica, R., Ed., and B. Haberman, "Special-Purpose IP Address Registries", BCP 153, RFC 6890, DOI 10.17487/RFC6890, April 2013, <<http://www.rfc-editor.org/info/rfc6890>>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre,  
 "Recommendations for Secure Use of Transport Layer  
 Security (TLS) and Datagram Transport Layer Security  
 (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May  
 2015, <<http://www.rfc-editor.org/info/rfc7525>>.

#### Appendix A. Discovery Signal Flows

Router	Modem	Signal Description
=====		
		Router initiates discovery, starts
		a timer, send Peer Discovery
-----Peer Discovery----->X		Signal.
~ ~ ~ ~ ~		
		Router discovery timer expires
		without receiving Peer Offer.
		Router sends another Peer
-----Peer Discovery----->		Discovery Signal.
		Modem receives Peer Discovery
		Signal.
<-----Peer Offer-----		Modem sends Peer Offer with
:		Connection Point information.
:		
:		Router MAY cancel discovery timer
:		and stop sending Peer Discovery
:		Signals.

#### Appendix B. Peer Level Message Flows

##### B.1. Session Initialization

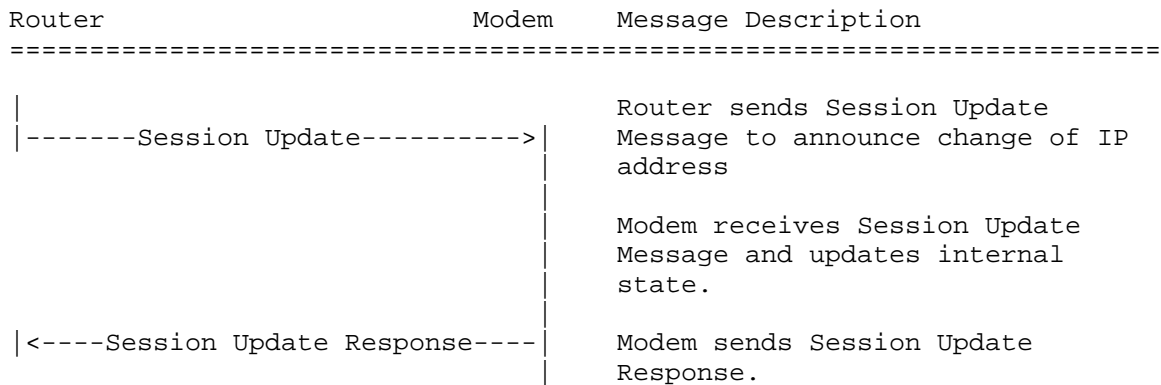


Router	Modem	Message Description
=====		
		Router connects to discovered or pre-configured Modem Connection Point.
--TCP connection established--->		
		Router sends Session Initialization Message.
----Session Initialization----->		
		Modem receives Session Initialization Message.
		Modem sends Session Initialization Response, with Success Status Data Item.
<--Session Initialization Resp.-		
		Session established. Heartbeats begin.
<<=====>>		
:	:	

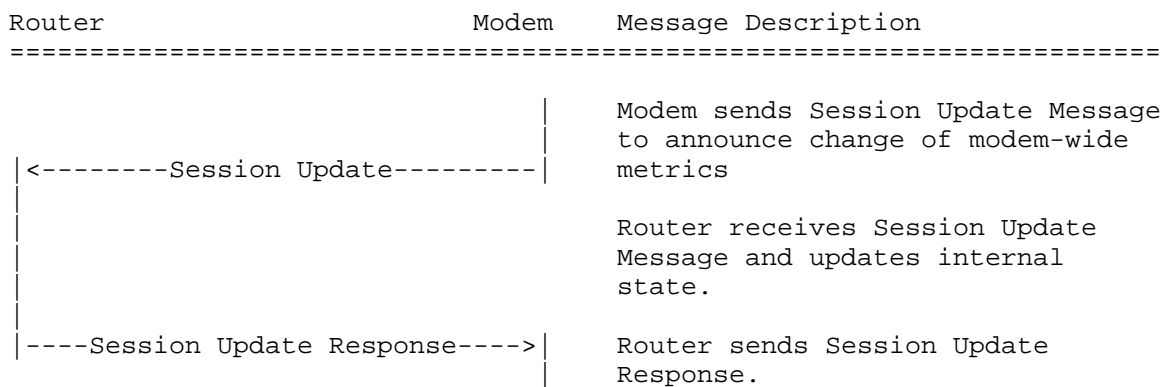
## B.2. Session Initialization - Refused

Router	Modem	Message Description
=====		
		Router connects to discovered or pre-configured Modem Connection Point.
--TCP connection established--->		
		Router sends Session Initialization Message.
----Session Initialization----->		
		Modem receives Session Initialization Message, and will not support the advertised extensions.
		Modem sends Session Initialization Response, with 'Request Denied' Status Data Item.
<-Session Initialization Resp.--		
		Router receives negative Session Initialization Response, closes TCP connection.
-----TCP close-----		

## B.3. Router Changes IP Addresses



## B.4. Modem Changes Session-wide Metrics



## B.5. Router Terminates Session

Router	Modem	Message Description
=====		
-----Session Termination----->		Router sends Session Termination Message with Status Data Item.
-----TCP shutdown (send)--->		Router stops sending Messages.
		Modem receives Session Termination, stops counting received heartbeats and stops sending heartbeats.
<---Session Termination Resp.---		Modem sends Session Termination Response with Status 'Success'.
		Modem stops sending Messages.
-----TCP close-----		Session terminated.

## B.6. Modem Terminates Session

Router	Modem	Message Description
=====		
<-----Session Termination-----		Modem sends Session Termination Message with Status Data Item.
		Modem stops sending Messages.
		Router receives Session Termination, stops counting received heartbeats and stops sending heartbeats.
---Session Termination Resp.--->		Router sends Session Termination Response with Status 'Success'.
		Router stops sending Messages.
-----TCP close-----		Session terminated.

## B.7. Session Heartbeats

Router	Modem	Message Description
=====		
-----Heartbeat----->		Router sends heartbeat Message
		Modem resets heartbeats missed counter.
~ ~ ~ ~ ~ ~ ~		
-----[Any Message]----->		When the Modem receives any Message from the Router.
		Modem resets heartbeats missed counter.
~ ~ ~ ~ ~ ~ ~		
<-----Heartbeat-----		Modem sends heartbeat Message
		Router resets heartbeats missed counter.
~ ~ ~ ~ ~ ~ ~		
<-----[Any Message]-----		When the Router receives any Message from the Modem.
		Modem resets heartbeats missed counter.

#### B.8. Router Detects a Heartbeat timeout

Router	Modem	Message Description
=====		
X<-----		Router misses a heartbeat
X<-----		Router misses too many heartbeats
-----Session Termination----->		Router sends Session Termination Message with 'Timeout' Status Data Item.
:		
:		Termination proceeds...

## B.9. Modem Detects a Heartbeat timeout

Router	Modem	Message Description
=====		
----->X		Modem misses a heartbeat
----->X		Modem misses too many heartbeats
<-----Session Termination-----		Modem sends Session Termination
		Message with 'Timeout' Status
		Data Item.
	:	
	:	Termination proceeds...

## Appendix C. Destination Specific Message Flows

## C.1. Common Destination Notification

Router	Modem	Message Description
=====		
		Modem detects a new logical destination is reachable, and sends Destination Up Message.
<-----Destination Up-----		
-----Destination Up Resp.---->		Router sends Destination Up Response.
~ ~ ~ ~ ~ ~ ~		
		Modem detects change in logical destination metrics, and sends Destination Update Message.
<-----Destination Update-----		
~ ~ ~ ~ ~ ~ ~		
		Modem detects change in logical destination metrics, and sends Destination Update Message.
<-----Destination Update-----		
~ ~ ~ ~ ~ ~ ~		
		Modem detects logical destination is no longer reachable, and sends Destination Down Message.
<-----Destination Down-----		
		Router receives Destination Down, updates internal state, and sends Destination Down Response Message.
-----Destination Down Resp.--->		

## C.2. Multicast Destination Notification

Router	Modem	Message Description
=====		
		Router detects a new multicast destination is in use, and sends Destination Announce Message.
-----Destination Announce----->		
		Modem updates internal state to monitor multicast destination, and sends Destination Announce Response.
<-----Dest. Announce Resp.-----		
~ ~ ~ ~ ~ ~ ~		
<-----Destination Update-----		Modem detects change in multicast destination metrics, and sends Destination Update Message.
~ ~ ~ ~ ~ ~ ~		
<-----Destination Update-----		Modem detects change in multicast destination metrics, and sends Destination Update Message.
~ ~ ~ ~ ~ ~ ~		
		Router detects multicast destination is no longer in use, and sends Destination Down Message.
-----Destination Down----->		
		Modem receives Destination Down, updates internal state, and sends Destination Down Response Message.
<-----Destination Down Resp.-----		

### C.3. Link Characteristics Request

Router	Modem	Message Description
=====		
	~ ~ ~ ~ ~ ~ ~	Destination has already been announced by either peer.
		Router requires different Characteristics for the destination, and sends Link Characteristics Request Message.
--Link Characteristics Request-->		
		Modem attempts to adjust link properties to meet the received request, and sends a Link Characteristics Response Message with the new values.
<---Link Characteristics Resp.--		

## Authors' Addresses

Stan Ratliff  
 VT iDirect  
 13861 Sunrise Valley Drive, Suite 300  
 Herndon, VA 20171  
 USA

Email: [sratliff@idirect.net](mailto:sratliff@idirect.net)

Shawn Jury  
 Cisco Systems  
 170 West Tasman Drive  
 San Jose, CA 95134  
 USA

Email: [sjury@cisco.com](mailto:sjury@cisco.com)

Darryl Satterwhite  
 Broadcom

Email: [dsatterw@broadcom.com](mailto:dsatterw@broadcom.com)



Rick Taylor  
Airbus Defence & Space  
Quadrant House  
Celtic Springs  
Coedkernew  
Newport NP10 8FZ  
UK

Email: [rick.taylor@airbus.com](mailto:rick.taylor@airbus.com)

Bo Berry

Mobile Ad hoc Networks Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: August 29, 2013

C. Perkins  
Futurewei  
S. Ratliff  
Cisco  
J. Dowdell  
Cassidian  
February 25, 2013

Dynamic MANET On-demand (AODVv2) Routing  
draft-ietf-manet-dymo-26

Abstract

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

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on August 29, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Overview . . . . .	4
2. Terminology . . . . .	4
3. Notational Conventions . . . . .	8
4. Applicability Statement . . . . .	8
5. Data Structures . . . . .	10
5.1. Route Table Entry . . . . .	10
5.2. Bidirectional Connectivity and Blacklists . . . . .	12
5.3. Router Clients and Client Networks . . . . .	13
5.4. AODVv2 Packet Header Fields and Information Elements . . . . .	13
5.5. Sequence Numbers . . . . .	14
5.6. Enabling Alternate Metrics . . . . .	15
5.7. RREQ Table: Received RREQ Messages . . . . .	17
6. AODVv2 Operations on Route Table Entries . . . . .	18
6.1. Evaluating Incoming Routing Information . . . . .	19
6.2. Applying Route Updates To Route Table Entries . . . . .	20
6.3. Route Table Entry Timeouts . . . . .	21
7. Routing Messages RREQ and RREP (RteMsgs) . . . . .	21
7.1. Route Discovery Retries and Buffering . . . . .	22
7.2. RteMsg Structure . . . . .	23
7.3. RREQ Generation . . . . .	25
7.4. RREP Generation . . . . .	26
7.5. Handling a Received RteMsg . . . . .	27
7.5.1. Additional Handling for Incoming RREQ . . . . .	28
7.5.2. Additional Handling for Incoming RREP . . . . .	29
7.6. Suppressing Redundant RREQ messages . . . . .	30
8. Route Maintenance and RERR Messages . . . . .	30
8.1. Maintaining Route Lifetimes During Packet Forwarding . . . . .	30
8.2. Active Next-hop Router Adjacency Monitoring . . . . .	31
8.3. RERR Generation . . . . .	32
8.3.1. Case 1: Undeliverable Packet . . . . .	33
8.3.2. Case 2: Broken Link . . . . .	33
8.4. Receiving and Handling RERR Messages . . . . .	34
9. Unknown Message and TLV Types . . . . .	35
10. Simple Internet Attachment . . . . .	35
11. Multiple Interfaces . . . . .	36
12. AODVv2 Control Packet/Message Generation Limits . . . . .	37
13. Optional Features . . . . .	37
13.1. Expanding Rings Multicast . . . . .	37
13.2. Intermediate RREP . . . . .	37
13.3. Precursor Lists and Notifications . . . . .	38
13.3.1. Overview . . . . .	38

13.3.2. Precursor Notification Details . . . . .	38
13.4. Multicast RREP Response to RREQ . . . . .	39
13.5. RREP_ACK . . . . .	39
13.6. Message Aggregation . . . . .	40
13.7. Added Routing Information in RteMsgs . . . . .	40
13.7.1. Including Added Node Information . . . . .	40
13.7.2. Handling Added Node Information . . . . .	41
14. Administratively Configurable Parameters and Timer Values . . . . .	42
14.1. Timers . . . . .	43
14.2. Protocol constants . . . . .	43
14.3. Administrative (functional) controls . . . . .	44
14.4. Other administrative parameters and lists . . . . .	44
15. IANA Considerations . . . . .	44
15.1. AODVv2 Message Types Specification . . . . .	45
15.2. Message TLV Type Specification . . . . .	45
15.3. Address Block TLV Specification . . . . .	45
15.4. Metric Type Number Allocation . . . . .	46
16. Security Considerations . . . . .	46
17. Acknowledgments . . . . .	48
18. References . . . . .	48
18.1. Normative References . . . . .	48
18.2. Informative References . . . . .	49
Appendix A. Example RFC 5444-compliant packet formats . . . . .	50
A.1. RREQ Message Format . . . . .	51
A.2. RREP Message Format . . . . .	53
A.3. RERR Message Format . . . . .	55
A.4. RREP_ACK Message Format . . . . .	56
Appendix B. Changes since revision ...-25.txt . . . . .	56
Appendix C. Changes since revision ...-24.txt . . . . .	57
Appendix D. Changes between revisions ...-21.txt and ...-24.txt . . . . .	57
Appendix E. Shifting Network Prefix Advertisement Between AODVv2 Routers . . . . .	59
Authors' Addresses . . . . .	59

## 1. Overview

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

During route discovery, the originating AODVv2 router (RREQ\_Gen) multicasts a Route Request message (RREQ) to find a route toward some target destination. Using a hop-by-hop retransmission algorithm, each AODVv2 router receiving the RREQ message records a route toward the originator. When the target's AODVv2 router (RREP\_Gen) receives the RREQ, it records a route toward RREQ\_Gen and generates a Route Reply (RREP) unicast toward RREQ\_Gen. Each AODVv2 router that receives the RREP stores a route toward the target, and again unicasts the RREP toward the originator. When RREQ\_Gen receives the RREP, routes have then been established between RREQ\_Gen (the originating AODVv2 router) and RREP\_Gen (the target's AODVv2 router) in both directions.

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

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

## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

This document also uses some terminology from [RFC5444].

This document defines the following terminology:

**Adjacency**

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

**AODVv2 Router**

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

**AODVv2 Sequence Number (SeqNum)**

Same as Sequence Number.

**Current\_Time**

The current time as maintained by the AODVv2 router.

**disregard**

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

**Handling Router (HandlingRtr)**

HandlingRtr denotes the AODVv2 router receiving and handling an AODVv2 message.

**Incoming Link**

A link over which an AODVv2 router has received a message from an adjacent router.

**MANET**

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

**node**

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

**Originating Node (OrigNode)**

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

**reactive**

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

**Routable Unicast IP Address**

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

**Route Error (RERR)**

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

**Route Reply (RREP)**

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

**Route Request (RREQ)**

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

**Router Client**

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

**RREP Generating Router (RREP\_Gen)**

The RREP Generating Router is the AODVv2 router that serves TargNode. RREP\_Gen generates the RREP message to advertise a route for TargNode.

**RREQ Generating Router (RREQ\_Gen)**

The RREQ Generating Router is the AODVv2 router that serves OrigNode. RREQ\_Gen generates the RREQ message to discover a route for TargNode.

**Sequence Number (SeqNum)**

AODVv2 mandates that each AODVv2 router maintain an unsigned integer known as the router's "Sequence Number". The Sequence Number guarantees the temporal order of routing information to maintain loop-free routes, and fulfills the same role as the "Destination Sequence Number" of DSDV, and as the AODV Sequence Number in RFC 3561[RFC3561]. The value zero (0) is reserved to indicate that the Sequence Number for an address is unknown.

**Target Node (TargNode)**

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

**Type-Length-Value structure (TLV)**

A generic way to represent information as specified in [RFC5444].

**Unreachable Node (UnreachableNode)**

An UnreachableNode is a node for which a forwarding route is unknown.

**valid route**

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



### 3. Notational Conventions

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

Notation	Information Location and/or Meaning
Route[Addr]	A route table entry towards Addr
Route[Addr].{field}	A field in a route table entry
--	--
<msg-hop-count>	RFC 5444 Message Header <msg-hop-count>
<msg-hop-limit>	RFC 5444 Message Header <msg-hop-limit>
AddrBlk	an RFC 5444 Address TLV Block
AddrBlk[1]	The first address slot in AddrBlk
AddrBlk[N]	The Nth address slot in AddrBlk
OrigNdx	The index of OrigNode within the AddrBlk
TargNdx	The index of TargNode within the AddrBlk
AddrBlk[OrigNode]	AddrBlk[OrigNdx]
AddrBlk[TargNode]	AddrBlk[TargNdx]
AddrTLV	an RFC 5444 Address Block TLV
AddrTLV[1]	the first item in AddrTLV
AddrTLV[N]	the Nth item in AddrTLV
AddrTLV[OrigNode]	AddrTLV[OrigNdx]
AddrTLV[TargNode]	AddrTLV[TargNdx]
MetricTLV	Metric AddrTLV for AddrBlk
SeqNumTLV	Sequence Number AddrTLV for AddrBlk
OrigSeqNumTLV	Originating Node Sequence Number AddrTLV
TargSeqNumTLV	Target Node Sequence Number AddrTLV
--	--
OrigNode	Originating Node
RREQ_Gen	AODVv2 router originating an RREQ
RREP_Gen	AODVv2 router responding to an RREQ
RteMsg	Either RREQ or RREP
RteMsg.{field}	Field in RREQ or RREP
HandlingRtr	Handling Router
TargNode	Target Node
UnreachableNode	Unreachable Node

Table 1

### 4. Applicability Statement

The AODVv2 routing protocol is designed for stub (i.e., non-transit) or disconnected (i.e., from the Internet) mobile ad hoc networks (MANETs). AODVv2 handles a wide variety of mobility patterns by

determining routes on-demand. AODVv2 also handles a wide variety of traffic patterns. In networks with a large number of routers, AODVv2 is best suited for relatively sparse traffic scenarios where any particular router forwards packets to only a small percentage of the AODVv2 routers in the network, due to the on-demand nature of route discovery and route maintenance. AODVv2 supports routers with multiple interfaces, as long as each interface has its own (unicast routeable) IP address; the set of all network interfaces supporting AODVv2 is administratively configured in a list (namely, AODVv2\_INTERFACES).

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

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

In addition to routing for its own local applications, each AODVv2 router can also route on behalf of other non-routing nodes (i.e., "hosts", or, in this document, "clients"), reachable via those interfaces. Each AODVv2 router, if serving router clients other than itself, is configured with information about the IP addresses of its clients. No AODVv2 router is required to have information about the relationship between any other AODVv2 router and its router clients (see Section 5.3).

The coordination among multiple AODVv2 routers to distribute routing information correctly for a shared address (i.e. an address that is advertised and can be reached via multiple AODVv2 routers) is not described in this document. The AODVv2 router operation of shifting responsibility for a routing client from one AODVv2 router to another is mentioned in Appendix E. Address assignment procedures are entirely out of scope for AODVv2. Any such node which is not itself an AODVv2 router SHOULD NOT be served by more than one AODVv2 router at any one time.

Multi-homing is difficult unless the sequence number is expanded to include the AODVv2 router's IP address as well as SeqNum. Otherwise, comparing sequence numbers would not work to evaluate freshness. Even when the IP address is included, there isn't a good way to compare sequence numbers from different IP addresses, but at least a handling node can determine whether the two given sequence numbers are comparable. If the route table can store multiple routes for the

same destination, then multi-homing can work with sequence numbers augmented by IP addresses.

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

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

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

## 5. Data Structures

### 5.1. Route Table Entry

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

Conceptually, a route table entry has the following fields:

#### Route.Address

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

#### Route.PrefixLength

The length of the netmask/prefix. If the value of the `Route.PrefixLength` is not `INVALID_PREFIX_LENGTH` and is different than the length of addresses in the address family used by the AODVv2 routers, the associated address is a routing prefix, rather than a host address.

**Route.SeqNum**

The Sequence Number associated with a route table entry

**Route.NextHopAddress**

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

**Route.NextHopInterface**

The interface used to send packets toward the Route.Address

**Route.LastUsed**

The time that this route was last used

**Route.ExpirationTime**

The time at which this route must expire

**Route.Broken**

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

**Route.MetricType**

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

**Route.Metric**

The cost of the route towards Route.Address

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

**Active**

An Active route is in current use for forwarding packets

**Idle**

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

**Expired**

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

**Broken**

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

#### Timed

The expiration of a Timed route is controlled by the `Route.ExpirationTime` time of the route table entry (instead of `MAX_IDLETIME`). Until that time, a Timed route can be used for forwarding packets. Afterwards, the route must be Expired (or expunged).

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

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

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

### 5.2. Bidirectional Connectivity and Blacklists

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

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

**Blacklist.Node**

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

**Blacklist.RemoveTime**

The time at which Blacklist.Node will be removed from the blacklist.

### 5.3. Router Clients and Client Networks

An AODVv2 router may offer routing services to other nodes that are not AODVv2 routers. AODVv2 defines the Sequence Number to be the same for the AODVv2 router and each of its clients.

For this purpose, CLIENT\_ADDRESSES must be configured on each AODVv2 router with the following information:

**Client IP address**

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

**Client Prefix Length**

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

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

### 5.4. AODVv2 Packet Header Fields and Information Elements

In its default mode of operation, AODVv2 transmits UDP packets using the parameters for port number and IP protocol specified in [RFC5498] to carry protocol packets. By default, AODVv2 packets are sent with the IP destination address set to the link-local multicast address LL-MANET-Routers [RFC5498] unless otherwise specified. Therefore, all AODVv2 routers MUST subscribe to LL-MANET-Routers [RFC5498] to receiving AODVv2 messages. In order to reduce multicast overhead, retransmitting multicast packets in MANETs SHOULD be done according to methods specified in [RFC6621]. AODVv2 does not specify which method should be used to restrict the set of AODVv2 routers that have the responsibility to retransmit multicast packets. Note that multicast packets MAY be sent via unicast. For example, this may occur for certain link-types (non-broadcast media), for manually

configured router adjacencies, or in order to improve robustness.

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

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

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

A packet formatted according to RFC 5444 contains zero or more messages.

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

Each address block may also have an associated TLV block; this TLV block may encode multiple TLVs. According to RFC 5444, each such TLV may itself include an array of values.

If a packet contains only a single AODVv2 message and no packet TLVs, it need only include a minimal Packet-Header [RFC5444]. The length of an address (32 bits for IPv4 and 128 bits for IPv6) inside an AODVv2 message is indicated by the msg-addr-length (MAL) in the msg-header, as specified in [RFC5444].

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

## 5.5. Sequence Numbers

Sequence Numbers allow AODVv2 routers to evaluate the freshness of routing information. Proper maintenance of sequence numbers assures that the destination sequence number value stored by intermediate

AODVv2 routers is monotonically increasing along any path from any source to the destination. As a consequence, loop freedom is assured.

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

An AODVv2 router SHOULD maintain its SeqNum in persistent storage. If an AODVv2 router's SeqNum is lost, it MUST take the following actions to avoid the danger of routing loops. First, the AODVv2 router MUST invalidate all route table entries, by setting Route.Broken for each entry. Furthermore the AODVv2 router MUST wait for at least MAX\_SEQNUM\_LIFETIME before transmitting or retransmitting any AODVv2 RREQ or RREP messages. If an AODVv2 protocol message is received during this waiting period, the AODVv2 router SHOULD perform normal route table entry updates, but not forward the message to other nodes. If a data packet is received for forwarding to another destination during this waiting period, the AODVv2 router MUST transmit a RERR message indicating that no route is available. At the end of the waiting period the AODVv2 router sets its SeqNum to one (1) and begins performing AODVv2 protocol operations again.

#### 5.6. Enabling Alternate Metrics

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

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



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

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

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

For some metrics, a maximum value is defined, namely MAX\_METRIC[i] where 'i' is the Metric Type. AODVv2 does not store routes that cost more than MAX\_METRIC[i]. MAX\_METRIC[3] is defined to be MAX\_HOPCOUNT, where as before 3 is the Metric Type of the HopCount metric. MAX\_HOPCOUNT MUST be larger than the AODVv2 network diameter. Otherwise, AODVv2 protocol messages may not reach their intended destinations.

#### 5.7. RREQ Table: Received RREQ Messages

Two incoming RREQ messages are considered to be "comparable" if they were generated by the same AODVv2 router in order to discover a route for the same destination with the same metric type. According to that notion of comparability, when RREQ messages are flooded in a MANET, an AODVv2 router may well receive comparable RREQ messages from more than one of its neighbors. A router, after receiving an RREQ message, MUST check against previous RREQs to assure that its response message would contain information that is not redundant. Otherwise, multicast RREQs are likely to be retransmitted again and again with almost no additional benefit, but generating a great deal of unnecessary signaling traffic and interference.

To avoid transmission of redundant RREQ messages, while still enabling the proper handling of earlier RREQ messages that may have somehow been delayed in the network, it is needed for each AODVv2 router to keep a list of the certain information about RREQ messages which it has recently received.

This list is called the AODVv2 Received RREQ Table -- or, more briefly, the RREQ Table. Two AODVv2 RREQ messages are comparable if:

- o they have the same metric type
- o they have the same OrigNode and TargNode addresses

Each entry in the RREQ Table has the following fields:

- o Metric Type
- o OrigNode address
- o TargNode address
- o Sequence Number
- o Metric

- o Timestamp

The RREQ Table is maintained so that no two entries in the RREQ Table are comparable -- that is, all RREQs represented in the RREQ Table either have different OrigNode addresses, different TargNode addresses, or different metric types. If two RREQs have the same metric type and OrigNode and Targnode addresses, the information from the one with the older Sequence Number is not needed in the table; in case they have the same Sequence Number, the one with the greater Metric value is not needed; in case they have the same Metric as well, it does not matter which table entry is maintained. Whenever a RREQ Table entry is updated, its Timestamp field should also be updated to reflect the Current\_Time.

When optional multicast RREP (see Section 13.4) is used to enable selection from among multiple possible return routes, an AODVv2 router can eliminate redundant RREP messages using the analogous mechanism along with a RREP Table. Nevertheless, the description in this section only refers to RREQ multicast messages.

Protocol handling of RERR messages eliminates the need for tracking RERR messages, since the rules for RERR regeneration prevent the phenomenon of redundant retransmission that affects RREQ and RREP multicast.

## 6. AODVv2 Operations on Route Table Entries

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

### 6.1. Evaluating Incoming Routing Information

If the incoming RteMsg does not have a MetricType Message TLV, then the metric information contained by RteMsg is considered to be of type DEFAULT\_METRIC\_TYPE -- which is 3 (for HopCount) unless changed by administrative action. Whenever an AODVv2 router (HandlingRtr) handles an incoming RteMsg (i.e., RREQ or RREP), for every relevant address (RteMsg.Addr) in the RteMsg, HandlingRtr searches its route table to see if there is a route table entry with the same MetricType of the RteMsg, matching RteMsg.Addr. If not, HandlingRtr creates a route table entry for RteMsg.Addr as described in Section 6.2. Otherwise, HandlingRtr compares the incoming routing information in RteMsg against the already stored routing information in the route table entry (Route) for RteMsg.Addr, as described below.

Suppose Route[RteMsg.Addr] uses the same metric type as the incoming routing information, and the route entry contains Route.SeqNum, Route.Metric, and Route.Broken. Suppose the incoming routing information for Route.Addr is RteMsg.SeqNum and RteMsg.Metric. Define RteMsg.Cost to be (RteMsg.Metric + Cost(L)), where L is the incoming link. The incoming routing information is classified as follows:

1. Stale:: RteMsg.SeqNum < Route.SeqNum :  
If RteMsg.SeqNum < Route.SeqNum the incoming information is stale. Using stale routing information is not allowed, since that might result in routing loops. HandlingRtr MUST NOT update the route table entry using the routing information for RteMsg.Addr.
2. Unsafe against loops:: (TRUE != LoopFree (RteMsg, Route)) :  
If RteMsg is not Stale (as in (1) above), RteMsg.Cost is next considered to insure loop freedom. If (TRUE != LoopFree (RteMsg, Route)) (see Section 5.6), then the incoming RteMsg information is not guaranteed to prevent routing loops, and it MUST NOT be used to update any route table entry.
3. More costly::  
(RteMsg.Cost >= Route.Metric) && (Route.Broken==FALSE)  
When RteMsg.SeqNum is the same as in a valid route table entry, and LoopFree (RteMsg, Route) assures loop freedom, incoming information still does not offer any improvement over the existing route table information if RteMsg.Cost >= Route.Metric. Using such incoming routing information to update a route table entry is not recommended.

#### 4. Offers improvement::

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

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

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

- \* it is more recent, or
- \* it is not stale and is less costly, or
- \* it can safely repair a broken route.

#### 6.2. Applying Route Updates To Route Table Entries

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

- o Route.Address := RteMsg.Addr
- o If RteMsg.PrefixLength exists and is not INVALID\_PREFIX\_LENGTH, then Route.PrefixLength := RteMsg.PrefixLength
- o Route.SeqNum := RteMsg.SeqNum
- o Route.NextHopAddress := IP.SourceAddress (i.e., an address of the node from which the RteMsg was received)
- o Route.NextHopInterface is set to the interface on which RteMsg was received
- o Route.Broken flag := FALSE
- o If RteMsg.MetricType is included, then Route.MetricType := RteMsg.MetricType. Otherwise, Route.MetricType := DEFAULT\_METRIC\_TYPE.

- o `Route.Metric := (RteMsg.Metric + Cost(L))`, where `L` is the incoming link.
- o `Route.LastUsed := Current_Time`
- o If `RteMsg.VALIDITY_TIME` is included, then  
    `Route.ExpirationTime := Current_Time + RteMsg.VALIDITY_TIME`,  
    otherwise, `Route.ExpirationTime := Current_Time + (ACTIVE_INTERVAL + MAX_IDLETIME)`.

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

### 6.3. Route Table Entry Timeouts

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

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

- o An Active route **MUST NOT** be expunged.
- o An Idle route **SHOULD NOT** be expunged.
- o An Expired route **MAY** be expunged (least recently used first).
- o A route **MUST** be expunged if  $(Current\_Time - Route.LastUsed) \geq MAX\_SEQNUM\_LIFETIME$ .
- o A route **MUST** be expunged if  $Current\_Time \geq Route.ExpirationTime$

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

## 7. Routing Messages RREQ and RREP (RteMsgs)

AODVv2 message types RREQ and RREP are together known as Routing

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

When an AODVv2 router needs to forward a data packet from a node (OrigNode) in its set of router clients, and it does not have a forwarding route toward the packet's IP destination address (TargNode), the AODVv2 router (RREQ\_Gen) generates a RREQ (as described in Section 7.3) to discover a route toward TargNode. Subsequently RREQ\_Gen awaits reception of an RREP message (see Section 7.4) or other route table update (see Section 6.2) to establish a route toward TargNode. Optionally, RREQ\_Gen MAY specify that only the router serving TargNode is allowed to generate an RREP message, by including the DestOnly message TLV (see Section 7.3). The RREQ message contains routing information to enable RREQ recipients to route packets back to OrigNode, and the RREP message contains routing information enabling RREP recipients to route packets to TargNode.

#### 7.1. Route Discovery Retries and Buffering

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

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

Data packets awaiting a route SHOULD be buffered by RREQ\_Gen. This buffer SHOULD have a fixed limited size (BUFFER\_SIZE\_PACKETS or BUFFER\_SIZE\_BYTES). Determining which packets to discard first is a matter of policy at each AODVv2 router; in the absence of policy constraints, by default older data packets SHOULD be discarded first. Buffering of data packets can have both positive and negative effects (albeit usually positive). Nodes without sufficient memory available for buffering SHOULD be configured to disable buffering by

configuring `BUFFER_SIZE_PACKETS == 0` and `BUFFER_SIZE_BYTES == 0`. Doing so will affect the latency required for launching TCP applications to new destinations.

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

## 7.2. RteMsg Structure

RteMsgs have the following general format:

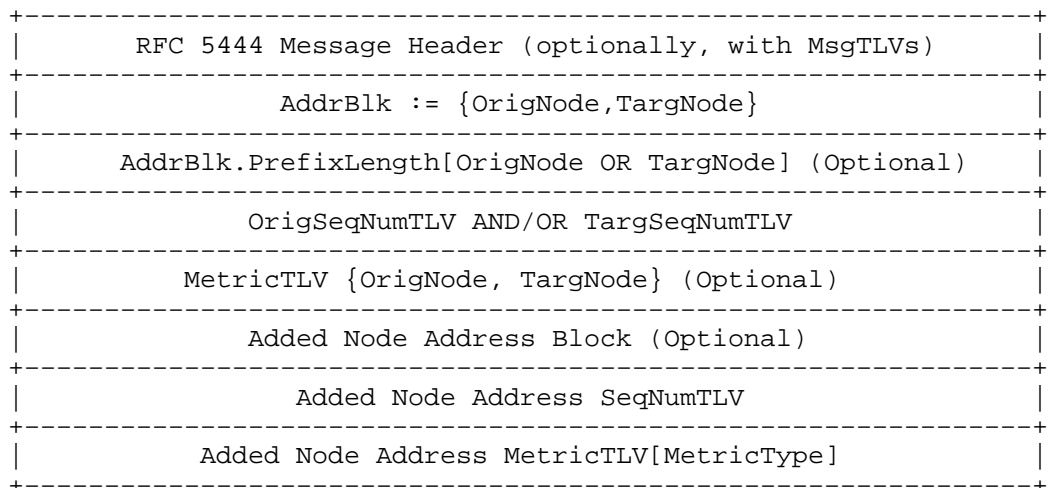


Figure 1: RREQ and RREP (RteMsg) message structure

### Required Message Header Fields

The RteMsg MUST contain the following:

- \* <msg-hop-limit>
- \* Metric Type Message TLV, if `MetricType != 3`



## Optional Message Header Fields

The RteMsg may contain the following:

- \* <msg-hop-count>
- \* DestOnly TLV (RREQ only: no Intermediate RREP)
- \* MetricType TLV (Metric Type for Metric AddrTLV)
- \* AckReq TLV (Acknowledgement Requested)

## AddrBlk

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

## OrigSeqNum AND/OR TargSeqNum AddrTLV

At least one of OrigSeqNum or TargSeqNum Address Block TLV is REQUIRED and carries the destination sequence numbers associated with either OrigNode or TargNode. Both may appear when SeqNum information is available for both OrigNode and TargNode.

## (Optional) Added Node AddrBlk

AODVv2 allows the inclusion of routing information for other nodes in addition to OrigNode and TargNode.

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

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

RteMsgs carry information about OrigNode and TargNode. Since their addresses may appear in arbitrary order within the RFC 5444 AddrBlk, the OrigSeqNum and/or TargSeqNum TLVs must be used to distinguish the nature of the node addresses present in the AddrBlk. In each RteMsg, at least one of OrigSeqNumTLV or TargSeqNumTLV MUST appear. Both TLVs MAY appear in the same RteMsg, but each one MUST NOT appear more than once, because there is only one OrigNode and only one TargNode address in the AddrBlk.

If the OrigSeqNum TLV appears, then the address range for the OrigSeqNum TLV MUST be limited to a single position in the AddrBlk. That position is used as the OrigNdx, identifying the OrigNode address. The other address in the AddrBlk is, by elimination, the

TargNode address, and TargNdx is set appropriately.

Otherwise, if the TargSeqNum TLV appears, then the address range for the TargSeqNum TLV MUST be limited to a single position in the AddrBlk. That position is used as the TargNdx, identifying the TargNode address. The other address in the AddrBlk is, by elimination, the OrigNode address, and OrigNdx is set appropriately.

### 7.3. RREQ Generation

The AODVv2 router generating the RREQ (RREQ\_Gen) on behalf of its client OrigNode follows the steps in this section. OrigNode MUST be a unicast address. The order of protocol elements is illustrated schematically in Figure 1.

1. RREQ\_Gen MUST increment its SeqNum by one (1) according to the rules specified in Section 5.5. This assures that each node receiving the RREQ will update its route table using the information in the RREQ.
2. If RREQ\_Gen requires that only the router providing connectivity to TargNode is allowed to generate a RREP, then RREQ\_Gen includes the "Destination RREP Only" (DestOnly) TLV as part of the RFC 5444 message header. This also assures that RREP\_Gen increments its sequence number. Otherwise, (if the optional behavior is enabled) other AODVv2 routers MAY respond to the RREQ if they have a valid route to TargNode (see Section 13.2).
3. <msg-hop-limit> SHOULD be set to MAX\_HOPCOUNT.
4. <msg-hop-count>, if included, MUST be set to 0.
  - \* This RFC 5444 constraint causes the typical RREQ payload to incur additional enlargement (otherwise, <msg-hop-count> could often be used as the metric).
5. RREQ.AddrBlk := {OrigNode.Addr, TargNode.Addr}  
  
Let OrigNodeNdx and TargNodeNdx denote the indexes of OrigNode and TargNode respectively in the RREQ.AddrBlk list.
6. If Route[OrigNode].PrefixLength/8 is equal to the number of bytes in the addresses of the RREQ (4 for IPv4, 16 for IPv6), then no <prefix-length> is included with the RREQ.AddrBlk. Otherwise, RREQ.PrefixLength[OrigNodeNdx] := Route[OrigNode].PrefixLength according to the rules of RFC 5444 AddrBlk encoding.

7. RREQ.OrigSeqNumTLV[OrigNodeNdx] := RREQ\_Gen SeqNum
8. RREQ.TargSeqNumTLV[TargNodeNdx] := TargNode SeqNum (only if known)

RREQ\_Gen SHOULD include TargNode's SeqNum, if a previous value of the TargNode's SeqNum is known (e.g., from an invalid routing table entry using longest-prefix matching). If TargNode's SeqNum is not included, AODVv2 routers handling the RREQ assume that RREQ\_Gen does not have that information. If ENABLE\_IRREP is enabled, then any route to TargNode will satisfy the RREQ [I-D.perkins-irrep].

9. RREQ.MetricTLV[1] := Route[OrigNode].Metric

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

#### 7.4. RREP Generation

This section specifies the generation of an RREP by an AODVv2 router (RREP\_Gen) that provides connectivity for the Target Node (TargNode) of a RREQ, thus enabling the establishment of a route between OrigNode and TargNode. If TargNode is not a unicast IP address the RREP MUST NOT be generated, and processing for the RREQ is complete. Before transmitting a RREP, the routing information of the RREQ is processed as specified in Section 6.2; after such processing, RREP\_Gen has an updated route to OrigNode as well as TargNode. The basic format of an RREP conforms to the structure for RteMsgs as shown in Figure 1.

RREP\_Gen generates the RREP as follows:

1. RREP\_Gen checks the RREQ against recently received RREQ information as specified in Section 7.6. If a previously received RREQ has made the information in the incoming RREQ to be redundant, no RREP is generated and processing is complete.
2. RREP\_Gen MUST increment its SeqNum by one (1) according to the rules specified in Section 5.5.
3. RREP.AddrBlk := {OrigNode.Addr, TargNode.Addr}

Let OrigNodeNdx and TargNodeNdx denote the indexes of OrigNode and TargNode respectively in the RREP.AddrBlk list.

4. RREP.OrigSeqNumTLV[OrigNodeNdx] := Route[OrigNode].Seqnum

5. `RREP.TargSeqNumTLV[TargNodeNdx] := RREP_Gen's SeqNum`
6. If `Route[TargNode].PrefixLength/8` is equal to the number of bytes in the addresses of the RREQ (4 for IPv4, 16 for IPv6), then no `<prefix-length>` is included with the `RREP.AddrBlk`. Otherwise, `RREP.PrefixLength[TargNodeNdx] := Route[TargNode].PrefixLength` according to the rules of RFC 5444 AddrBlk encoding.
7. `RREP.MetricType[TargNodeNdx] := Route[TargNode].MetricType`
8. `RREP.Metric[TargNodeNdx] := Route[TargNode].Metric`
9. `<msg-hop-count>`, if included, MUST be set to 0.
10. `<msg-hop-limit>` SHOULD be set to `RREQ.<msg-hop-count>`.
11. `IP.DestinationAddr := Route[OrigNode].NextHop`

An example message format for RREP is illustrated in Appendix A.2.

#### 7.5. Handling a Received RteMsg

Before an AODVv2 router can make use of a received RteMsg (i.e., RREQ or RREP), the router first must verify that the RteMsg is permissible according to the following steps. `OrigNodeNdx` and `TargNodeNdx` are set according to the rules in Section 7.2. For RREQ, `RteMsg.Metric` is `MetricTLV[OrigNodeNdx]`. For RREP, `RteMsg.Metric` is `MetricTLV[TargNodeNdx]`. In this section (unless qualified by additional description such as "upstream" or "neighboring") all occurrences of the term "router" refer to the AODVv2 router handling the received RteMsg.

1. A router MUST handle RteMsgs only from neighbors as specified in Section 5.4. RteMsgs from other sources MUST be disregarded.
2. The router examines the RteMsg to ascertain that it contains the required information: `<msg-hop-limit>`, `TargNode.Addr`, `OrigNode.Addr`, `RteMsg.Metric`, and either `RteMsg.OrigSeqNum` or `RteMsg.TargSeqNum`. If the required information does not exist, the message is disregarded.
3. The router checks that `OrigNode.Addr` and `TargNode.Addr` are valid routable unicast addresses. If not, the message is disregarded.
4. The router checks the Metric Type MsgTLV (if present) to assure that the Metric Type associated with the Metric AddrTLV information in the RREQ or RREP is known, and that `Cost(L)` can be

computed, where 'L' is the incoming link. If not, the message is disregarded.

\* DISCUSSION: or, can change the AddrBlk metric to use HopCount, e.g., measured from <msg-hop-count>.

5. If  $(\text{MAX\_METRIC}[\text{RteMsg.MetricType}] - \text{Cost}(L)) \leq \text{RteMsg.Metric}$ , the RteMsg is disregarded, where  $\text{Cost}(L)$  denotes the cost of traversing the incoming link (i.e., as measured by the network interface receiving the incoming RteMsg).

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

1. The router MUST process the routing information for OrigNode and TargNode contained in the RteMsg as specified in Section 6.1.
2. The router MAY process AddedNode routing information (if present) as specified in Section 13.7.1. Otherwise, if AddedNode information is not processed, it MUST be deleted, because it may no longer be accurate as a route update to any upstream router.
3. If  $\text{RteMsg}.\text{<msg-hop-limit>}$  is zero (0), no further action is taken, and the RteMsg is not retransmitted. Otherwise, the router MUST decrement  $\text{RteMsg}.\text{<msg-hop-limit>}$ .
4. If the  $\text{RteMsg}.\text{<msg-hop-count>}$  is present, and  $\text{<msg-hop-count>} == \text{MAX\_HOPCOUNT}$ , then no further action is taken. Otherwise, the router MUST increment  $\text{RteMsg}.\text{<msg-hop-count>}$ .

Further actions to transmit an updated RteMsg depend upon whether the incoming RteMsg is an RREP or an RREQ.

#### 7.5.1. Additional Handling for Incoming RREQ

- o By sending a RREQ, a router advertises that it will route for addresses contained in the RteMsg based on the information enclosed. The router MAY choose not to send the RREQ, though not resending the RREQ could decrease connectivity in the network or result in nonoptimal paths. The circumstances under which a router might choose not to re-transmit a RREQ are not specified in this document. Some examples might include the following:
  - \* The router is already heavily loaded and does not want to advertise routing for more traffic
  - \* The router recently transmitted identical routing information (e.g. in a RREQ advertising the same metric) Section 7.6

- \* The router is low on energy and has to reduce energy expended for sending protocol messages or packet forwarding

Unless the router is prepared to send a RREQ, it halts processing.

- o If the upstream router sending a RREQ is in the Blacklist, and `Current_Time < Blacklist.RemoveTime`, then the router receiving that RREQ MUST NOT transmit any outgoing RteMsg, and processing is complete.
- o Otherwise, if the upstream router is in the Blacklist, and `Current_Time >= Blacklist.RemoveTime`, then the upstream router SHOULD be removed from the Blacklist, and message processing continued.
- o The incoming RREQ MUST be checked against previously received information from the RREQ Table Section 7.6. If the information in the incoming RteMsg is redundant, then then no further action is taken.
- o If TargNode is a client of the router receiving the RREQ, then the router generates a RREP message as specified in Section 7.4, and subsequently processing for the RREQ is complete. Otherwise, processing continues as follows.
- o `RREQ.MetricType := Route[OrigNode].MetricType`
- o `RREQ.MetricTLV[OrigNodeNdx] := Route[OrigNode].Metric`
- o The RREQ (with updated fields as specified above) SHOULD be sent to the IP multicast address LL-MANET-Routers [RFC5498]. If the RREQ is unicast, the `IP.DestinationAddress` is set to `Route[RREQ.TargNode].NextHopAddress`.

#### 7.5.2. Additional Handling for Incoming RREP

As before, `OrigNode` and `TargNode` are named in the context of `RREQ_Gen` (i.e., the router originating the RREQ for which the RREP was generated) (see Table 1). `OrigNodeNdx` and `TargNodeNdx` are set according to the rules in Section 7.2.

- o If no forwarding route exists to `OrigNode`, then a RERR SHOULD be transmitted to `RREP.AddrBlk[TargNodeNdx]`. Otherwise, if `HandlingRtr` is not `RREQ_Gen` then the outgoing RREP is sent to the `Route.NextHopAddress` for the `RREP.AddrBlk[OrigNodeNdx]`.
- o If `HandlingRtr` is `RREQ_Gen` then the RREP satisfies `RREQ_Gen`'s earlier RREQ, and RREP processing is completed. Any packets

buffered for OrigNode should be transmitted.

#### 7.6. Suppressing Redundant RREQ messages

Since RREQ messages are multicast, there are common circumstances in which an AODVv2 router might transmit a redundant response (RREQ or RREP), duplicating the information transmitted in response to some other recent RREQ (see Section 5.7). Before responding, an AODVv2 router MUST suppress such redundant RREQ messages. This is done by checking the list of recently received RREQs to determine whether the incoming RREQ contains new information, as follows:

- o The AODVv2 router searches the RREQ Table for recent entries with the same OrigNode, TargNode, and Metric Type. If there is no such entry, the incoming RREQ message is not suppressed. A new entry for the incoming RREQ is created in the RREQ Table.
- o If there is such an entry, and the incoming RREQ has a newer sequence number, the incoming RREQ is not suppressed, and the existing table entry MUST be updated to reflect the new Sequence Number and Metric.
- o Similarly, if the Sequence Numbers are the same, and the incoming RREQ offers a better Metric, the incoming RREQ is not suppressed, and the RREQ Table entry MUST be updated to reflect the new Metric.
- o Otherwise, the incoming RREQ is suppressed.

#### 8. Route Maintenance and RERR Messages

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

##### 8.1. Maintaining Route Lifetimes During Packet Forwarding

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

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

If `Current_Time > Route.ExpirationTime`, the route table entry has expired, and cannot be used for forwarding.

Similarly, if `(Route.ExpirationTime == MAXTIME)`, and if `(Current_Time - Route.LastUsed) > (ACTIVE_INTERVAL + MAX_IDLETIME)`, the route has expired, and cannot be used for forwarding.

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

If any of the above route error conditions hold true, the route cannot be used to forward the packet, and an RERR message MUST be generated (see Section 8.3).

Otherwise, `Route.LastUsed := Current_Time`, and the packet is forwarded to the route's next hop.

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

## 8.2. Active Next-hop Router Adjacency Monitoring

AODVv2 routers SHOULD monitor connectivity to adjacent routers along active routes. This monitoring can be accomplished by one or several mechanisms, including:

- o Neighborhood discovery [RFC6130]
- o Route timeout
- o Lower layer trigger that a link is broken
- o TCP timeouts
- o Promiscuous listening
- o Other monitoring mechanisms or heuristics

If a next-hop AODVv2 router has become unreachable, RERR\_Gen follows the procedures specified in Section 8.3.2.



### 8.3. RERR Generation

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

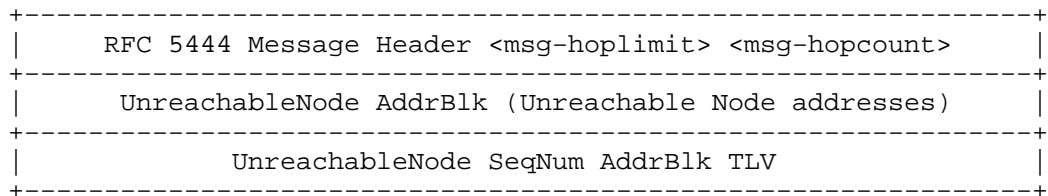


Figure 2: RERR message structure

#### Required Message Header Fields

The RERR MUST contain the following:

- \* <msg-hop-limit>
- \* PktSource Message TLV (see Section 15), if the RERR is unicast
- \* Metric Type Message TLV (see Section 15), if MetricType != 3

#### Optional Message Header Fields

The RERR may contain the following:

- \* <msg-hop-count>

#### UnreachableNode AddrBlk

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

#### Sequence Number AddrBlk TLV

This Address Block TLV carries the destination sequence number associated with each UnreachableNode when that information is available.

#### UnreachableNode.PrefixLength

The prefix length associated with an UnreachableNode.

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

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

#### 8.3.1. Case 1: Undeliverable Packet

The first case happens when the router receives a packet from another AODVv2 router but does not have a valid route for the destination of the packet. In this case, there is exactly one UnreachableNode to be included in the RERR's AddrBlk (either IP.DestinationAddress from a data packet or RREP.AddrBlk[OrigNode]). The RERR SHOULD be sent to the multicast address LL-MANET-Routers, but RERR\_Gen MAY instead send the RERR to the next hop towards the source IP address of the packet which was undeliverable. For unicast RERR, the PktSource Message TLV MUST be included, containing the the source IP address of the undeliverable packet, or the IP address of TargRtr in case the undeliverable packet was an RREP message generated by TargRtr. If a Sequence Number for UnreachableNode is known, that Sequence Number SHOULD be included in a Seqnum AddrTLV the RERR. Otherwise all nodes handling the RERR will assume their route through RERR\_Gen towards the UnreachableNode is no longer valid and flag those routes as broken, regardless of the Sequence Number information for those routes. RERR\_Gen MUST discard the packet or message that triggered generation of the RERR.

If an AODVv2 router receives an ICMP packet from the address of one of its client nodes, it simply relays the packet to the ICMP packet's destination address, and does not generate any RERR message.

#### 8.3.2. Case 2: Broken Link

The second case happens when the link breaks to an active adjacent AODVv2 router (i.e., the next hop of an active route). In this case, the RERR MUST be sent to the multicast address LL-MANET-Routers, except when the optional feature of maintaining precursor lists is used as specified in Section 13.3. All routes (Active, Idle and Expired) that use the broken link MUST be marked as Broken. The set of UnreachableNodes is initialized by identifying those Active routes which use the broken link. For each such Active Route, Route.Dest is added to the set of Unreachable Nodes. After the Active Routes using the broken link have all been included as UnreachableNodes, Idle routes MAY also be included, if allowed by the setting of ENABLE\_IDLE\_UNREACHABLE, as long as the packet size of the RERR does not exceed the MTU (interface "Maximum Transfer Unit") of the physical medium.

If the set of UnreachableNodes is empty, no RERR is generated.

Otherwise, RERR\_Gen generates a new RERR, and the address of each UnreachableNode is inserted into an AddrBlock. If a prefix is known for the UnreachableNode.Address, it SHOULD be included. Otherwise, the UnreachableNode.Address is assumed to be a host address with a full length prefix. The value for each UnreachableNode's SeqNum (UnreachableNode.SeqNum) MUST be placed in a SeqNum AddrTLV. If none of UnreachableNode.Addr entries are associated with known prefix lengths, then the AddrBlk SHOULD NOT include any prefix-length information. Otherwise, for each UnreachableNode.Addr that does not have any associated prefix-length information, the prefix-length for that address MUST be assigned to INVALID\_PREFIX\_LENGTH, which is a length strictly greater than the length of any valid address.

Every broken route reported in the RERR MUST have the same Metric Type. If the Metric Type is not 3, then the RERR message MUST contain a MetricType MsgTLV indicating the Metric Type of the broken route(s).

#### 8.4. Receiving and Handling RERR Messages

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

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

1. The UnreachableNode.Address is a routable unicast address.
2. Route.NextHopAddress is the same as RERR IP.SourceAddress.
3. Route.NextHopInterface is the same as the interface on which the RERR was received.
4. The UnreachableNode.SeqNum is unknown, OR Route.SeqNum <= UnreachableNode.SeqNum (using signed 16-bit arithmetic).

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

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

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

- o (Optional) If precursor lists are maintained, the outgoing RERR SHOULD be sent to the active precursors of the broken route as specified in Section 13.3.
- o Otherwise, if the incoming RERR message was received at the LL-MANET-Routers [RFC5498] multicast address, the outgoing RERR SHOULD also be sent to LL-MANET-Routers.
- o Otherwise, if the PktSource Message TLV is present, and HandlingRtr has a Route to PktSource.Addr, then HandlingRtr MUST send the outgoing RERR to Route[PktSource.Addr].NextHop.
- o Otherwise, the outgoing RERR MUST be sent to LL-MANET-Routers.

## 9. Unknown Message and TLV Types

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

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

## 10. Simple Internet Attachment

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

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

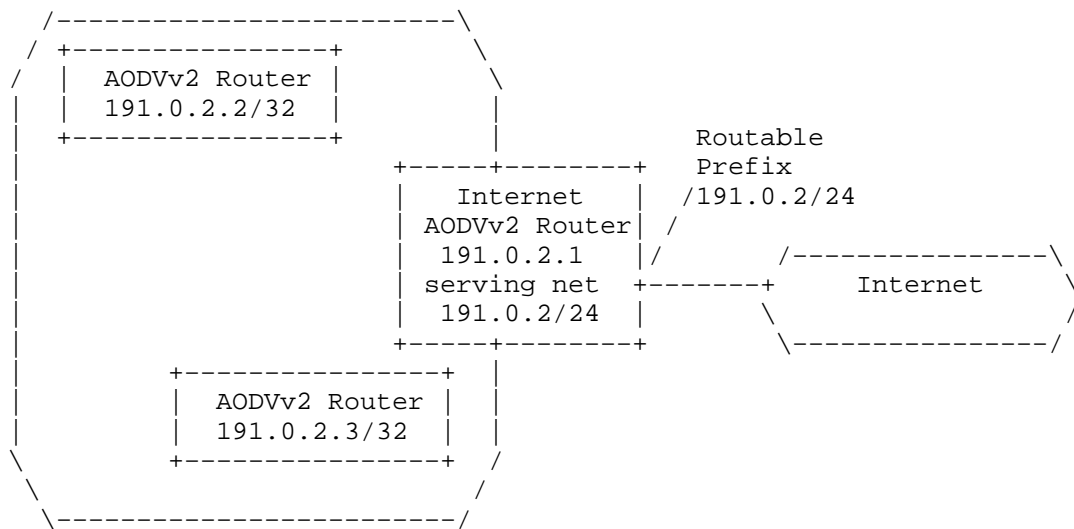


Figure 3: Simple Internet Attachment Example

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

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

## 11. Multiple Interfaces

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

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

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

## 12. AODVv2 Control Packet/Message Generation Limits

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

## 13. Optional Features

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

- o Expanding Rings Multicast
- o Intermediate RREPs (irREPs): Without irREP, only the destination can respond to a RREQ.
- o Precursor lists.
- o Reporting Multiple Unreachable Nodes. An RERR message can carry more than one Unreachable Destination node for cases when a single link breakage causes multiple destinations to become unreachable from an intermediate router.
- o RREP\_ACK.
- o Message Aggregation.
- o Inclusion of Added Routing Information.

### 13.1. Expanding Rings Multicast

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

### 13.2. Intermediate RREP

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

### 13.3. Precursor Lists and Notifications

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

#### 13.3.1. Overview

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

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

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

#### 13.3.2. Precursor Notification Details

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

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

unicast RERR to each Active precursor

This option is applicable when there are few Active precursors compared to the number of neighboring AODVv2 routers.

multicast RERR to RERR\_PRECURSORS

RERR\_PRECURSORS is, by default, LL-MANET-Routers [RFC5498]. This option is typically preferable when there are many precursors, since fewer packet transmissions are required.

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

#### 13.4. Multicast RREP Response to RREQ

The RREQ Target Router (RREP\_Gen) MAY, as an alternative to unicasting a RREP, be configured to distribute routing information about the route toward the RREQ TargNode (RREP\_Gen's client) more widely. That is, RREP\_Gen MAY be configured respond to a route discovery by generating a RREP, using the procedure in Section 7.4, but multicasting the RREP to LL-MANET-Routers [RFC5498] (subject to similar suppression algorithm for redundant RREP multicasts as described in Section 7.6). The redundant message suppression must occur at every router handling the multicast RREP. Afterwards, RREP\_Gen processing for the incoming RREQ is complete.

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

#### 13.5. RREP\_ACK

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

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



### 13.6. Message Aggregation

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

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

### 13.7. Added Routing Information in RteMsgs

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

Including additional routing information in outgoing RREQ or RREP messages can eliminate some route discovery attempts to the nodes whose information is included, if AODVv2 routers receiving the information use it to update their routing tables.

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

#### 13.7.1. Including Added Node Information

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

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

##### AddedNode

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

**AddedNode.SeqNum**

The Sequence Number associated with the AddedNode's routing information.

**AddedNode.Metric**

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

The VALIDITY\_TIME of routing information for appended address(es) MUST be included, to inform routers about when to expire this information. A typical value for VALIDITY\_TIME is (ACTIVE\_INTERVAL+MAX\_IDLETIME) - (Current\_Time - Route.LastUsed) but other values (less than MAX\_SEQNUM\_TIME) MAY be chosen. The VALIDITY\_TIME TLV is defined in [RFC5497].

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

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

### 13.7.2. Handling Added Node Information

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

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

needs to be forwarded to the IP destination(s) now or in the near future.

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

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

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

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

#### 14. Administratively Configurable Parameters and Timer Values

AODVv2 uses various configurable parameters of various types:

- o Timers
- o Protocol constants
- o Administrative (functional) controls
- o Other administrative parameters and lists

The tables in the following sections show the parameters along their definitions and default values (if any).

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

## 14.1. Timers

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

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

Table 2: Timing Parameter Values

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

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

## 14.2. Protocol constants

AODVv2 protocol constants typically do not require changes. The following table lists these constants, along with their values and a reference to the specification describing their use.

Name	Default Value	Description
DISCOVERY_ATTEMPTS_MAX	3	Section 7.1
INVALID_PREFIX_LENGTH	255	Section 8.3.2
MAX_HOPCOUNT	20 hops	Section 5.6
MAX_METRIC[i]	Specified only for HopCount	Section 5.6
MAXTIME	[TBD]	Maximum expressible clock time

Table 3: Parameter Values

## 14.3. Administrative (functional) controls

The following administrative controls may be used to change the operation of the network, by enabling optional behaviors. These options are not required for correct routing behavior, although they may potentially reduce AODVv2 protocol messaging in certain situations. The default behavior is to NOT enable most such options, options. Packet buffering is enabled by default.

Name	Description
APPEND_INFORMATION	Section 13.7.1
DEFAULT_METRIC_TYPE	3 {Hop Count (see [RFC6551])}
ENABLE_IDLE_UNREACHABLE	Section 8.3.2
ENABLE_IRREP	Section 7.3
USE_MULTICAST_RREP	Section 13.4

Table 4: Administratively Configured Controls

## 14.4. Other administrative parameters and lists

The following table lists contains AODVv2 parameters which should be administratively configured for each specific network.

Name	Default Value	Cross Reference
AODVv2_INTERFACES		Section 4
BUFFER_SIZE_PACKETS	2	Section 7.1
BUFFER_SIZE_BYTES	MAX_PACKET_SIZE [TBD]	Section 7.1
CLIENT_ADDRESSES	AODVv2_INTERFACES	Section 5.3
CONTROL_TRAFFIC_LIMIT	TBD [50 packets/sec?]	Section 12

Table 5: Other Administrative Parameters

## 15. IANA Considerations

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

## 15.1. AODVv2 Message Types Specification

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

Table 6: AODVv2 Message Types

## 15.2. Message TLV Type Specification

Name	Type (TBD)	Length in octets	Cross Reference
Acknowledgment Request (AckReq)	10	0	Section 5.2
Destination RREP Only (DestOnly)	11	0	Section 7.3
Packet Source (PktSource)	12	4 or 16	Section 8.3
Metric Type	13	1	Section 7.2

Table 7: Message TLV Types

## 15.3. Address Block TLV Specification

Name	Type (TBD)	Length	Value
Metric	10	depends on Metric Type	Section 7.2
Sequence Number (SeqNum)	11	2 octets	Section 7.2
Originating Node Sequence Number (OrigSeqNum)	12	2 octets	Section 7.2
Target Node Sequence Number (TargSeqNum)	13	2 octets	Section 7.2
VALIDITY_TIME	1	1 octet	[RFC5497]

Table 8: Address Block TLV (AddrTLV) Types

#### 15.4. Metric Type Number Allocation

Metric types are identified according to the assignments as specified in [RFC6551]. The metric type of the Hop Count metric is assigned to be 3, in order to maintain compatibility with that existing table of values from RFC 6551. Non-additive metrics are not supported in this draft.

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

Table 9: Metric Types

#### 16. Security Considerations

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

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

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

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

A digital signature could be used to identify the source of AODVv2 messages and information, along with its authenticity. A nonce or timestamp SHOULD also be used to protect against replay attacks.

S/MIME and OpenPGP are two authentication/integrity protocols that could be adapted for this purpose.

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

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

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

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

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

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

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



security that AODVv2 neighbors exchange security information that can be used to insert an ICV [RFC6621] into the AODVv2 message block [RFC5444]. This enables hop-by-hop security, which is proper for these message types that may have mutable fields. For destination-only RREP discovery procedures, AODVv2 routers that share a security association SHOULD use the appropriate mechanisms as specified in RFC 6621. The establishment of these security associations is out of scope for this document.

## 17. Acknowledgments

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

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

## 18. References

### 18.1. Normative References

- [RFC1812] Baker, F., "Requirements for IP Version 4 Routers", RFC 1812, June 1995.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, October 2007.

- [RFC5444] Clausen, T., Dearlove, C., Dean, J., and C. Adjih, "Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format", RFC 5444, February 2009.
- [RFC5497] Clausen, T. and C. Dearlove, "Representing Multi-Value Time in Mobile Ad Hoc Networks (MANETs)", RFC 5497, March 2009.
- [RFC5498] Chakeres, I., "IANA Allocations for Mobile Ad Hoc Network (MANET) Protocols", RFC 5498, March 2009.
- [RFC6551] Vasseur, JP., Kim, M., Pister, K., Dejean, N., and D. Barthel, "Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks", RFC 6551, March 2012.

## 18.2. Informative References

- [I-D.clausen-lln-loadng]  
Clausen, T., Verdiere, A., Yi, J., Niktash, A., Igarashi, Y., Satoh, H., Herberg, U., Lavenu, C., Lys, T., Perkins, C., and J. Dean, "The Lightweight On-demand Ad hoc Distance-vector Routing Protocol - Next Generation (LOADng)", draft-clausen-lln-loadng-08 (work in progress), January 2013.
- [I-D.perkins-irrep]  
Perkins, C. and I. Chakeres, "Intermediate RREP for dynamic MANET On-demand (AODVv2) Routing", draft-perkins-irrep-02 (work in progress), November 2012.
- [Perkins99]  
Perkins, C. and E. Belding-Royer, "Ad hoc On-Demand Distance Vector (AODV) Routing", Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, pp. 90-100, February 1999.
- [RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.
- [RFC2501] Corson, M. and J. Macker, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations", RFC 2501, January 1999.
- [RFC3561] Perkins, C., Belding-Royer, E., and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing", RFC 3561, July 2003.
- [RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.

- [RFC4728] Johnson, D., Hu, Y., and D. Maltz, "The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4", RFC 4728, February 2007.
- [RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.
- [RFC5148] Clausen, T., Dearlove, C., and B. Adamson, "Jitter Considerations in Mobile Ad Hoc Networks (MANETs)", RFC 5148, February 2008.
- [RFC5340] Coltun, R., Ferguson, D., Moy, J., and A. Lindem, "OSPF for IPv6", RFC 5340, July 2008.
- [RFC6130] Clausen, T., Dearlove, C., and J. Dean, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)", RFC 6130, April 2011.
- [RFC6549] Lindem, A., Roy, A., and S. Mirtorabi, "OSPFv2 Multi-Instance Extensions", RFC 6549, March 2012.
- [RFC6621] Macker, J., "Simplified Multicast Forwarding", RFC 6621, May 2012.

#### Appendix A. Example RFC 5444-compliant packet formats

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

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

In addition to the Seqnum TLV, there MUST be an AddrTLV of type Metric. The msg-hop-count counts the number of hops followed by the RteMsg from point of generation to the current intermediate AODVv2 router handling the RteMsg. Alternate metrics are enabled by the inclusion of the MetricType Message TLV. When there is no such MetricType Message TLV present, then the Metric AddrTLV measures HopCount. The Metric AddrTLV also provides a way for the AODV router generating the RREQ or RREP to supply an initial nonzero cost for the

route to its client node (OrigNode or TargNode, for RREQ or RREP respectively).

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

In all cases, the length of an address (32 bits for IPv4 and 128 bits for IPv6) inside an AODVv2 message is indicated by the msg-addr-length (MAL) in the msg-header, as specified in [RFC5444].

#### A.1. RREQ Message Format

Figure 4 illustrates a packet format for an example RREQ message.

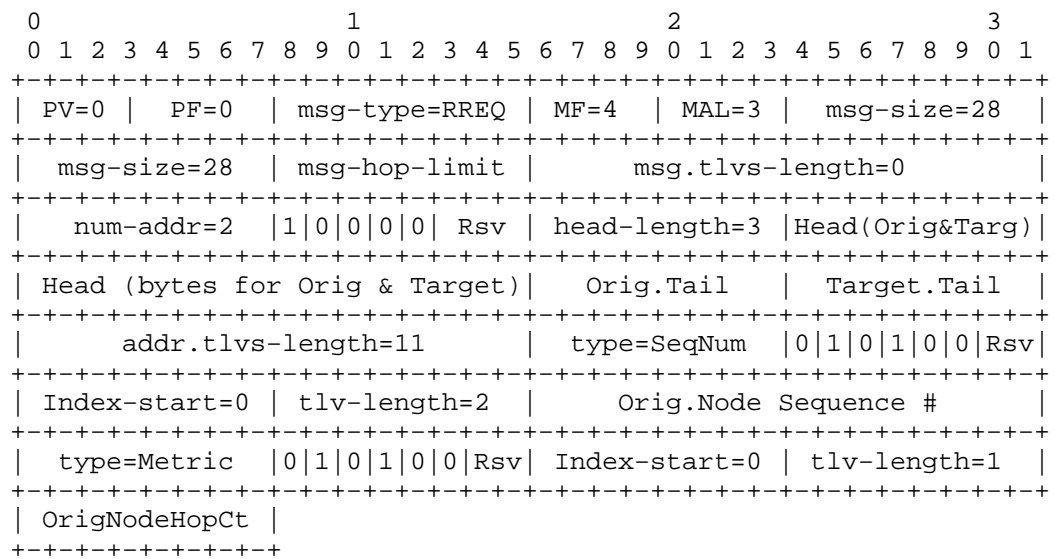


Figure 4: Example IPv4 RREQ, with SeqNum and Metric AddrTLVs

The fields in Figure 4 are to be interpreted as follows:

- o PV=0 (Packet Header Version = 0)
- o PF=0 (Packet Flags = 0)
- o msg-type=RREQ (first [and only] message is of type RREQ)
- o MF=4 (Message Flags = 4 [only msg-hop-limit field is present])
- o MAL=3 (Message Address Length indicator [3 for IPv4, 15 for IPv6])
- o msg-size=28 (octets -- counting MsgHdr, MsgTLVs, and AddrBlks)
- o msg-hop-limit (initially MAX\_HOPCOUNT by default)
- o msg.tlvs-length=0 (no Message TLVs)
- o num-addr=2 (OrigNode and TargNode addresses in RteMsg AddrBlock)
- o AddrBlk flags:
  - \* bit 0 (ahashead): 1
  - \* bit 1 (ahasfulltail): 0
  - \* bit 2 (ahaszerotail): 0
  - \* bit 3 (ahassingleprelen): 0
  - \* bit 4 (ahasmultiprelen): 0
  - \* bits 5-7: RESERVED
- o head-length=3 (length of head part of each address is 3 octets)
- o Head (3 initial bytes for both Originating & Target addresses)
- o Orig.Tail (4th byte of Originating Node IP address)
- o Target.Tail (4th byte of Target Node IP address)
- o addr.tlvs-length=11 (length in bytes for SeqNum and Metric TLVs)
- o type=SeqNum (AddrTLV type of first AddrBlk TLV, values 2 octets)
- o AddrTLV flags for SeqNumTLV:
  - \* bit 0 (thastypeext): 0
  - \* bit 1 (thassingleindex): 1
  - \* bit 2 (thasmultiindex): 0
  - \* bit 3 (thasvalue): 1
  - \* bit 4 (thasextlen): 0
  - \* bit 5 (tismultivalue): 0
  - \* bits 6-7: RESERVED
- o Index-start=0 (SeqNum TLV values start at index 0)
- o tlv-length=2 (so there is only one TLV value, [1 = 2/2])
- o Orig.Node Sequence # (first [and only] TLV value for SeqNum TLVs)
- o type=Metric (AddrTLV type of second AddrBlk TLV, values 1 octet)
- o AddrTLV flags for MetricTLV:
  - \* bit 0 (thastypeext): 0
  - \* bit 1 (thassingleindex): 1
  - \* bit 2 (thasmultiindex): 0
  - \* bit 3 (thasvalue): 1
  - \* bit 4 (thasextlen): 0
  - \* bit 5 (tismultivalue): 0
  - \* bits 6-7: RESERVED
- o Index-start=0 (Metric TLV values start at index 0)
- o tlv-length=1 (so there is only one TLV value, [1 = 1/1])
- o OrigNodeHopCt (first [and only] TLV value for Metric TLVs)

## A.2. RREP Message Format

Figure 5 illustrates a packet format for an example RREP message.

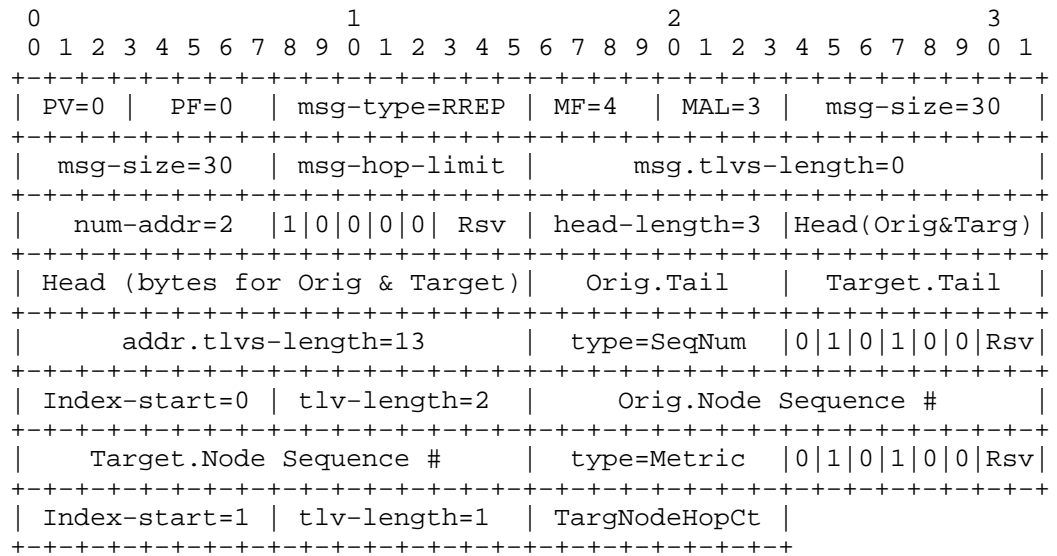


Figure 5: Example IPv4 RREP, with 2 SeqNums and 1 Metric

The fields in Figure 5 are to be interpreted as follows:

- o PV=0 (Packet Header Version = 0)
- o PF=0 (Packet Flags = 0)
- o msg-type=RREP (first [and only] message is of type RREP)
- o MF=4 (Message Flags = 4 [only msg-hop-limit field is present])
- o MAL=3 (Message Address Length indicator [3 for IPv4, 15 for IPv6])
- o msg-size=28 (octets -- counting MsgHdr, MsgTLVs, and AddrBlks)
- o msg-hop-limit (initially MAX\_HOPCOUNT by default)
- o msg.tlvs-length=0 (no Message TLVs)
- o num-addr=2 (OrigNode and TargNode addresses in RteMsg AddrBlock)
- o AddrBlk flags:
  - \* bit 0 (ahashead): 1
  - \* bit 1 (ahasfulltail): 0
  - \* bit 2 (ahaszerotail): 0
  - \* bit 3 (ahassingleprelen): 0
  - \* bit 4 (ahasmultiprelen): 0
  - \* bits 5-7: RESERVED
- o head-length=3 (length of head part of each address is 3 octets)
- o Head (3 initial bytes for both Originating & Target addresses)
- o Orig.Tail (4th byte of Originating Node IP address)
- o Target.Tail (4th byte of Target Node IP address)
- o addr.tlvs-length=13 (length in bytes for SeqNum and Metric TLVs)
- o type=SeqNum (AddrTLV type of first AddrBlk TLV, values 2 octets)
- o AddrTLV flags for SeqNumTLV:
  - \* bit 0 (thastypeext): 0
  - \* bit 1 (thassingleindex): 1
  - \* bit 2 (thasmultiindex): 0
  - \* bit 3 (thasvalue): 1
  - \* bit 4 (thasextlen): 0
  - \* bit 5 (tismultivalue): 0
  - \* bits 6-7: RESERVED
- o Index-start=0 (SeqNum TLV values start at index 0)
- o tlv-length=4 (so there is are two TLV values, [2 = 4/2])
- o Orig.Node Sequence # (first of two TLV values for SeqNum TLVs)
- o Targ.Node Sequence # (second of two TLV values for SeqNum TLVs)
- o type=Metric (AddrTLV type of second AddrBlk TLV, values 1 octet)
- o AddrTLV flags for MetricTLV [01010000, same as for SeqNumTLV]
- o Index-start=1 (Metric TLV values start at index 1)
- o tlv-length=1 (so there is only one TLV value, [1 = 1/1])
- o TargNodeHopCt (first [and only] TLV value for Metric TLVs)

## A.3. RERR Message Format

Figure 6 illustrates a packet format for an example RERR message.

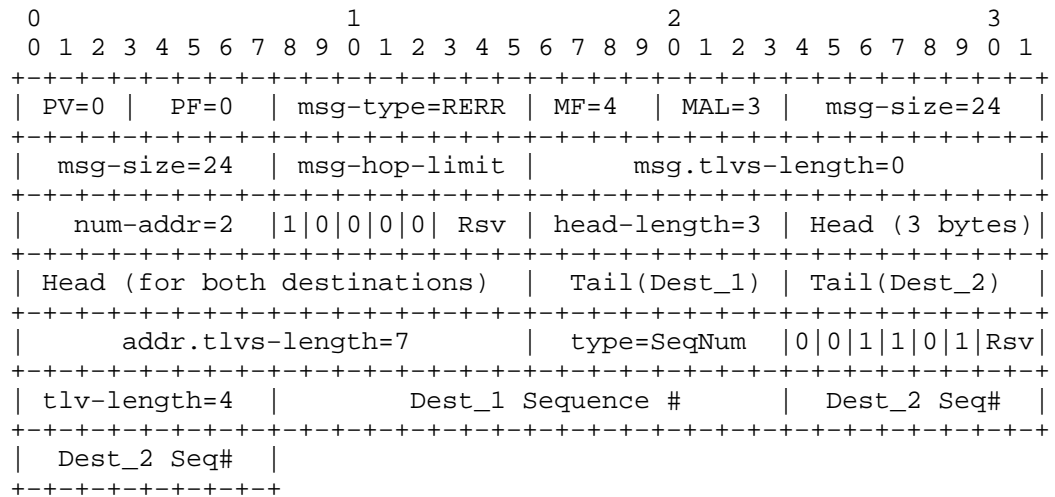


Figure 6: Example IPv4 RERR with Two Unreachable Nodes

The fields in Figure 6 are to be interpreted as follows:

- o PV=0 (Packet Header Version = 0)
- o PF=0 (Packet Flags = 0)
- o msg-type=RERR (first [and only] message is of type RERR)
- o MF=4 (Message Flags = 4 [only msg-hop-limit field is present])
- o MAL=3 (Message Address Length indicator [3 for IPv4, 15 for IPv6])
- o msg-size=24 (octets -- counting MsgHdr, MsgTLVs, and AddrBlks)
- o msg-hop-limit (initially MAX\_HOPCOUNT by default)
- o msg.tlvs-length=0 (no Message TLVs)
- o num-addr=2 (OrigNode and TargNode addresses in RteMsg AddrBlock)
- o AddrBlk flags == 10000000 [same as RREQ and RREP AddrBlk examples]
- o head-length=3 (length of head part of each address is 3 octets)
- o Head (3 initial bytes for both Unreachable Nodes, Dest\_1 and Dest\_2)
- o Dest\_1.Tail (4th byte of Dest\_1 IP address)
- o Dest\_2.Tail (4th byte of Dest\_2 IP address)
- o addr.tlvs-length=7 (length in bytes for SeqNum TLV)
- o type=SeqNum (AddrTLV type of AddrBlk TLV, values 2 octets each)
- o AddrTLV flags for SeqNumTLV:
  - \* bit 0 (thastypeext): 0
  - \* bit 1 (thassingleindex): 0
  - \* bit 2 (thasmultiindex): 1



- \* bit 3 (thasvalue): 1
- \* bit 4 (thasextlen): 0
- \* bit 5 (tismultivalued): 1
- \* bits 6-7: RESERVED
- o Index-start=0 (SeqNum TLV values start at index 0)
- o tlv-length=4 (so there is are two TLV values, [2 = 4/2])
- o Dest\_1 Sequence # (first of two TLV values for SeqNum TLVs)
- o Dest\_2 Sequence # (second of two TLV values for SeqNum TLVs)

#### A.4. RREP\_ACK Message Format

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

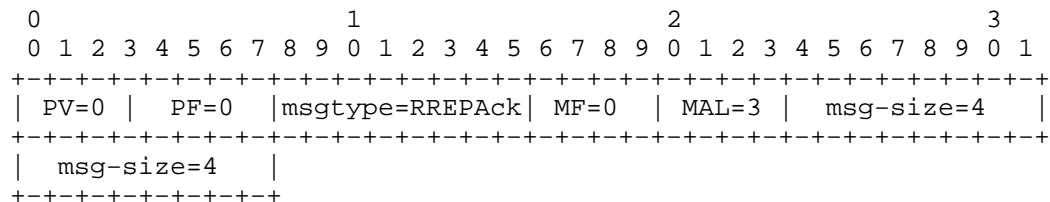


Figure 7: Example IPv4 RREP\_ACK

#### Appendix B. Changes since revision ...-25.txt

The main goals of this revision are to improve readability and to introduce a protocol update which enables order-independent listing of the Originating Node and Target Node (OrigNode and TargNode) in the AddrBlk of RREQ and RREP messages.

- o Added two new AddrTLV types, OrigSeqNum and TargSeqNum. Changed processing description to identify OrigNdx and TargNdx, instead of implicitly assuming OrigNdx = 1 and TargNdx = 2 as in previous versions of the specification. See Section 7.2, Section 7.3, Section 7.4, Section 7.5, and Section 15.3.
- o Reworded initial paragraph of Section 6 to eliminate the use of terminology "DestIP", in order to reduce possible confusion with the meaning of the term "TargNode", etc.
- o Moved description of reasons why a node might not elect to retransmit a RteMsg from Section 7.5 to section Section 7.5.1. If an AODVv2 router would elect to not send an RREP message, it should not send the RREQ message which might elicit that RREP message. Otherwise, valid routes will go undiscovered.
- o Eliminated use of terminology for "Msg." to indicate fields in the RFC 5444 Message Header.

- o Replaced instances of "useless" by "redundant". Made numerous other editorial changes and corrections.
- o Changed membership of editorial team.
- o Formally changed document name to "aodvv2" instead of "dymo".

#### Appendix C. Changes since revision ...-24.txt

The main goals of this revision are to improve readability and to introduce a protocol update to handle suppression of unnecessary multicast RREQs and certain other messages.

- o Specified operations for maintenance and use of RREQ Table (see Section 5.7, Section 7.6).
- o Inserted explanations for example packet formats in appendix (see Appendix A).
- o Eliminated OwnSeqNum, RERR\_dest, and various other abbreviations, reworded relevant text.
- o Reorganized Section 14 into four sections so that the various parameters are grouped more naturally into tables of similar types.
- o Replaced parameter descriptions in the tables in Section 14, with cross references to the parameter descriptions in the body of the specification.
- o Created parameters and administrative controls ENABLE\_IRREP and MAX\_BLACKLIST\_TIME which had been alluded to in the body of the specification.
- o Corrected metric comparison formulae to include cost of incoming link.
- o Renamed Unicast Response Request MsgTLV to be Acknowledgment Request.
- o Clarified <msg-hop-limit> and <msg-hop-count> mandates and initialization.
- o Reformatted various tables to improve readability.
- o Changed some descriptions to apply to "Incoming" messages instead of "Outgoing" messages, enabling simpler specification.
- o Many other minor editorial improvements to improve readability and eliminate possibly ambiguities.

#### Appendix D. Changes between revisions ...-21.txt and ...-24.txt

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

- o Alternate metrics enabled:
  - \* New section added to describe general design approach.
  - \* Abstract functions "Cost()" and "LoopFree()" defined.
  - \* MAX\_HOPCOUNT typically replaced by MAX\_METRIC.
  - \* DEFAULT\_METRIC\_TYPE parameter defined, defaulting to HopCount.
  - \* MetricType Message TLV defined.
  - \* Metric Address TLV defined.
- o Many changes for RFC 5444 compliance
- o New section added for "Notational Conventions" (see Table 1). Many changes to improve readability and accuracy (e.g., eliminate use of "Flooding", "ThisNode", ...).
- o Reorganized and simplified route lifetime management (see Section 5.1).
- o Reorganized document structure, combining closely related small sections and eliminating top-level "Detailed ..." section.
  - \* RREQ and RREP specification sections coalesced.
  - \* RERR specification sections coalesced.
  - \* Eliminated resulting duplicated specification.
  - \* New section added for "Notational Conventions".
- o Internet-Facing AODVv2 router renamed to be IAR
- o "Optional Features" section (see Section 13) created to contain features not required within base specification, including:
  - \* Adding RREP-ACK message type instead of relying on reception of arbitrary packets as sufficient response to establish bidirectionality.
  - \* Expanding Rings Multicast
  - \* Intermediate RREPs (iRREPs): Without iRREP, only the destination can respond to a RREQ.
  - \* Precursor lists.
  - \* Reporting Multiple Unreachable Nodes. An RERR message can carry more than one Unreachable Destination node for cases when a single link breakage causes multiple destinations to become unreachable from an intermediate router.
  - \* Message Aggregation.
  - \* Inclusion of Added Routing Information.
- o Sequence number MUST be incremented after generating any RteMsg.
- o Resulting simplifications for accepting route updates in RteMsgs.
- o Sequence number MUST (instead of SHOULD) be set to 1 after rollover.
- o AODVv2 routers MUST (instead of SHOULD) only handle AODVv2 messages from adjacent routers.
- o Clarification that Added Routing information in RteMsgs is optional (MAY) to use.
- o Clarification that if Added Routing information in RteMsgs is used, then the Route Table Entry SHOULD be updated using normal procedures as described in Section 6.2.

- o Clarification in Section 7.1 that nodes may be configured to buffer zero packets.
- o Clarification in Section 7.1 that buffered packets MUST be dropped if route discovery fails.
- o In Section 8.2, relax mandate for monitoring connectivity to next-hop AODVv2 neighbors (from MUST to SHOULD), in order to allow for minimal implementations
- o Remove Route.Forwarding flag; identical to "NOT" Route.Broken.
- o Routing Messages MUST be originated with the <msg-hop-limit> set to MAX\_HOPCOUNT.
- o Maximum hop count set to MAX\_HOPCOUNT, and 255 is reserved for "unknown". Since the current draft only uses hop-count as distance, this is also the current maximum distance.

#### Appendix E. Shifting Network Prefix Advertisement Between AODVv2 Routers

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

#### Authors' Addresses

Charles E. Perkins  
Futurewei Inc.  
2330 Central Expressway  
Santa Clara, CA 95050  
USA  
  
Phone: +1-408-330-5305  
Email: charliep@computer.org

Stan Ratliff  
Cisco  
170 West Tasman Drive  
San Jose, CA 95134  
USA

Email: [sratliff@cisco.com](mailto:sratliff@cisco.com)

John Dowdell  
Cassidian  
Celtic Springs  
Newport, Wales NP10 8FZ  
United Kingdom

Email: [John.Dowdell@Cassidian.com](mailto:John.Dowdell@Cassidian.com)



Mobile Ad hoc Networking (MANET)  
Internet-Draft  
Intended status: Informational  
Expires: December 19, 2013

J. Yi  
LIX, Ecole Polytechnique  
U. Herberg  
Fujitsu Laboratories of America  
T. Clausen  
LIX, Ecole Polytechnique  
June 17, 2013

Security Threats for NHDP  
draft-ietf-manet-nhdp-sec-threats-06

Abstract

This document analyzes common security threats of the Neighborhood Discovery Protocol (NHDP), and describes their potential impacts on MANET routing protocols using NHDP. This document is not intended to propose solutions to the threats described.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on December 19, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Terminology . . . . .	4
3. NHDP Threat Overview . . . . .	4
4. Detailed Threat Description . . . . .	5
4.1. Jamming . . . . .	5
4.2. Denial of Service Attack . . . . .	5
4.3. Eavesdropping and Traffic Analysis . . . . .	6
4.4. Incorrect HELLO Message Generation . . . . .	7
4.4.1. Identity Spoofing . . . . .	7
4.4.2. Link Spoofing . . . . .	8
4.5. Replay Attack . . . . .	9
4.6. Message Timing Attacks . . . . .	9
4.6.1. Interval Time Attack . . . . .	9
4.6.2. Validity Time Attack . . . . .	10
4.7. Indirect Channel Overloading . . . . .	10
4.8. Attack on Link Quality Update . . . . .	11
5. Impact of inconsistent Information Bases on Protocols using NHDP . . . . .	12
5.1. MPR Calculation . . . . .	12
5.1.1. Flooding Disruption due to Identity Spoofing . . . . .	12
5.1.2. Flooding Disruption due to Link Spoofing . . . . .	13
5.1.3. Broadcast Storm . . . . .	14
5.2. Routing Loops . . . . .	15
5.3. Invalid or Non-Existing Paths to Destinations . . . . .	15
5.4. Data Sinkhole . . . . .	16
6. Future Work . . . . .	16
7. Security Considerations . . . . .	17
8. IANA Considerations . . . . .	17
9. Acknowledgments . . . . .	17
10. References . . . . .	18
10.1. Normative References . . . . .	18
10.2. Informative References . . . . .	18
Authors' Addresses . . . . .	19



## 1. Introduction

The Neighborhood Discovery Protocol (NHDP) [RFC6130] allows routers to acquire topological information up to two hops away from themselves, by way of periodic HELLO message exchanges. The information acquired by NHDP is used by other protocols, such as OLSRv2 [I-D.ietf-manet-olsrv2] and SMF [RFC6621]. The topology information, acquired by way of NHDP, serves these routing protocols by detecting and maintaining local 1-hop and 2-hop neighborhood information.

As NHDP is typically used in wireless environments, it is potentially exposed to different kinds of security threats, some of which are of particular significance as compared to wired networks. As radio signals can be received as well as transmitted by any compatible wireless device within radio range, there is commonly no physical protection as otherwise known for wired networks. NHDP does not define any explicit security measures for protecting the integrity of the information it acquires, however suggests that the integrity protection be addressed in a fashion appropriate to the deployment of the network.

This document is based on the assumption that no additional security mechanism such as IPsec is used in the IP layer, as not all MANET deployments may be suitable to deploy common IP protection mechanisms (e.g., because of limited resources of MANET routers to support the IPsec stack). The document analyzes possible attacks on and mis-configurations of NHDP and outlines the consequences of such attacks/mis-configurations to the state maintained by NHDP in each router (and, thus, made available to protocols using this state).

This document is not intended to propose solutions to the threats described. [I-D.ietf-manet-nhdp-olsrv2-sec] provides further information on how to enable integrity protection to NHDP, which can help mitigating the threats described related to identity spoofing.

It should be noted that many NHDP implementations are configurable and so an attack on the configuration system (such as [RFC6779]) can be used to adversely affect the operation of an NHDP implementation.

The NHDP MIB module [RFC6779] might help monitoring some of the security attacks mentioned in this document. Note that, [I-D.nguyen-manet-management] contains specific guidelines on MANET network management, taking into account the specific nature of MANETs.

## 2. Terminology

This document uses the terminology and notation defined in [RFC5444], NHDP [RFC6130] and [RFC4949].

Additionally, this document introduces the following terminology:

**NHDP Router:** A MANET router, running NHDP as specified in [RFC6130].

**Attacker:** A device, present in the network and which intentionally seeks to compromise the information bases in NHDP routers.

**Compromised NHDP Router:** An attacker, present in the network and which generates syntactically correct NHDP control messages. Control messages emitted by a Compromised NHDP router may contain additional information, or omit information, as compared to a control message generated by a non-compromised NHDP router located in the same topological position in the network.

**Legitimate NHDP Router:** An NHDP router, which is not a Compromised NHDP Router.

## 3. NHDP Threat Overview

NHDP defines a HELLO messages exchange, enabling each NHDP Router to acquire topological information describing its 1-hop and 2-hop neighbors, and specifies information bases for recording this information.

An NHDP Router periodically transmits HELLO messages using a link-local multicast on each of its interfaces with a hop-limit of 1 (i.e., HELLOs are never forwarded). In these HELLO messages, an NHDP Router announces the IP addresses as heard, symmetric or lost neighbor interface addresses.

An Attacker has several ways of harming this neighbor discovery process: It can announce "wrong" information about its identity, postulate non-existent links, and replay HELLO messages. These attacks are presented in detail in Section 4.

The different ways of attacking an NHDP deployment may eventually lead to inconsistent information bases, not accurately reflecting the correct topology of the MANET. The consequence hereof is that protocols using NHDP will base their operation on incorrect information, causing routing protocols to not be able to calculate correct (or any) paths, degrade the performance of flooding operations based on reduced relay sets, etc. These consequences to

protocols using NHDP are described in detail in Section 5.

#### 4. Detailed Threat Description

For each threat, described in the below, a description of the mechanism of the corresponding attack is given, followed by a description of how the attack affects NHDP. The impacts from each attack on protocols using NHDP are given in Section 5.

For simplicity in the description, examples given assume that NHDP Routers have a single interface with a single IP address configured. All the attacks apply, however, for NHDP Routers with multiple interfaces and multiple addresses as well.

##### 4.1. Jamming

One vulnerability, common for all protocols operating a wireless ad hoc network, is that of "jamming", i.e., that a device generates massive amounts of interfering radio transmissions, which will prevent legitimate traffic (e.g., control traffic as well as data traffic) on part of a network. Jamming is a form of Interference and Overload with threat consequences of Disruption [RFC4593].

Depending on lower layers, this may not affect transmissions: HELLO messages from an NHDP Router with "jammed" interfaces may be received by other NHDP Routers. As NHDP identifies whether a link to a neighbor is uni-directional or bi-directional, a routing protocol that uses NHDP for neighborhood discovery may ignore a link from a jammed NHDP Router to a non-jammed NHDP Router. The jammed router (a router with jammed carrier) would appear simply as "disconnected" for the un-jammed part of the network - which is able to maintain accurate topology maps.

If, due to a jamming attack, a considerable amount of HELLO messages are lost or corrupted due to collisions, neighbor NHDP Routers are not able to establish links between themselves any more. Thus, NHDP will present empty information bases to the protocols using it.

##### 4.2. Denial of Service Attack

A Denial of Service (DoS) attack can be a result of misconfiguration of Legitimate NHDP Routers (e.g., very short HELLO transmission interval) or malicious behavior of Compromised NHDP Routers [ACCT2012], so called byzantine routers [RFC4593]. DoS is a form of Interference and Overload with threat consequences of Disruption [RFC4593].

By transmitting a huge amount of HELLO messages in a short period of time, NHDP Routers can increase channel occupation as introduced in Section 4.1. Furthermore, a Compromised NHDP Router can spoof a large amount of different IP addresses, and send HELLOs to its neighbors to fill their Link/Neighbor Sets. This may result in memory overflow, and makes the processing of legitimate HELLO messages impossible. A Compromised NHDP Router can also use link spoofing in its HELLO messages, generating huge 2-hop Sets in adjacent NHDP Routers and therefore potentially a memory overflow. Moreover, protocols such as SMF and OLSRv2, using the 2-hop information for MPR calculation, may exhaust the available computational resources of the router if the Neighbor Set and 2-hop Sets have too many entries.

By exhausting the memory, CPU, or (and) channel resources of a router in a DoS attack or a misconfiguration, NHDP Routers may not be able to accomplish their specified tasks of exchanging 1-hop and 2-hop neighborhood information, and thereby disturbing the operation of routing protocols using NHDP.

In some MANETs, the routers are powered by battery. Another consequence of DoS attack in such networks is that the power will be drained quickly by unnecessary message processing, transmission and receiving.

#### 4.3. Eavesdropping and Traffic Analysis

Eavesdropping, sometimes referred as sniffing, is a common and easy passive attack in a wireless environment. Once a packet is transmitted, any adjacent NHDP Router can potentially obtain a copy, for immediate or later processing. Neither the source nor the intended destination can detect this. A malicious NHDP Router can eavesdrop on the NHDP message exchange and thus learn the local topology. It may also eavesdrop on data traffic to learn source and destination addresses of data packets, or other header information, as well as the packet payload.

Eavesdropping does not pose a direct threat to the network nor to NHDP, in as much as that it does not alter the information recorded by NHDP in its information bases and presented to other protocols using it, but it can provide network information required for enabling other attacks, such as the identity of communicating NHDP Routers, detection of link characteristic, and NHDP Router configuration. The compromised NHDP Routers may use the obtained information to launch subsequent attacks, and they may also share NHDP routing information with other NHDP or non-NHDP entities. [RFC4593] would categorize the threat consequence as Disclosure.

Traffic analysis normally comes along with eavesdropping, which is the process of intercepting messages in order to deduce information from communication patterns. It can be performed even HELLO messages are encrypted (encryption is not a part of NHDP), for example:

- o Triggered HELLO messages: an attacker could figure out that messages are triggered and determine that there was a change of symmetric neighbors of an NHDP Router sending the HELLO (as well get the frequency).
- o Message size: the message grows exactly by x bytes per neighbor. Depending on which cipher is used for the encryption, some information about the size could be inferred and thus the number of neighbors guessed.

[RFC4593] would categorize the threat consequence as Disclosure.

#### 4.4. Incorrect HELLO Message Generation

An NHDP Router performs two distinct tasks: it periodically generates HELLO messages, and it processes incoming HELLO messages from neighbor NHDP Routers. This section describes security attacks involving the HELLO generation.

##### 4.4.1. Identity Spoofing

Identity spoofing implies that a Compromised NHDP Router sends HELLO messages, pretending to have the identity of another NHDP Router, or even a router that does not exist in the networks. A Compromised NHDP Router can accomplish this by using another IP address in an address block of a HELLO message, and associating this address with a LOCAL\_IF Address Block TLV [IJNSIA2010].

An NHDP Router receiving the HELLO message from a neighbor, will assume that it originated from the NHDP Router with the spoofed interface address. As a consequence, it will add a Link Tuple to that neighbor with the spoofed address, and include it in its next HELLO messages as a heard neighbor (and possibly as symmetric neighbor after another HELLO exchange).

Identity spoofing is particular harmful if a Compromised NHDP Router spoofs the identity of another NHDP Router that exists in the same routing domain. With respect to NHDP, such a duplicated, spoofed address can lead to an inconsistent state up to two hops from an NHDP Router. [RFC4593] would categorize the threat consequence as Disclosure and Deception.

Figure 1 depicts a simple example. In that example, NHDP Router A is

in radio range of C, but not of the Compromised NHDP Router X. If X spoofs the address of A, that can lead to conflicts for routing protocol that uses NHDP, and therefore for wrong path calculations as well as incorrect data traffic forwarding.

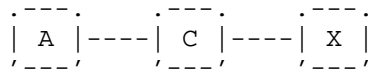


Figure 1

Figure 2 depicts another example. In this example, A is two hops away from NHDP Router C, reachable through NHDP Router B. If the Compromised NHDP Router X spoofs the address of A, D will take A as its one hop neighbor, and C may think that A is indeed reachable through NHDP Router D.

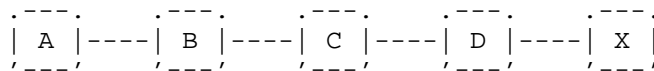


Figure 2

#### 4.4.2. Link Spoofing

Similar to identity spoofing, link spoofing implies that a Compromised NHDP Router sends HELLO messages, signaling an incorrect set of neighbors, sometimes referred to as Falsification [RFC4593]. This may take either of two forms:

- o A Compromised NHDP Router can postulate addresses of non-present neighbor NHDP Routers in an address block of a HELLO, associated with LINK\_STATUS TLVs.
- o A Compromised NHDP Router can "ignore" otherwise existing neighbors by not advertising them in its HELLO messages.

The effect of link spoofing with respect to NHDP are twofold, depending on the two cases mentioned above: If the Compromised NHDP Router ignores existing neighbors in its advertisements, links will be missing in the information bases maintained by other routers, and there may not be any connectivity to or from these NHDP Routers to others NHDP Routers in the MANET. If, on the other hand, the Compromised NHDP Router advertises non-existing links, this will lead to inclusion of topological information in the information base, describing non-existing links in the network (which, then, may be used by other protocols using NHDP in place of other, existing, links). [RFC4593] would categorize the threat consequence as

Usurpation, Deception and Disruption.

#### 4.5. Replay Attack

A replay attack implies that control traffic from one region of the network is recorded and replayed in a different region at (almost) the same time, or in the same region at a different time. This may, for example, happen when two Compromised NHDP Routers collaborate on an attack, one recording traffic in its proximity and tunneling it to the other Compromised NHDP Router, which replays the traffic. In a protocol where links are discovered by testing reception, this will result in extraneous link creation (basically, a "virtual" link between the two Compromised NHDP Routers will appear in the information bases of neighboring NHDP Routers). [RFC4593] would categorize this as a Falsification and Interference threat with a threat consequence of Usurpation, Deception, and Disruption.

While this situation may result from an attack, it may also be intentional: if data-traffic also is relayed over the "virtual" link, the link being detected is indeed valid for use. This is, for instance, used in wireless repeaters. If data traffic is not carried over the virtual link, an imaginary, useless, link between the two Compromised NHDP Routers, has been advertised, and is being recorded in the information bases of their neighboring NHDP Routers.

Compared to Incorrect HELLO Message attacks described in Section 4.4, the messages used in Replay attack are legitimate messages sent out by (non-malicious) NHDP Routers and replayed at a later time or different locality by malicious routers. This makes this kind of attack harder to be detect and to counteract: integrity checks cannot help in this case as the original message ICV (Integrity Check Values) was correctly calculated.

#### 4.6. Message Timing Attacks

In NHDP, each HELLO message contains a "validity time" and may contain an "interval time" field, identifying the time for which information in that control message should be considered valid until discarded, and the time until the next control message of the same type should be expected [RFC5497].

##### 4.6.1. Interval Time Attack

A use of the expected interval between two successive HELLO messages is for determining the link quality in NHDP: if messages are not received within the expected intervals (e.g., a certain fraction of messages are missing), then this may be used to exclude a link from being considered as useful, even if (some) bi-directional

communication has been verified. If a Compromised NHDP Router X spoofs the identity of an existing NHDP Router A, and sends HELLOs indicating a low interval time, an NHDP Router B receiving this HELLO will expect the following HELLO to arrive within the interval time indicated - or otherwise, decrease the link quality for the link A-B. Thus, X may cause NHDP Router B's estimate of the link quality for the link A-B to fall below the limit, where it is no longer considered as useful and, thus, not used [CPSCOM2011]. [RFC4593] would categorize the threat consequence as Usurpation.

#### 4.6.2. Validity Time Attack

A Compromised NHDP Router X can spoof the identity of an NHDP Router A and send a HELLO using a low validity time (e.g., 1 ms). A receiving NHDP Router B will discard the information upon expiration of that interval, i.e., a link between NHDP Router A and B will be "torn down" by X. It can be caused by intended malicious behaviors, or simply mis-configuration in the NHDP Routers. [RFC4593] would categorize the threat consequence as Usurpation.

#### 4.7. Indirect Channel Overloading

Indirect Channel Overloading is when a Compromised NHDP Router X by its actions causes other legitimate NHDP Routers to generate inordinate amounts of control traffic. This increases channel occupation, and the overhead in each receiving NHDP Router processing this control traffic. With this traffic originating from Legitimate NHDP Routers, the malicious device may remain undetected to the wider network. It is a form of Interference and Overload with threat consequences of Disruption [RFC4593].

Figure 3 illustrates Indirect Channel Overloading with NHDP. A Compromised NHDP Router X advertises a symmetric spoofed link to the non-existing NHDP Router B (at time t0). Router A selects X as MPR upon reception of the HELLO, and will trigger a HELLO at t1. Overhearing this triggered HELLO, the attacker sends another HELLO at t2, advertising the link to B as lost, which leads to NHDP Router A deselecting the attacker as MPR, and another triggered message at t3. The cycle may be repeated, alternating advertising the link X-B as LOST and SYM.



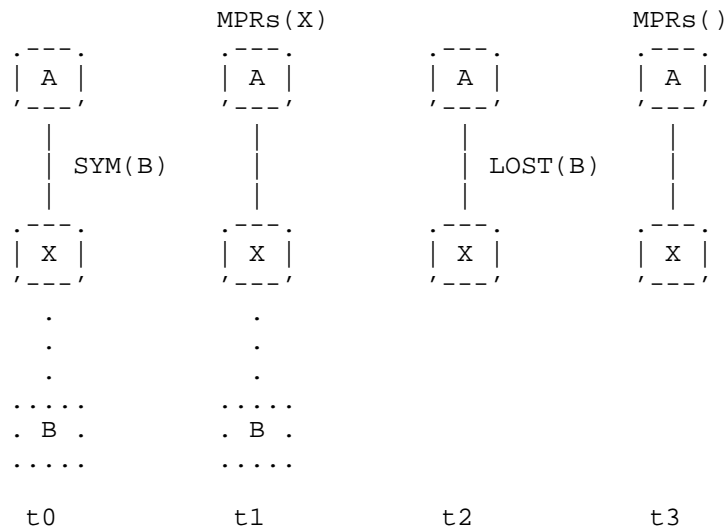


Figure 3

#### 4.8. Attack on Link Quality Update

According to NHDP, "Link quality is a mechanism whereby a router MAY take considerations other than message exchange into account for determining when a link is and is not a candidate for being considered as HEARD or SYMMETRIC. As such, it is a link admission mechanism."

Section 14.4 of NHDP [RFC6130] then lists several examples of which information can be used to update link quality. One of the listed examples is to update link quality based on [RFC5444] packet exchanges between neighbor routers, e.g., an NHDP Router may update the link quality of a neighbor based on receipt or loss of packets if they include a sequential packet sequence number.

NHDP does not specify how to acquire link quality updates normatively, however, attack vectors may be introduced if an implementation chooses to calculate link quality based on packet sequence numbers. The consequences of such threats would depend on specific implementations. For example, if the link quality update is based on sequential packet sequence number from neighbor routers, a Comprised NDHP Router can spoof packets appearing to be from another Legitimate NHDP Router that skips some packet sequence numbers. The NHDP Router receiving the spoofed packets may degrade the link quality as it appears that several packets have been dropped. Eventually, the router remove the neighbor when the link quality drops below `HYST_REJECT`.

## 5. Impact of inconsistent Information Bases on Protocols using NHDP

This section describes the impact on protocols, using NHDP, of NHDP failing to obtain and represent accurate information, possibly as a consequence of the attacks described in Section 4. This description emphasizes the impacts on the MANET protocols OLSRv2 [I-D.ietf-manet-olsrv2], and SMF [RFC6621].

### 5.1. MPR Calculation

MPR selection (as used in e.g., [I-D.ietf-manet-olsrv2] and [RFC6621]) uses information about a router's 1-hop and 2-hop neighborhood, assuming that (i) this information is accurate, and (ii) all 1-hop neighbors are apt to act as as MPR, depending on the willingness they report. Thus, a Compromised NHDP router may seek to manipulate the 1-hop and 2-hop neighborhood information in a router such as to cause the MPR selection to fail, leading to a flooding disruption of TC messages, which can result in incomplete topology advertisement, or degrade the optimized flooding to classical flooding.

#### 5.1.1. Flooding Disruption due to Identity Spoofing

A Compromised NHDP router can spoof the identify of other routers, to disrupt the MPR selection, so as to cache certain parts of the network from the flooding traffic [IJNSIA2010].

In Figure 4, a Compromised NHDP router X spoofs the identity of B. The link between X and C is correctly detected and listed in X's HELLOs. Router A will receive HELLOs indicating links from, respectively B:{B-E}, X:{X-C, X-E}, and D:{D-E, D-C}. For router A, X and D are equal candidates for MPR selection. To make sure the X can be selected as MPR for router A, X can set its willingness to the maximum value.

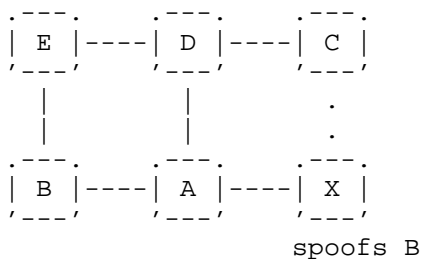


Figure 4

If B and X (i) accept MPR selection and (ii) forward flooded traffic

as-if they were both B, identity spoofing by X is harmless. However, if X does not forward flooded traffic (i.e., does not accept MPR selection), its presence entails flooding disruption: selecting B over D renders C unreachable by flooded traffic.

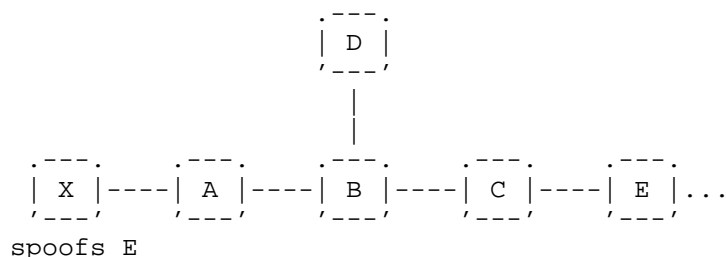


Figure 5

In Figure 5, the Compromised NHDP router X spoofs the identity of E, i.e., router A and C both receive HELLOs from a router identifying as E. For router B, A and C present the same neighbor sets, and are equal candidates for MPR selection. If router B selects only router A as MPR, C will not relay flooded traffic from or transiting via B, and router X (and routers to the "right" of it) will not receive flooded traffic.

#### 5.1.2. Flooding Disruption due to Link Spoofing

A Compromised NHDP router can also spoof links to other NHDP routers, and thereby makes itself appear as the most appealing candidate of MPR for its neighbors, possibly to the exclusion of other NHDP routers in the neighborhood (this, in particular, if the Compromised NHDP router spoof links to all other NHDP routers in the neighborhood, plus to one other NHDP router). By thus excluding other legitimate NHDP routers from being selected as MPR, the Compromised NHDP router will receive and be expected to relay all flooded traffic (e.g., TC messages in OLSRv2 or data traffic in SMF) - which it can then drop or otherwise manipulate.

In the network in Figure 6, the Compromised NHDP router X spoofs links to the existing router C, as well as to a fictitious W. Router A receives HELLOs from X and B, reporting X: {X-C, X-W}, b: {B-C}. All else being equal, X appears a better choice as MPR than B, as X appears to cover all neighbors of B, plus W.

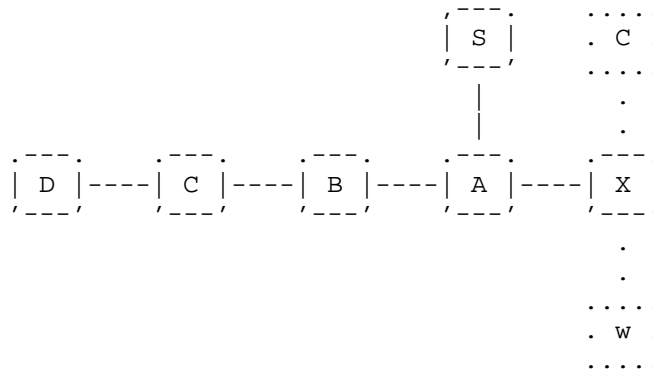


Figure 6

As router A will not select B as MPR, B will not relay flooded messages received from router A. The NHDP routers on the left of B (starting with C) will, thus, not receive any flooded messages from or transiting NHDP router A (e.g., a message originating from S).

#### 5.1.3. Broadcast Storm

Compromised NHDP router may attack the network by attempting to degrade the performance of optimized flooding algorithms so as to be equivalent to classic flooding. This can be achieved by forcing an NHDP router into choosing all its 1-hop neighbors as MPRs. In MANETs, a broadcast storm caused by classic flooding is a serious problem which can result in redundancy, contention and collisions [MOBICOM99].

As shown in Figure 7, the Compromised NHDP router X spoofs the identity of NHDP router B and, spoofs a link to router Y {B-Y} (Y does not have to exist). By doing so, the legitimate NHDP router A has to select the legitimate NHDP router B as its MPR, in order for it to reach all its 2-hop neighbors. The Compromised NHDP router Y can perform this identity+link spoofing for all of NHDP router A's 1-hop neighbors, thereby forcing NHDP router A to select all its neighbors as MPR - disabling the optimization sought by the MPR mechanism.

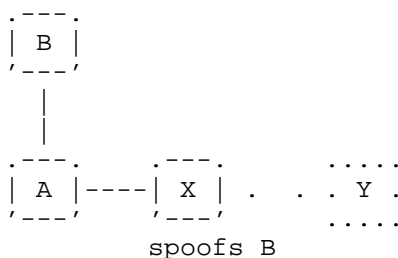


Figure 7

## 5.2. Routing Loops

Inconsistent information bases, provided by NHDP to other protocols, can also cause routing loops. In Figure 8, the Compromised NHDP router X spoofs the identity of NHDP router E. NHDP router D has data traffic to send to NHDP router A. The topology recorded in the information base of router D indicates that the shortest path to router A is {D->E->A}, because of the link {A-E} reported by X. Therefore, the data traffic will be routed to the NHDP router E. As the link {A-E} does not exist in NHDP router E's information bases, it will identify the next hop for data traffic to NHDP router A as being NHDP router D. A loop between the NHDP routers D and E is thus created.

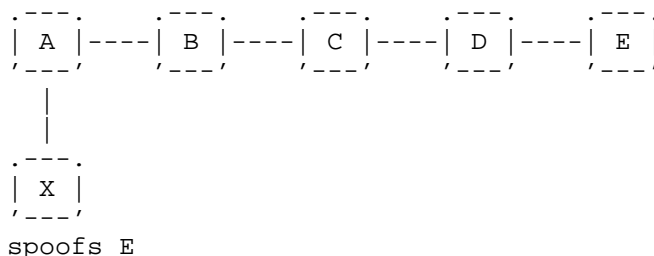


Figure 8

## 5.3. Invalid or Non-Existing Paths to Destinations

By reporting inconsistent topology information in NHDP, the invalid links/routers can be propagated as link state information with TC messages and results in route failure. As illustrated in Figure 8, if NHDP router B tries to send data packets to NHDP router E, it will choose router A as its next hop, based on the information of non-existing link {A-E} reported by the Compromised NHDP router X.

#### 5.4. Data Sinkhole

With the ability to spoof multiple identities of legitimate NHDP routers (by eavesdropping, for example), the Compromised NHDP router can represent a "data sinkhole" for its 1-hop and 2-hop neighbors. Data packets that come across its neighbors may be forwarded to the Compromised NHDP router instead of to the real destination. The packet can then be dropped, manipulated, duplicated, etc., by the Compromised NHDP router. As shown in Figure 8, if the Compromised NHDP router X spoofs the identity of NHDP router E, all the data packets to E that cross NHDP routers A and B will be sent to NHDP router X, instead of to E.

#### 6. Future Work

This document does not propose solutions to mitigate the security threats described in Section 4. However, this section aims at driving new work by suggesting which threats discussed in Section 4 could be addressed in new protocol work, which in deployment, and which by applications:

- o Section 4.1: Jamming - If a single router or a small area of the MANET is jammed, protocols could be specified that increase link metrics in NHDP for the jammed links. When a routing protocol, such as OLSRv2, uses NHDP for neighborhood discovery, other paths leading "around" the jammed area would be preferred, and therefore mitigate the threat to some extent.
- o Section 4.2: DoS - DoS using a massive amount of HELLO messages can be mitigated by admitting only trusted routers to the network. [I-D.ietf-manet-nhdp-olsrv2-sec] specifies a mechanism for adding Integrity Check Values (ICVs) to HELLO messages and therefore providing an admittance mechanism for NHDP Routers to a MANET. (Note that adding ICVs adds itself a new DoS attack vector, as ICV verification requires CPU and memory resources.) Using ICVs does however not address the problem of compromised routers. Detecting compromised routers could be addressed in new work. [I-D.ietf-manet-nhdp-olsrv2-sec] mandates to implement a security mechanism based on shared keys, which makes excluding single compromised routers difficult; work could be done to facilitate revocation mechanisms in certain MANET use cases where routers have sufficient capabilities to support asymmetric keys.
- o Section 4.3: Eavesdropping - [I-D.ietf-manet-nhdp-olsrv2-sec] adds ICVs to HELLO messages, but does not encrypt them. Therefore, eavesdropping of control traffic is not mitigated. Future work could provide encryption of control traffic for sensitive MANET

topologies. Note that, other than using a single shared secret key, encryption to a potentially a priori unknown set of neighbors, especially without multiplying overheads, is non-trivial. By traffic analyzing, attackers could still deduce the network information like HELLO message triggering, and HELLO message size, even HELLO messages are encrypted.

- o Section 4.4.2: Link spoofing - [I-D.ietf-manet-nhdp-olsrv2-sec] provides certain protection against link spoofing, but an NHDP Router has to "trust" the originator of a HELLO that the advertized links are correct. For example, if a router A reports a link to B, routers receiving HELLOs from A have to trust that B is actually a (symmetric) neighbor of A. New protocol work could address protection of links without overly increasing space and time overheads. An immediate suggestion for deployments is to protect routers against being compromised and distributing keys only to trusted routers.
- o Section 4.5: Replay Attacks - [I-D.ietf-manet-nhdp-olsrv2-sec] provides certain protection against replay attacks, using ICVs and timestamps. It is still feasible to replay control messages within limited time. A suggestion for deployments is to provide time synchronization between routers. New work could provide time synchronization mechanisms for certain MANET use cases, or specify a mechanism using nonces instead of time stamps in HELLO messages.
- o Section 4.4.1: Identity spoofing, Section 4.6: Message timing attacks, Section 4.7: Indirect channel overloading, and Section 4.8: Attack on link quality update - [I-D.ietf-manet-nhdp-olsrv2-sec] provides protection against these attacks, assuming that routers are not compromised.

## 7. Security Considerations

This document does not specify a protocol or a procedure. The document, however, reflects on security considerations for NHDP and MANET routing protocols using NHDP for neighborhood discovery.

## 8. IANA Considerations

This document contains no actions for IANA.

## 9. Acknowledgments

The authors would like to gratefully acknowledge the following people

for valuable comments and technical discussions: Teco Boot, Henning Rogge, Christopher Dearlove, John Dowdell, Joseph Macker, and the all the other participants of IETF MANET working group.

## 10. References

### 10.1. Normative References

- [RFC5444] Clausen, T., Dearlove, C., Dean, J., and C. Adjih, "Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format", RFC 5444, February 2009.
- [RFC5497] Clausen, T. and C. Dearlove, "Representing Multi-Value Time in Mobile Ad Hoc Networks (MANETs)", RFC 5497, March 2009.
- [RFC6130] Clausen, T., Dearlove, C., and J. Dean, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)", RFC 6130, April 2011.

### 10.2. Informative References

- [ACCT2012] Jhaveri, R. and S. Patel, "DoS Attacks in Mobile Ad Hoc Networks: A Survey", Second International Conference on Advanced Computing & Communication Technologies (ACCT), Jan 2012.
- [CPSCOM2011] Yi, J., Clausen, T., and U. Herberg, "Vulnerability Analysis of the Simple Multicast Forwarding (SMF) Protocol for Mobile Ad Hoc Networks", Proceedings of the IEEE International Conference on Cyber, Physical, and Social Computing (CPSCom), October 2011.
- [I-D.ietf-manet-nhdp-olsrv2-sec] Herberg, U., Dearlove, C., and T. Clausen, "Integrity Protection for Control Messages in NHDP and OLSRv2", draft-ietf-manet-nhdp-olsrv2-sec-02 (work in progress), April 2013.
- [I-D.ietf-manet-olsrv2] Clausen, T., Dearlove, C., Jacquet, P., and U. Herberg, "The Optimized Link State Routing Protocol version 2", draft-ietf-manet-olsrv2-19 (work in progress), March 2013.
- [I-D.nguyen-manet-management]



Nguyen, J., Cole, R., Herberg, U., Yi, J., and J. Dean,  
"Network Management of Mobile Ad hoc Networks (MANET):  
Architecture, Use Cases, and Applicability",  
draft-nguyen-manet-management-00 (work in progress),  
February 2013.

[IJNSIA2010]

Herberg, U. and T. Clausen, "Security Issues in the  
Optimized Link State Routing Protocol version 2",  
International Journal of Network Security & Its  
Applications, April 2010.

[MOBICOM99]

Ni, S., Tseng, Y., Chen, Y., and J. Sheu, "The Broadcast  
Storm Problem in a Mobile Ad Hoc Network", Proceedings of  
the 5th annual ACM/IEEE international conference on Mobile  
computing and networking, 1999.

[RFC4593] Barbir, A., Murphy, S., and Y. Yang, "Generic Threats to  
Routing Protocols", RFC 4593, October 2006.

[RFC4949] Shirey, R., "Internet Security Glossary, Version 2",  
RFC 4949, August 2007.

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

[RFC6779] Herberg, U., Cole, R., and I. Chakeres, "Definition of  
Managed Objects for the Neighborhood Discovery Protocol",  
RFC 6779, October 2012.

Authors' Addresses

Jiazi Yi  
LIX, Ecole Polytechnique  
91128 Palaiseau Cedex,  
France  
  
Phone: +33 1 77 57 80 85  
Email: [jiazi@jiaziyi.com](mailto:jiazi@jiaziyi.com)  
URI: <http://www.jiaziyi.com/>

Ulrich Herberg  
Fujitsu Laboratories of America  
1240 E Arques Ave  
Sunnyvale, CA 94085  
USA

Email: [ulrich@herberg.name](mailto:ulrich@herberg.name)  
URI: <http://www.herberg.name/>

Thomas Heide Clausen  
LIX, Ecole Polytechnique  
91128 Palaiseau Cedex,  
France

Phone: +33 6 6058 9349  
Email: [T.Clausen@computer.org](mailto:T.Clausen@computer.org)  
URI: <http://www.thomasclausen.org/>



Mobile Ad hoc Networking (MANET)  
Internet-Draft  
Intended status: Informational  
Expires: November 2, 2013

C. Dearlove  
BAE Systems ATC  
T. Clausen  
LIX, Ecole Polytechnique  
P. Jacquet  
Alcatel-Lucent Bell Labs  
May 1, 2013

Link Metrics for the Mobile Ad Hoc Network (MANET) Routing Protocol  
OLSRv2 - Rationale  
draft-ietf-manet-olsrv2-metrics-rationale-04

Abstract

OLSRv2 includes the ability to assign metrics to links and to use those metrics to allow routing by other than minimum hop count routes. This document provides a historic record of the rationale for, and design considerations behind, how link metrics were included in OLSRv2.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on November 2, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

## Table of Contents

1. Introduction . . . . .	3
2. Terminology . . . . .	6
3. Applicability . . . . .	7
4. Motivational Scenarios . . . . .	8
5. Link Metrics . . . . .	10
5.1. Link Metric Properties . . . . .	10
5.2. Link Metric Types . . . . .	11
5.3. Directional Link Metrics . . . . .	12
5.4. Reporting Link and Neighbor Metrics . . . . .	13
5.5. Defining Incoming Link Metrics . . . . .	14
5.6. Link Metric Values . . . . .	15
6. MPRs with Link Metrics . . . . .	17
6.1. Flooding MPRs . . . . .	17
6.2. Routing MPRs . . . . .	19
6.3. Relationship Between MPR Sets . . . . .	22
7. IANA Considerations . . . . .	24
8. Security Considerations . . . . .	25
9. Acknowledgements . . . . .	26
10. Informative References . . . . .	27
Appendix A. MPR Routing Property . . . . .	28

## 1. Introduction

The Optimized Link State Routing Protocol version 1 (OLSRv1) [RFC3626] is a proactive routing protocol for Mobile Ad hoc NETWORKS (MANETs) [RFC2501]. OLSRv1 finds shortest, defined as minimum number of hops, routes from a router to all possible destinations.

Using only minimum hop routes may result in what are, in practice, inferior routes. Some examples are given in Section 4. Thus, one of the distinguishing features of the Optimized Link State Routing Protocol version 2 (OLSRv2) [OLSRv2] is the introduction of the ability to select routes using link metrics other than the number of hops.

During the development of OLSRv2 the working group and authors repeatedly discussed how and why some choices were made in the protocol specification, particularly at the metric integration level. Some of the issues may be non-intuitive and this document is presented as a record of the considerations and decisions to provide informational discussion about motivation and historic design choices. This document is intended to be useful as a reference if those questions arise again.

OLSRv2 essentially first determines local link metrics from 1-hop neighbors, these being defined by a process outside OLSRv2, then distributes required link metric values in HELLO messages and TC messages, and then finally forms routes with minimum total link metric. Using a definition of route metric other than number of hops is a natural extension that is commonly used in link state protocols.

Use of the extensible message format [RFC5444] by OLSRv2 has allowed the addition, by OLSRv2, of link metric information to the HELLO messages defined in the MANET NeighborHood Discovery Protocol (NHDP) [RFC6130] as well as inclusion in the Topology Control (TC) messages defined in [OLSRv2].

A metric-based route selection processes for OLSRv2 could have been handled as an extension to OLSRv2. However were this to have been done, OLSRv2 routers that did not implement this extension would not recognize any link metric information, and would attempt to use minimum hop-count routes. This would have meant that, in effect, routers that did and did not implement this extension would differ over their valuation of links and routes. This would have led to the fundamental routing problem of "looping". Thus if metric-based route selection were to have been considered only as an extension to OLSRv2, then routers that did, and routers that did not, implement this extension would not have been able to interoperate. This would have been a significant limitation of such an extension. Link

metrics were therefore included as standard in OLSRv2.

This document discusses the motivation and design rationale behind how link metrics were included in OLSRv2. The principal issues involved when including link metrics in OLSRv2 were:

- o Assigning metrics to links involved considering separate metrics for the two directions of a link, with the receiving router determining the metric from transmitter to receiver. A metric used by OLSRv2 may be either of:
  - \* A link metric, the metric of a specific link from an OLSRv2 interface of the transmitting router to an OLSRv2 interface of the receiving router.
  - \* A neighbor metric, the minimum of the link metrics between two OLSRv2 routers, in the indicated direction.

These metrics are necessarily the same when these routers each have a single OLSRv2 interface, but may differ when either has more. HELLO messages may include both link metrics and neighbor metrics. TC messages include only neighbor metrics.

- o Metrics as used in OLSRv2 are defined to be dimensionless and additive. The assignment of metrics, including their relationship to real parameters such as data rate, loss rate and delay, and the management of the choice of metric, is outside the scope of [OLSRv2], which simply uses these metrics in a consistent manner. Within a single MANET, including all components of a temporarily fragmented MANET, a single choice of link metric is used. By use of a registry of metric types (employing extended types of a single Address Block TLV type), routers can be configured to use only a subset of the available metric types.
- o Node metrics were not included in OLSRv2. Node metrics can be implemented by the addition of the corresponding value to all incoming link metrics by the corresponding router.
- o The separation of the two functions performed by MultiPoint Relays (MPRs) in OLSRv1, optimized flooding and reduced topology advertisement for routing, into separate sets of MPRs in OLSRv2 [OLSRv2], denoted "flooding MPRs" and "routing MPRs". Flooding MPRs can be calculated as in [RFC3626], but the use of link metrics in OLSRv2 can improve the MPR selection. Routing MPRs need a metric-aware selection algorithm. The selection of routing MPRs guarantees the use of minimum distance routes using the chosen metric, while using only symmetric 2-hop neighborhood information from HELLO messages and routing MPR selector

information from TC messages.

- o The protocol Information Bases defined in OLSRv2 include required metric values. This has included additions to the protocol Information Bases defined in NHDP [RFC6130] when used by OLSRv2.



## 2. Terminology

All terms introduced in [RFC5444], including "message" and "TLV" (type-length-value), are to be interpreted as described there.

All terms introduced in [RFC6130], including "MANET interface", "HELLO message", "heard", "link", "symmetric link", "1-hop neighbor", "symmetric 1-hop neighbor", "2-hop neighbor", "symmetric 2-hop neighbor", "symmetric 2-hop neighborhood", and the symbolic constants SYMMETRIC and HEARD, are to be interpreted as described there.

All terms introduced in [OLSRv2], including "router", "OLSRv2 interface", "willingness", "MultiPoint Relay (MPR)", "MPR selector", "MPR flooding" and the TLV type LINK\_METRIC are to be interpreted as described there.

### 3. Applicability

The objective of this document is to retain the design considerations behind how link metrics were included in [OLSRv2]. This document does not prescribe any behavior, but explains some aspects of the operation of OLSRv2.

#### 4. Motivational Scenarios

The basic situation that suggests the desirability of use of routes other than minimum hop routes is shown in Figure 1.

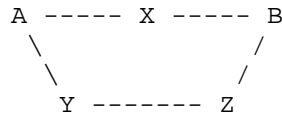


Figure 1

The minimum hop route from A to B is via X. However if the links A to X and X to B are poor (e.g., having low data rate or being unreliable) but the links A to Y, Y to Z and Z to B are better (e.g., having reliable high data rate) then the route A to B via Y and Z may be preferred to that via X.

There are other situations where, even if the avoidance of some links does not show immediately obvious benefits to users, their use should be discouraged. Consider a network with many short range links, and a few long range links. Use of minimum hop routes will immediately lead to heavy use of the long range links. This will be particularly undesirable if those links achieve their longer range through reduced data rate, or through being less reliable. However, even if the long range links have the same characteristics as the short range links, it may be better to reserve usage of the long range links for when this usage is particularly valuable - for example when the use of one long range link saves several short range links, rather than the single link saving that is all that is needed for a minimum hop route.

A related case is that of a privileged relay. An example is an aerial router in an otherwise ground based network. The aerial router may have a link to many, or even all, other routers. That would lead to all routers attempting to send all their traffic (other than to symmetric 1-hop neighbors and some symmetric 2-hop neighbors) via the aerial router. It may however be important to reserve that capacity for cases where the aerial router is actually essential, such as if the ground based portion of the network is not connected.

Link metrics provide a possible solution to these scenarios. For example, in Figure 1 the route A to Y to Z to B could be preferred to A to X to B by making the metrics on the former path 1 and those on the latter path 2. The aerial privileged relay could be used only when necessary by giving its links maximal metric values, with much smaller other metric values, or if the aerial link is to be preferred to N ground links, giving the ground links metric values of 1, while

making the sum of the aerial node uplink and downlink metrics equal to N.

Other cases may involve attempts to avoid areas of congestion, to route around insecure routers (by preference, but prepared to use them if there is no other alternative) and routers attempting to discourage their use as relays due to, for example, limited battery power. OLSRv2 does have another mechanism to aid in this, a router's willingness to act as an MPR. However there are cases where that cannot help, but where use of non-minimum hop routes could.

Similarly, note that OLSRv2's optional use of link quality (through its use of [RFC6130]) is not a solution to these problems. Use of link quality as specified in [RFC6130] allows a router to decline to use a link, not only on its own, but on all routers' behalf. It does not, for example, allow the use of a link otherwise determined to be too low quality to be generally useful, as part of a route where no better links exist. These mechanisms (link quality and link metrics) solve distinctly different problems.

It should also be noted that the loop-free property of OLSRv2 applies strictly only in the static state. When the network topology is changing, and when messages can be lost, it is possible for transient loops to form. However with update rates appropriate to the rate of topology change, such loops will be sufficiently rare. Changing link metrics is a form of network topology change, and should be limited to a rate slower than the message information update rate (defined by the parameters HELLO\_INTERVAL, HELLO\_MIN\_INTERVAL, REFRESH\_INTERVAL, TC\_INTERVAL and TC\_MIN\_INTERVAL).

## 5. Link Metrics

This section describes the required and selected properties of the link metrics used in OLSRv2, followed by implementation details achieving those properties.

### 5.1. Link Metric Properties

Link metrics in OLSRv2 are:

- o Dimensionless. While they may, directly or indirectly, correspond to specific physical information (such as delay, loss rate or data rate), this knowledge is not used by OLSRv2. Instead, generating the metric value is the responsibility of a mechanism external to OLSRv2.
- o Additive, so that the metric of a route is the sum of the metrics of the links forming that route. Note that this requires a metric where a low value of a link metric indicates a "good" link and a high value of a link metric indicates a "bad" link, and the former will be preferred to the latter.
- o Directional, the metric from router A to router B need not be the same as the metric from router B to router A, even when using the same OLSRv2 interfaces. At router A, a link metric from router B to router A is referred to as an incoming link metric, while a link metric from router A to router B is referred to as an outgoing link metric. (These are, of course, reversed at router B.)
- o Specific to a pair of OLSRv2 interfaces, so that if there is more than one link from router A to router B, each has its own link metric in that direction. There is also an overall metric, a "neighbor metric", from router A to router B (its 1-hop neighbor). This is the minimum value of the link metrics from router A to router B, considering symmetric links only; it is undefined if there are no such symmetric links. A neighbor metric from one router to another is always equal to a link metric in the same direction between OLSRv2 interfaces of those routers. When referring to a specific OLSRv2 interface (for example in a Link Tuple or a HELLO message sent on that OLSRv2 interface) a link metric always refers to a link on that OLSRv2 interface, to or from the indicated 1-hop neighbor OLSRv2 interface, while a neighbor metric may be equal to a link metric to and/or from another OLSRv2 interface.

## 5.2. Link Metric Types

There are various physical characteristics that may be used to define a link metric. Some examples, which also illustrate some characteristics of metrics that result, are:

- o Delay is a straightforward metric, as it is naturally additive, the delay of a multi-link route is the sum of the delays of the links. This does not directly take into account delays due to routers (such as due to router queues or transiting packets between router interfaces), rather than links, but these delays can be divided among incoming and outgoing links.
- o Probability of loss on a link is, as long as probabilities of loss are small and independent, approximately additive. (A slightly more accurate approach is using a negatively scaled logarithm of the probability of not losing a packet.) If losses are not independent then this will be pessimistic.
- o Data rates are not additive, it even has the wrong characteristic of being good when high, bad when low; thus a mapping that inverts its ordering must be applied. Such a mapping can, at best, only produce a metric that it is acceptable to treat as additive. Consider, for example, a preference for a route that maximizes the minimum data rate link on the route, and then prefers a route with the fewest links of each data rate from the lowest. If links may be of three discrete data rates, "high", "medium", and "low", then this preference can be achieved, on the assumption that no route will have more than 10 links, with metric values of 1, 10 and 100 for the three data rates.

If routes can have more than 10 links, the range of metrics must be increased; this was one reason for a preference for a wide "dynamic range" of link metric values. Depending on the ratios of the numerical values of the three data rates, the same effect may be achieved by using a scaling of an inverse power of the numerical values of the data rates. For example if the three data rates were 2, 5 and 10 Mbit/s, then a possible mapping would be the fourth power of 10 Mbit/s divided by the data rate, giving metric values of 625, 16 and 1 (good for up to 16 links in a route). This mapping can be extended to a system with more data rate values, for example giving a 4 Mbit/s data rate a metric value of about 39. This may lose the capability to produce an absolutely maximum minimum data rate route, but will usually produce either that, or something close (and at times maybe better, is a route of three 5 Mbit/s links really better than one of a single 4 Mbit/s link?). Specific metrics will need to define the mapping.

There are also many other possible metrics, including using physical layer information (such as signal to noise ratio, and error control statistics) and information such as packet queuing statistics.

In a well-designed network, all routers will use the same metric type. It will not produce good routes if, for example, some link metrics are based on data rate and some on path loss (except to the extent that these may be correlated). How to achieve this is an administrative matter, outside the scope of OLSRv2. In fact even the actual physical meanings of the metrics is outside the scope of OLSRv2. This is because new metrics may be added in the future, for example as data rates increase, and may be based on new, possibly non-physical, considerations, for example financial cost. Each such type will have a metric type number. Initially a single link metric type zero is defined as indicating a dimensionless metric with no predefined physical meaning.

An OLSRv2 router is instructed which single link metric type to use and recognize, without knowing whether it represents delay, probability of loss, data rate, cost or any other quantity. This recognized link metric type number is a router parameter, and subject to change in case of reconfiguration, or possibly the use of a protocol (outside the scope of OLSRv2) permitting a process of link metric type agreement between routers.

The use of link metric type numbers also suggests the possibility of use of multiple link metric types and multiple network topologies. This is a possible future extension to OLSRv2. To allow for that future possibility, the sending of more than one metric, of different physical types, which should otherwise not be done for reasons of efficiency, is not prohibited, but types other than that configured will be ignored.

The following three sections assume a chosen single link metric type, of unspecified physical nature.

### 5.3. Directional Link Metrics

OLSRv2 uses only "symmetric" (bidirectional) links, which may carry traffic in either direction. A key decision was whether these links should each be assigned a single metric, used in both directions, or a metric in each direction, noting that:

- o Links can have different characteristics in each direction, use of directional link metrics recognizes this.
- o In many (possibly most) cases, the two ends of a link will naturally form different views as to what the link metric should

be. To use a single link metric requires a coordination between the two that can be avoided if using directional metrics. Note that if using a single metric, it would be essential that the two ends agree as to its value, otherwise it is possible for looping to occur. This problem does not occur for directional metrics.

Based on these considerations, directional metrics are used in OLSRv2. Each router must thus be responsible for defining the metric in one direction only. This could have been in either direction, i.e., that a router is responsible for either incoming or outgoing link metrics, as long as the choice is universal. The former (incoming) case is used in OLSRv2 because, in general, receiving routers have more information available to determine link metrics (for example received signal strength, interference levels, and error control coding statistics).

Note that, using directional metrics, if router A defines the metric of the link from router B to router A, then router B must use router A's definition of that metric on that link in that direction. (Router B could, if appropriate, use a bad mismatch between directional metrics as a reason to discontinue use of this link, using the link quality mechanism defined in [RFC6130]; note that this is a distinct mechanism from the use of link metrics.)

#### 5.4. Reporting Link and Neighbor Metrics

Links, and hence link metrics, are reported in HELLO messages. A router must report incoming link metrics in its HELLO messages in order that these are each available at the other end of the link. This means that, for a symmetric link, both ends of the link will know both of the incoming and outgoing link metrics.

As well as advertising incoming link metrics, HELLO messages also advertise incoming neighbor metrics. These are used for routing MPR selection (see Section 6.2), which requires use of the lowest metric link between two routers when more than one link exists. This neighbor metric may be using another OLSRv2 interface, and hence the link metric alone is insufficient.

Metrics are also reported in TC messages. It can be shown that these need to be outgoing metrics:

- o Router A must be responsible for advertising a metric from router A to router B in TC messages. This can be seen by considering a route connecting single OLSRv2 interface routers P to Q to R to S. Router P receives its only information about the link from R to S in the TC messages transmitted by router R, which is an MPR of router S (assuming that only MPR selectors are reported in TC



messages). Router S may not even transmit TC messages (if no routers have selected it as an MPR and it has no attached networks to report). So any information about the metric of the link from R to S must also be included in the TC messages sent by router R, hence router R is responsible for reporting the metric for the link from R to S.

- o In a more general case, where there may be more than one link from R to S, the TC message must, in order that minimum metric routes can be constructed (e.g., by router P) report the minimum of these outgoing link metrics, i.e., the outgoing neighbor metric from R to S.

In this example, router P also receives information about the existence of a link between Q and R in the HELLO messages sent by router Q. Without the use of metrics, this link could be used by OLSRv2 for two hop routing to router R, using just HELLO messages sent by router Q. For this property (which accelerates local route formation) to be retained (from OLSRv1) router P must receive the metric from Q to R in HELLO messages sent by router Q. This indicates that router Q must be responsible for reporting the metric for the outgoing link from Q to R. This is in addition to the incoming link metric information that a HELLO message must report. Again, in general, this must be the outgoing neighbor metric, rather than the outgoing link metric.

In addition, Section 6.1 offers an additional reason for reporting outgoing neighbor metrics in HELLO messages, without which metrics can properly affect only routing, not flooding.

Note that there is no need to report an outgoing link metric in a HELLO message. The corresponding 1-hop neighbor knows that value, it specified it. Furthermore, for 2-hop neighborhood use, neighbor metrics are required (as these will, in general, not use the same OLSRv2 interface).

#### 5.5. Defining Incoming Link Metrics

When a router reports a 1-hop neighbor in a HELLO message, it may do so for the first time with link status HEARD. As the router is responsible for defining and reporting incoming link metrics, it must evaluate that metric, and attach that link metric to the appropriate address (which will have link status HEARD) in the next HELLO message reporting that address on that OLSRv2 interface. There will, at this time, be no outgoing link metric available to report, but a router must be able to immediately decide on an incoming link metric once it has heard a 1-hop neighbor on an OLSRv2 interface for the first time.

This is because, when receiving a HELLO message from this router, the 1-hop neighbor seeing its own address listed with link status HEARD will (unless the separate link quality mechanism indicates otherwise) immediately consider that link to be SYMMETRIC, advertise it with that link status in future HELLO messages, and use it (for MPR selection and data traffic forwarding).

It may, depending on the physical nature of the link metric, be too early for an ideal decision as to that metric, however a choice must be made. The metric value may later be refined based on further observation of HELLO messages, other message transmissions between the routers, or other observations of the environment. It will probably be best to over-estimate the metric if initially uncertain as to its value, to discourage, rather than over-encourage, its use. If no information other than the receipt of the HELLO message is available, then a conservative maximum link metric value, denoted `MAXIMUM_METRIC` in [OLSRv2], should be used.

#### 5.6. Link Metric Values

Link metric values are recorded in `LINK_METRIC` TLVs, defined in [OLSRv2], using a compressed (lossy) representation that occupies 12 bits. The use of 12 bits is convenient because, when combined with 4 flag bits of additional information, described below, this results in a 2 octet value field. However the use of 12 bits, and thus the availability of 4 flag bits, was a consequence of a design to use a modified exponent/mantissa form with the following characteristics:

- o The values represented are to be positive integers starting 1, 2, ...
- o The maximum value represented should be close to, but less than  $2^{24}$  (^ denotes exponentiation in this section). This is so that with a route limited to no more than 255 hops, the maximum route metric is less than  $2^{32}$ , i.e., can be stored in 32 bits. (The link metric value can be stored in 24 bits.)

A representation, modified from an exponent/mantissa form with  $e$  bits of exponent and  $m$  bits of mantissa, and which has the first of these properties is one that starts at 1, then is incremented by 1 up to  $2^m$ , then has a further  $2^m$  increments by 2, then a further  $2^m$  increments by 4, and so on for  $2^e$  sets of increments. This means that the represented value is never in error by more than a half (if rounding) or one (if truncating) part in  $2^m$ , usually less.

The position in the increment sequence, from 0 to  $2^m-1$ , is considered as a form of mantissa, and denoted  $a$ . The increment sequence number, from 0 to  $2^e-1$ , is considered as a form of

exponent, and denoted  $b$ .

The value represented by  $(b,a)$  can then be shown to be equal to  $(2^{m+a+1})2^b-2^m$ . To verify this, note that:

- o With fixed  $b$ , the difference between two values with consecutive values of  $a$  is  $2^b$ , as expected.
- o The value represented by  $(b,2^m-1)$  is  $(2^{m+2^m})2^b-2^m$ . The value represented by  $(b+1,0)$  is  $(2^{m+1})(2^{b+1})-2^m$ . The difference between these two values is  $2^{b+1}$ , as expected.

The maximum represented value has  $b = 2^e-1$  and  $a = 2^m-1$ , and is  $(2^{m+2^m})(2^{(2^e-1)})-2^m = 2^{(2^e+m)}-2^m$ . This is slightly less than  $2^{(2^e+m)}$ . The required 24 bit limit can be achieved if  $2^{e+m} = 24$ . Of the possible  $(e,m)$  pairs that satisfy this equation, the pair  $e = 4$ ,  $m = 8$  was selected as most appropriate, and is that used by OLSRv2. It uses the previously indicated  $e+m = 12$  bits. An algorithm for converting from a 24 bit value  $v$  to a 12 bit pair  $(b,a)$  is given in Section 6.2 of [OLSRv2].

As noted above, the 12 bit representation then shares two octets with 4 flag bits. Putting the flag bits first, it is then natural to put the exponent bits in the last four bits of the first octet, and to put the mantissa bits in the second octet. The 12 consecutive bits, using network byte order (most significant octet first), then represent  $256b+a$ . Note that the ordering of these 12 bit representation values is the same as the ordering of the 24 bit metric values. In other words, two 12 bit metrics fields can be compared for equality/ordering as if they were unsigned integers.

The four flag bits each represent one kind of metric, defined by its direction (incoming or outgoing) and whether the metric is a link metric or a neighbor metric. As indicated by the flag bits set, a metric value may be of any combination of these four kinds of metric.

## 6. MPRs with Link Metrics

MPRs are used for two purposes in OLSRv2. In both cases it is MPR selectors that are actually used, MPR selectors being determined from MPRs advertised in HELLO messages.

- o Optimized Flooding. This uses the MPR selector status of symmetric 1-hop neighbor routers from which messages are received in order to determine if these messages are to be forwarded. MPR selector status is recorded in the Neighbor Set (defined in [RFC6130] and extended in [OLSRv2]), and determined from received HELLO messages.
- o Routing. Non-local link information is based on information recorded in this router's Topology Information Base. That information is based on received TC messages. The neighbor information in these TC messages consists of addresses of the originating router's advertised (1-hop) neighbors, as recorded in that router's Neighbor Set (defined in [RFC6130] and extended in [OLSRv2]). These advertised neighbors include all of the MPR selectors of the originating router.

Metrics interact with these two uses of MPRs differently, as described in the following two sections, and which leads to the requirement for two separate sets of MPRs for these two uses when using metrics. The relationship between these two sets of MPRs is considered in Section 6.3.

### 6.1. Flooding MPRs

The essential detail of the "flooding MPR" selection specification is that a router must select a set of MPRs such that a message transmitted by a router, and re-transmitted by all its flooding MPRs, will reach all of the selecting router's symmetric 2-hop neighbors.

Flooding MPR selection can ignore metrics and produce a solution that meets the required specification. However, that does not mean that metrics cannot be usefully considered in selecting flooding MPRs. Consider the network in Figure 2, where numbers are metrics of links in the direction away from router A, towards router D.

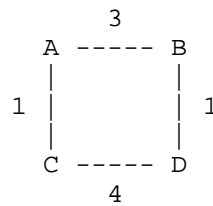


Figure 2

Which is the better flooding MPR selection by router A: B or C? If the metric represents probability of message loss, then clearly choosing B maximizes the probability of a message sent by A reaching D. This is despite that C has a lower metric in its connection to A than B does. (Similar arguments about a preference for B can be made if, for example, the metric represents data rate or delay rather than probability of loss.)

However, neither should only the second hop be considered. If this example is modified to that in Figure 3, where the numbers still are metrics of links in the direction away from router A, towards router D:

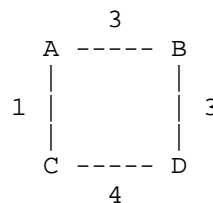


Figure 3

then it is possible that, when A is selecting flooding MPRs, selecting C is preferable to selecting B. If the metrics represent scaled values of delay, or the probability of loss, then selecting C is clearly better. This indicates that the sum of metrics is an appropriate measure to use to choose between B and C.

However, this is a particularly simple example. Usually it is not a simple choice between two routers as a flooding MPR, each only adding one router coverage. A more general process, when considering which router to next add as a flooding MPR, should incorporate the metric to that router, and the metric from that router to each symmetric 2-hop neighbor, as well as the number of newly covered symmetric 2-hop neighbors, and may include other factors.

The required specification for flooding MPR selection is in Section

18.4 (also using Section 18.3) of [OLSRv2]. which may use the example MPR selection algorithm in Appendix B of [OLSRv2]. However, note that (as in [RFC3626]) each router can make its own independent choice of flooding MPRs, and flooding MPR selection algorithm, and still interoperate.

Also note that the references above to the direction of the metrics is correct: for flooding, directional metrics outward from a router are appropriate, i.e., metrics in the direction of the flooding. This is an additional reason for including outward metrics in HELLO messages, as otherwise a metric-aware MPR selection for flooding is not possible. The second hop metrics are outgoing neighbor metrics because the OLSRv2 interface used for a second hop transmission may not be the same as that used for the first hop reception.

## 6.2. Routing MPRs

The essential detail of the "routing MPR" selection specification is that a router must, per OLSRv2 interface, select a set of MPRs such that there is a two hop route from each symmetric 2-hop neighbor of the selecting router to the selecting router, with the intermediate router on each such route being a routing MPR of the selecting router.

It is sufficient, when using an additive link metric rather than a hop count, to require that these routing MPRs provide not just a two hop route, but a minimum distance two hop route. In addition, a router is a symmetric 2-hop neighbor even if it is a symmetric 1-hop neighbor, as long as there is a two hop route from it that is shorter than the one hop link from it. (The property that no routes go through routers with willingness WILL\_NEVER is retained. Examples below assume that all routers are equally willing, with none having willingness WILL\_NEVER.)

For example, consider the network in Figure 4. Numbers are metrics of links in the direction towards router A, away from router D. Router A must pick router B as a routing MPR, whereas for minimum hop count routing it could alternatively pick router C. Note that the use of incoming neighbor metrics in this case follows the same reasoning as for the directionality of metrics in TC messages, as described in Section 5.4.

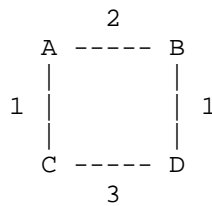


Figure 4

In Figure 5, where numbers are metrics of links in the direction towards router A, away from router C, router A must pick router B as a routing MPR, but for minimum hop count routing it would not need to pick any MPRs.

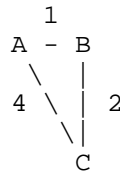


Figure 5

In Figure 6, where numbers are metrics of links in the direction towards router A, away from routers D and E, router A must pick both routers B and C as routing MPRs, but for minimum hop count routing it could pick either.

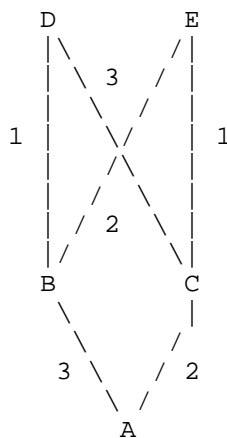


Figure 6

It is shown in Appendix A that selecting routing MPRs according to

this definition, and advertising only such links (plus knowledge of local links from HELLO messages), will result in selection of lowest total metric routes, even if all links (advertised or not) are considered in the definition of a shortest route.

However the definition noted above as sufficient for routing MPR selection is not necessary. For example, consider the network in Figure 7, where numbers are metrics of links in the direction towards router A, away from other routers; the metrics from B to C and C to B are both assumed to be 2.

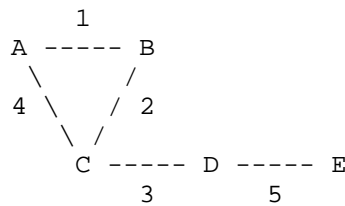


Figure 7

Using the above definition, A must pick both B and C as routing MPRs, in order to cover the symmetric 2-hop neighbors C and D, respectively. (C is a symmetric 2-hop neighbor because the route length via B is shorter than the 1-hop link.)

However, A only needs to pick B as a routing MPR, because the only reason to pick C as a routing MPR would be so that C can advertise the link to A for routing - to be used by, for example, E. But A knows that no other router should use the link C to A in a shortest route, because routing via B is shorter. So if there is no need to advertise the link from C to A, then there is no reason for A to select C as a routing MPR.

This process of "thinning out" the routing MPR selection uses only local information from HELLO messages. Using any minimum distance algorithm, the router identifies shortest routes, whether one, two or more hops, from all routers in its symmetric 2-hop neighborhood. It then selects as MPRs all symmetric 1-hop neighbors that are the last router (before the selecting router itself) on any such route. Where there is more than one shortest distance route from a router, only one such route is required. Alternative routes may be selected so as to minimize the number of last routers - this is the equivalent to the selection of a minimal set of MPRs in the non-metric case.

Note that this only removes routing MPRs whose selection can be directly seen to be unnecessary. Consequently if (as is shown in Appendix A) the first approach creates minimum distance routes, then



so does this process.

The examples in Figure 5 and Figure 6 show that use of link metrics may require a router to select more routing MPRs than when not using metrics, and even require a router to select routing MPRs when without metrics it would not need any routing MPRs. This may result in more, and larger, messages being generated, and forwarded more often. Thus the use of link metrics is not without cost, even excluding the cost of link metric signaling.

These examples consider only single OLSRv2 interface routers. However if routers have more than one OLSRv2 interface, then the process is unchanged, other than that if there is more than one known metric between two routers (on different OLSRv2 interfaces), then, considering symmetric links only (as only these are used for routing) the smallest link metric, i.e., the neighbor metric, is used. There is no need to calculate routing MPRs per OLSRv2 interface. That requirement results from the consideration of flooding and the need to avoid certain "race" conditions, which are not relevant to routing, only to flooding.

The required specification for routing MPR selection is in Section 18.5 (also using Section 18.3) of [OLSRv2]. which may use the example MPR selection algorithm in Appendix B of [OLSRv2]. However, note that (as in [RFC3626]) each router can make its own independent choice of routing MPRs, and routing MPR selection algorithm, and still interoperate.

### 6.3. Relationship Between MPR Sets

It would be convenient if the two sets of flooding and routing MPRs were the same. This can be the case if all metrics are equal, but in general, for "good" sets of MPRs they are not. (A reasonable definition of this is that there is no common minimal set of MPRs.) If metrics are asymmetrically valued (the two sets of MPRs use opposite direction metrics), or routers have multiple OLSRv2 interfaces (where routing MPRs can ignore this, but flooding MPRs cannot) this is particularly unlikely. However even using a symmetrically valued metric with a single OLSRv2 interface on each router, the ideal sets need not be equal, nor is one always a subset of the other. To show this, consider these examples, where all lettered routers are assumed equally willing to be MPRs, and numbers are bidirectional metrics for links.

In Figure 8, A does not require any flooding MPRs. However A must select B as a routing MPR.

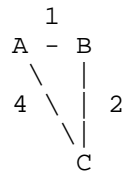


Figure 8

In Figure 9, A must select C and D as routing MPRs. However A's minimal set of flooding MPRs is just B. In this example the set of routing MPRs serves as a set of flooding MPRs, but a non-minimal one (although one that might be better, depending on the relative importance of number of MPRs and flooding link metrics).

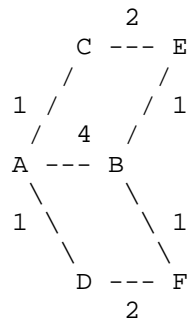


Figure 9

However, this is not always the case. In Figure 10, A's set of routing MPRs must contain B, but need not contain C. A's set of flooding MPRs need not contain B, but must contain C. (In this case, flooding with A selecting B rather than C as a flooding MPR will reach D, but in three hops rather than the minimum two that MPR flooding guarantees.)

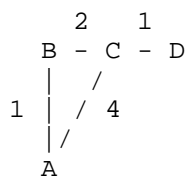


Figure 10

## 7. IANA Considerations

This document has no actions for IANA.

This section may be removed by the RFC Editor.

## 8. Security Considerations

An attacker can have an adverse impact on an OLSRv2 network by creating apparently valid messages that contain incorrect link metrics. This could take the form of influencing the choice of routes, or in some cases producing routing loops. This is a more subtle, and likely to be less effective, attack, than other forms of invalid message injection. These can add and remove other and more basic forms of network information, such as the existence of some routers and links.

As such, no significantly new security issues arose from the inclusion of metrics in OLSRv2. Defenses to the injection of invalid link metrics are the same as to other forms of invalid message injection, as discussed in the security considerations section of [OLSRv2].

There are possible uses for link metrics in the creation of security countermeasures, to prefer the use of links that have better security properties, including better availability, to those with poorer security properties. This however is beyond the scope of both this document and [OLSRv2].

## 9. Acknowledgements

The authors would like to gratefully acknowledge the following people for intense technical discussions, early reviews and comments on the documents and its components (listed alphabetically): Brian Adamson (NRL), Alan Cullen (BAE Systems), Justin Dean (NRL), Ulrich Herberg (Fujitsu), Stan Ratliff (Cisco), Charles Perkins (Huawei), and Henning Rogge (FGAN).

Finally, the authors would like to express their gratitude to (listed alphabetically) Benoit Claise, Adrian Farrel, Stephen Farrell and Suresh Krishnan for their reviews and comments on the later versions of this document.

## 10. Informative References

- [RFC2501] Macker, J. and S. Corson, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations", RFC 2501, January 1999.
- [RFC3626] Clausen, T. and P. Jacquet, "The Optimized Link State Routing Protocol", RFC 3626, October 2003.
- [RFC5444] Clausen, T., Dean, J., Dearlove, C., and C. Adjih, "Generalized MANET Packet/Message Format", RFC 5444, February 2009.
- [RFC6130] Clausen, T., Dean, J., and C. Dearlove, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)", RFC 6130, April 2011.
- [OLSRv2] Clausen, T., Dearlove, C., Jacquet, P., and U. Herberg, "The Optimized Link State Routing Protocol version 2", draft-ietf-manet-olsrv2-19.txt (work in progress), March 2013.

## Appendix A. MPR Routing Property

In order that routers can find and use shortest routes in a network while using the minimum reduced topology supported by OLSRv2 (that a router only advertises its MPR selectors in TC messages), routing MPR selection must result in the property that there are shortest routes with all intermediate routers being routing MPRs.

This appendix uses the following terminology and assumptions:

- o The network is a graph of nodes connected by arcs, where nodes correspond to routers with willingness not equal to WILL\_NEVER (except possibly at the ends of routes). An arc corresponds to the set of symmetric links connecting those routers; the OLSRv2 interfaces used by those links are not relevant.
- o Each arc has a metric in each direction, being the minimum of the corresponding link metrics in that direction, i.e., the corresponding neighbor metric. This metric must be positive.
- o A sequence of arcs joining two nodes is referred to as a path.
- o Node A is an MPR of node B, if corresponding router A is a routing MPR of router B.

The required property (of using shortest routes with reduced topology) is equivalent to that for any pair of distinct nodes X and Z there is a shortest path from X to Z,  $X - Y_1 - Y_2 - \dots - Y_m - Z$  such that  $Y_1$  is an MPR of  $Y_2$ , ...  $Y_m$  is an MPR of Z. Call such a path a routable path, and call this property the routable path property.

The required definition for a node X selecting MPRs is that for each distinct node Z from which there is a two arc path, there is a shorter, or equally short, path which is either  $Z - Y - X$  where Y is an MPR of X, or is the one arc path  $Z - X$ . Note that the existence of locally known, shorter, but more than two arc paths, which can be used to reduce the numbers of MPRs, is not considered here. (Such reductions are only when the remaining MPRs can be seen to retain all necessary shortest paths, and therefore retains the required property.)

Although this appendix is concerned with paths with minimum total metric, not number of arcs (hop count), it proceeds by induction on the number of arcs in a path. Although it considers minimum metric routes with a bounded number of arcs, it then allows that number of arcs to increase so that overall minimum metric paths, regardless of the number of arcs, are considered.

Specifically, the routable path property is a corollary of the property that for all positive integers  $n$ , and all distinct nodes  $X$  and  $Z$ , if there is any path from  $X$  to  $Z$  of  $n$  arcs or fewer, then there is a shortest path, from among those of  $n$  arcs or fewer, that is a routable path. This may be called the  $n$ -arc routable path property.

The  $n$ -arc routable path property is trivial for  $n = 1$ , and directly follows from the definition of the MPRs of  $Z$  for  $n = 2$ .

Proceeding by induction, assuming the  $n$ -arc routable path property is true for  $n = k$ , consider the case that  $n = k+1$ .

Suppose that  $X - V_1 - V_2 - \dots - V_k - Z$  is a shortest  $k+1$  arc path from  $X$  to  $Z$ . We construct a path which has no more than  $k+1$  arcs, has the same or shorter length (hence has the same, shortest, length considering only paths of up to  $k+1$  arcs, by assumption) and is a routable path.

First consider whether  $V_k$  is an MPR of  $Z$ . If it is not then consider the two arc path  $V_{k-1} - V_k - Z$ . This can be replaced either by a one arc path  $V_{k-1} - Z$  or by a two arc path  $V_{k-1} - W_k - Z$  where  $W_k$  is an MPR of  $Z$ , such that the metric from  $V_{k-1}$  to  $Z$  by the replacement path is no longer. In the former case (replacement one arc path) this now produces a path of length  $k$ , and the previous inductive step may be applied. In the latter case we have replaced  $V_k$  by  $W_k$ , where  $W_k$  is an MPR of  $Z$ . Thus we need only consider the case that  $V_k$  is an MPR of  $Z$ .

We now apply the previous inductive step to the path  $X - V_1 - \dots - V_{k-1} - V_k$ , replacing it by an equal length path  $X - W_1 - \dots - W_{m-1} - V_k$ , where  $m \leq k$ , where this path is a routable path. Then because  $V_k$  is an MPR of  $Z$ , the path  $X - W_1 - \dots - W_{m-1} - V_k - Z$  is a routable path, and demonstrates the  $n$ -arc routable path property for  $n = k+1$ .

This thus shows that for any distinct nodes  $X$  and  $Z$ , there is a routable path using the MPR-reduced topology from  $X$  to  $Z$ , i.e., that OLSRv2 finds minimum length paths (minimum total metric routes).



Authors' Addresses

Christopher Dearlove  
BAE Systems Advanced Technology Centre  
West Hanningfield Road  
Great Baddow, Chelmsford  
United Kingdom

Phone: +44 1245 242194  
EMail: [chris.dearlove@baesystems.com](mailto:chris.dearlove@baesystems.com)  
URI: <http://www.baesystems.com/>

Thomas Heide Clausen  
LIX, Ecole Polytechnique  
91128 Palaiseau Cedex  
France

Phone: +33 6 6058 9349  
EMail: [T.Clausen@computer.org](mailto:T.Clausen@computer.org)  
URI: <http://www.thomasclausen.org/>

Philippe Jacquet  
Alcatel-Lucent Bell Labs

Phone: +33 6 7337 1880  
EMail: [philippe.jacquet@alcatel-lucent.com](mailto:philippe.jacquet@alcatel-lucent.com)



Internet Engineering Task Force  
Internet-Draft  
Intended status: Standards Track  
Expires: December 26, 2013

U. Herberg  
Fujitsu Laboratories of America  
R. Cole  
US Army CERDEC  
T. Clausen  
LIX, Ecole Polytechnique  
June 24, 2013

Definition of Managed Objects for the Optimized Link State Routing  
Protocol version 2  
draft-ietf-manet-olsrv2-mib-12

Abstract

This document defines the Management Information Base (MIB) module for configuring and managing the Optimized Link State Routing protocol version 2 (OLSRv2). The OLSRv2-MIB module is structured into configuration information, state information, performance information, and notifications. This additional state and performance information is useful to troubleshoot problems and performance issues of the routing protocol. Two levels of compliance allow this MIB module to be deployed on constrained routers.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on December 26, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. The Internet-Standard Management Framework . . . . .	3
3. Conventions . . . . .	3
4. Overview . . . . .	3
4.1. Terms . . . . .	4
5. Structure of the MIB Module . . . . .	4
5.1. The Configuration Group . . . . .	5
5.2. The State Group . . . . .	5
5.3. The Performance Group . . . . .	5
5.4. The Notifications Group . . . . .	6
5.5. Tables and Indexing . . . . .	6
6. Relationship to Other MIB Modules . . . . .	9
6.1. Relationship to the SNMPv2-MIB . . . . .	9
6.2. Relationship to the NHDP-MIB . . . . .	9
6.3. MIB modules required for IMPORTS . . . . .	9
7. Definitions . . . . .	10
8. Security Considerations . . . . .	78
9. Applicability Statement . . . . .	80
10. IANA Considerations . . . . .	81
11. Acknowledgements . . . . .	82
12. References . . . . .	82
12.1. Normative References . . . . .	82
12.2. Informative References . . . . .	83
Appendix A. Appendix A: . . . . .	84
Appendix B. Note to the RFC Editor . . . . .	86

## 1. Introduction

This document defines the Management Information Base (MIB) module for configuring and managing the Optimized Link State Routing protocol version 2 (OLSRv2). The OLSRv2-MIB module is structured into configuration information, state information, performance information, and notifications. In addition to configuration, this additional state and performance information is useful to troubleshoot problems and performance issues of the routing protocol. Different levels of compliance allow implementers to use smaller subsets of all defined objects, allowing for this MIB module to be deployed on more constrained routers.

## 2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to Section 7 of [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB module are defined using the mechanisms defined in the Structure of Management Information (SMI). This document specifies a MIB module that is compliant to the SMIV2, which is described in [RFC2578], [RFC2579], and [RFC2580].

## 3. Conventions

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].

## 4. Overview

The Optimized Link State Routing Protocol version 2 (OLSRv2) [OLSRv2] is a table driven, proactive routing protocol, i.e., it exchanges topology information with other routers in the network periodically. OLSRv2 is an optimization of the classical link state routing protocol. Its key concept is that of MultiPoint Relays (MPRs). Each router selects a set of its neighbor routers (which "cover" all of its symmetrically connected 2-hop neighbor routers) as MPRs. MPRs are then used to achieve both flooding reduction and topology reduction.

This document provides management and control capabilities of an OLSRv2 instance, allowing management applications to monitor the

state and performance of an OLSRv2 router, as well as to change settings of the OLSRv2 instance (e.g., router or interface parameters such as message intervals, etc.).

As OLSRv2 relies on the neighborhood information discovered by the "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)" [RFC6130], the OLSRv2-MIB module is aligned with the NHDP-MIB [RFC6779] module and augments several of the tables and objects in the NHDP-MIB. In particular, common indexes for router interfaces and discovered neighbors are used, as described in Section 5.2.

#### 4.1. Terms

The following definitions apply throughout this document:

- o Configuration Objects - switches, tables, objects which are initialized to default settings or set through the management interface defined by this MIB module.
- o State Objects - automatically generated values which define the current operating state of the OLSRv2 protocol instance in the router.
- o Performance Objects - automatically generated values which help an administrator or automated tool to assess the performance of the OLSRv2 routing process on the router.
- o Notification Objects - define triggers and associated notification messages allowing for asynchronous tracking of pre-defined events on the managed router.

#### 5. Structure of the MIB Module

This section presents the structure of the OLSRv2-MIB module. The objects are arranged into the following structure:

- o `olsrv2MIBObjects` - defines objects forming the basis for the OLSRv2-MIB module. These objects are divided up by function into the following groups:
  - \* Configuration Group - defining objects related to the configuration of the OLSRv2 instance on the router.
  - \* State Group - defining objects which reflect the current state of the OLSRv2 instance running on the router.
  - \* Performance Group - defining objects which are useful to a management station when characterizing the performance of

OLSRv2 on the router and in the MANET.

- o `olsrv2MIBNotifications` - objects defining OLSRv2-MIB module notifications.
- o `olsrv2MIBConformance` - defining the minimal and maximal conformance requirements for implementations of this MIB module.

#### 5.1. The Configuration Group

The OLSRv2 router is configured with a set of controls. The authoritative list of configuration controls within the OLSRv2-MIB module is found within the MIB module itself. Generally, an attempt was made in developing the OLSRv2-MIB module to support all configuration objects defined in [OLSRv2]. For all of the configuration parameters, the same constraints and default values of these parameters as defined in [OLSRv2] are followed.

#### 5.2. The State Group

The State Group reports current state information of a router running [OLSRv2]. The OLSRv2-MIB module State Group tables were designed to contain the complete set of state information defined within the information bases in [OLSRv2].

The OLSRv2-MIB module State Group tables are constructed as extensions to the corresponding tables within the State Group of the NHDP-MIB [RFC6779] module. Use of the AUGMENTS clause is made, when possible, to accomplish these table extensions. Further, the State Group tables defined in this MIB module are aligned with the according tables in the NHDP-MIB [RFC6779] module, as described in Section 6.2.

#### 5.3. The Performance Group

The Performance Group reports values relevant to system performance. Frequent changes of sets or frequent recalculation of the routing set or the MPRs can have a negative influence on the performance of OLSRv2. This MIB module defines several objects that can be polled in order to, e.g., calculate histories or monitor frequencies of changes. This may help the network administrator to determine unusual topology changes or other changes that affect stability and reliability of the MANET. One such framework is specified in REPORT-MIB [REPORT-MIB].

#### 5.4. The Notifications Group

The Notifications Group contains Control (olsrv2NotificationsControl), Objects (olsrv2NotificationsObjects) and States (olsrv2NotificationsStates), where the Control contains definitions of objects to control the frequency of notifications being generated. The Objects define the supported notifications and the State is used to define additional information to be carried within the notifications.

The olsrv2NotificationsObjects sub-tree contains the list of notifications supported within the OLSRv2-MIB module and their intended purpose or utility.

The same mechanisms for improving the network performance by reducing the number of notifications apply as defined in Section 5.1 of [RFC6779]. The following objects are used to define the thresholds and time windows for specific notifications defined in the NHDP-MIB module: olsrv2RoutingSetRecalculationCountThreshold, olsrv2RoutingSetRecalculationCountWindow, olsrv2MPRSetRecalculationCountThreshold, and olsrv2MPRSetRecalculationCountWindow.

#### 5.5. Tables and Indexing

The OLSRv2-MIB module's tables are indexed by the following constructs:

- o nhdpIfIndex - the ifIndex of the local router on which NHDP is configured. This is defined in the NHDP-MIB.
- o nhdpDiscIfIndex - a locally managed index representing a known interface on a neighboring router. This is defined in the NHDP-MIB.
- o nhdpDiscRouterIndex - a locally managed index representing an ID of a known neighboring router. This is defined in the NHDP-MIB.
- o {olsrv2LibOrigSetIpAddressType, olsrv2LibOrigSetIpAddress} - this index (pair) uniquely identifies recently used originator addresses found within the olsrv2LibOrigSetTable.
- o {olsrv2LibLocAttNetSetIpAddressType, olsrv2LibLocAttNetSetIpAddress, olsrv2LibLocAttNetSetIpAddressPerfixLen} - this index (triplet) uniquely identifies local attached networks reachable through local (non-OLSRv2) interfaces on this router. These are recorded in the olsrv2LibLocAttNetSetTable.



- o {olsrv2TibAdRemoteRouterSetIpAddrType, olsrv2TibAdRemoteRouterSetIpAddr} - this index (pair) uniquely identifies each router in the network that transmits TC messages received by this router. These records are recorded in the olsrv2TibAdRemoteRouterSetIpAddr.
- o {olsrv2TibRouterTopologySetFromOrigIpAddrType, olsrv2TibRouterTopologySetFromOrigIpAddr, olsrv2TibRouterTopologySetToOrigIpAddrType, olsrv2TibRouterTopologySetToOrigIpAddr} - this index (quadruplet) uniquely identifies discovered links within the network recorded by this router. Information associated with each link is stored in the olsrv2TibRouterTopologySetTable.
- o {olsrv2TibRoutableAddressTopologySetFromOrigIpAddrType, olsrv2TibRoutableAddressTopologySetFromOrigIpAddr, olsrv2TibRoutableAddressTopologySetFromDestIpAddrType, olsrv2TibRoutableAddressTopologySetFromDestIpAddr} - this index (quadruplet) uniquely identifies reachable addresses within the network and the router's advertising these addresses. This information is stored in the olsrv2TibRoutableAddressTopologySetTable.
- o {olsrv2TibAttNetworksSetOrigIpAddrType, olsrv2TibAttNetworksSetOrigIpAddr, olsrv2TibAttNetworksSetNetIpAddrType, olsrv2TibAttNetworksSetNetIpAddr, olsrv2TibAttNetworksSetNetIpAddrPrefixLen} - this index (quintuplet) uniquely identifies the networks (which may be outside the MANET) and the routers through which these networks can be reached. This information is stored in the olsrv2TibAttNetworksSetTable.
- o {olsrv2TibRoutingSetDestIpAddrType, olsrv2TibRoutingSetDestIpAddr, olsrv2TibRoutingSetDestIpAddrPrefixLen} - this index (triplet) uniquely identifies the address of a reachable destination in the network. This indexes the olsrv2TibRoutingSetTable which contains the next hop information to reach the indexed addresses.

These tables and their indexing are:

- o olsrv2InterfaceTable - describes the OLSRv2 status on the NHDP interfaces of this router. This table augments nhdpInterfaceEntry and as such it is indexed by the {nhdpIfIndex} from the NHDP-MIB.
- o olsrv2IibLinkSetTable - records all links from other routers which are, or recently were, 1-hop neighbors. This table augments nhdpIibLinkSetEntry and as such it is indexed by nhdpIfIndex and

nhdpDiscIfIndex.

- o olsrv2Iib2HopSetTable - records network addresses of symmetric 2-hop neighbors and the links to the associated 1-hop neighbors. This table augments nhdpIib2HopSetEntry and as such it is indexed by {nhdpIfIndex, nhdpDiscIfIndex, nhdpIib2HopSetIpAddressType, nhdpIib2HopSetIpAddress}.
- o olsrv2LibOrigSetTable - records addresses that were recently used as originator addresses by this router. This table is indexed by {olsrv2LibOrigSetIpAddrType, olsrv2LibOrigSetIpAddr}.
- o olsrv2LibLocAttNetSetTable - records its local non-OLSRv2 interfaces via which it can act as gateways to other networks. This table is indexed by {olsrv2LibLocAttNetSetIpAddrType, olsrv2LibLocAttNetSetIpAddr, olsrv2LibLocAttNetSetIpAddrPerfixLen}.
- o olsrv2NibNeighborSetTable - records all network addresses of each 1-hop neighbor. This table augments nhdpNibNeighborSetEntry and as such it is indexed by the {nhdpDiscRouterIndex}.
- o olsrv2TibAdRemoteRouterSetTable - records information describing each remote router in the network that transmits TC messages. This table is indexed by {olsrv2TibAdRemoteRouterSetIpAddrType, olsrv2TibAdRemoteRouterSetIpAddr}.
- o olsrv2TibRouterTopologySetTable - records topology information about the network. This table is indexed by {olsrv2TibRouterTopologySetFromOrigIpAddrType, olsrv2TibRouterTopologySetFromOrigIpAddr, olsrv2TibRouterTopologySetToOrigIpAddrType, olsrv2TibRouterTopologySetToOrigIpAddr}.
- o olsrv2TibRoutableAddressTopologySetTable - records topology information about the routable addresses within the MANET, and via which routers they may be reached. This table is indexed by {olsrv2TibRoutableAddressTopologySetFromOrigIpAddrType, olsrv2TibRoutableAddressTopologySetFromOrigIpAddr, olsrv2TibRoutableAddressTopologySetFromDestIpAddrType, olsrv2TibRoutableAddressTopologySetFromDestIpAddr}.
- o olsrv2TibAttNetworksSetTable - records information about networks (which may be outside the MANET) attached to other routers and their routable addresses. This table is indexed by {olsrv2TibAttNetworksSetOrigIpAddrType, olsrv2TibAttNetworksSetOrigIpAddr, olsrv2TibAttNetworksSetNetIpAddrType,

```
olsrv2TibAttNetworksSetNetIpAddress,  
olsrv2TibAttNetworksSetNetIpAddressPrefixLen}.
```

- o olsrv2TibRoutingSetTable - records the first hop along a selected path to each destination for which any such path is known. This table is indexed by {olsrv2TibRoutingSetDestIpAddressType, olsrv2TibRoutingSetDestIpAddress, olsrv2TibRoutingSetDestIpAddressPrefixLen}.
- o olsrv2InterfacePerfTable - records performance counters for each active OLSRv2 interface on this device. This table augments nhdpInterfacePerfEntry and as such it is indexed by {nhdpIfIndex} from the NHDP-MIB.

## 6. Relationship to Other MIB Modules

This section specifies the relationship of the MIB modules contained in this document to other standards, particularly to standards containing other MIB modules. MIB modules and specific definitions imported from MIB modules that SHOULD be implemented in conjunction with the MIB module contained within this document are identified in this section.

### 6.1. Relationship to the SNMPv2-MIB

The System group in the SNMPv2-MIB [RFC3418] module is defined as being mandatory for all systems, and the objects apply to the entity as a whole. The System group provides identification of the management entity and certain other system-wide data. The OLSRv2-MIB module does not duplicate those objects.

### 6.2. Relationship to the NHDP-MIB

OLSRv2 depends on the neighborhood information that is discovered by [RFC6130]. An instance of OLSRv2 MUST have an associated instance of NHDP running on the same device for proper operations of the discovery and routing system. In order for the OLSRv2-MIB module to correctly populate the objects relating to discovered neighbors, the State Group tables of the NHDP-MIB [RFC6779] module are aligned with the State Group tables of this MIB module. This is accomplished through the use of the AUGMENTS capability of SMIV2 (where appropriate). This will allow for cross referencing of information between the two MIB modules within a given SNMP context.

### 6.3. MIB modules required for IMPORTS

The following OLSRv2-MIB module IMPORTS objects from NHDP-MIB [RFC6779], SNMPv2-SMI [RFC2578], SNMPv2-TC [RFC2579], SNMPv2-CONF

[RFC2580], IF-MIB [RFC2863] and INET-ADDRESS-MIB [RFC4001]. The OLSRv2-MIB module also IMPORTS objects from the IANAolsrv2LinkMetricType-MIB which is defined in Appendix A of this document.

## 7. Definitions

This section contains the OLSRv2-MIB module defined by the specification.

```
OLSRv2-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```
    MODULE-IDENTITY, OBJECT-TYPE, Counter32, Counter64,  
    Integer32, Unsigned32, mib-2, TimeTicks,  
    NOTIFICATION-TYPE  
        FROM SNMPv2-SMI -- RFC 2578
```

```
    TEXTUAL-CONVENTION, TimeStamp, TruthValue  
        FROM SNMPv2-TC -- RFC 2579
```

```
    MODULE-COMPLIANCE, OBJECT-GROUP, NOTIFICATION-GROUP  
        FROM SNMPv2-CONF -- STD 58
```

```
    InetAddressType, InetAddress,  
    InetAddressPrefixLength  
        FROM INET-ADDRESS-MIB -- RFC 3291
```

```
    nhdpInterfaceEntry,  
    nhdpIibLinkSetEntry, nhdpIib2HopSetEntry,  
    nhdpNibNeighborSetEntry, nhdpInterfacePerfEntry  
        FROM NHDP-MIB -- RFC 6779
```

```
    IANAolsrv2LinkMetricTypeTC  
        FROM IANAolsrv2LinkMetricType-MIB  
    ;
```

```
manetOlsrv2MIB MODULE-IDENTITY
```

```
    LAST-UPDATED "201306240000Z" --24 June 2013
```

```
    ORGANIZATION "IETF MANET Working Group"
```

```
    CONTACT-INFO
```

```
        "WG E-Mail: manet@ietf.org"
```

```
        WG Chairs: sratliff@cisco.com
```

```
                  jmacker@nrl.navy.mil
```

Editors: Ulrich Herberg  
Fujitsu Laboratories of America  
1240 East Arques Avenue  
Sunnyvale, CA 94085  
USA  
ulrich@herberg.name  
<http://www.herberg.name/>

Thomas Heide Clausen  
Ecole Polytechnique  
LIX  
91128 Palaiseau Cedex  
France  
<http://www.thomasclausen.org/>  
T.Clausen@computer.org

Robert G. Cole  
US Army CERDEC  
Space and Terrestrial Communications  
6010 Frankford Street  
Bldg 6010, Room 453H  
Aberdeen Proving Ground, MD 21005  
USA  
+1 443 395-8744  
robert.g.cole@us.army.mil  
<http://www.cs.jhu.edu/~rgcole/>

#### DESCRIPTION

"This OLSRv2-MIB module is applicable to routers implementing the Optimized Link State Routing Protocol version 2 (OLSRv2) defined in RFC XXXX.

Copyright (c) 2013 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in Section 4.c of the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>).

This version of this MIB module is part of RFC YYYY; see the RFC itself for full legal notices."

-- Revision History  
REVISION "201306240000Z" -- 24 June 2013  
DESCRIPTION

```
"Initial version of this MIB module,
published as RFC YYYY."

-- RFC-Editor assigns ZZZZ (this comment can be removed)
 ::= { mib-2 ZZZZ }

--
-- TEXTUAL CONVENTIONS
--

Olsrv2MetricValueCompressedFormTC ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "d"
    STATUS          current
    DESCRIPTION
        "OLSRv2 Metrics are expressed in terms of a Link Metric
        Compressed Form within the OLSRv2 protocol. This textual
        convention defines the syntax of the metric objects
        consistent with the definitions of the OLSRv2 Link
        Metric Compressed Form in Section 6.2 of RFC XXXX.

        The 12-bit compressed form of a link metric uses a modified
        form of a representation with an 8-bit mantissa (denoted a)
        and a 4-bit exponent (denoted b). Note that if represented
        as the 12 bit value 256b+a then the ordering of those 12 bit
        values is identical to the ordering of the represented values.

        The value so represented is  $(257+a)2^b - 256$ , where ^ denotes
        exponentiation. This has a minimum value
        (when a = 0 and b = 0) of MINIMUM_METRIC = 1 and a maximum
        value (when a = 255 and b = 15) of MAXIMUM_METRIC =  $2^{24} - 256$ .

        Hence the compressed form metric values range from 1 to
        16776960. The special value of 0 is reserved for the
        UNKNOWN_METRIC value.

        If a network manager sets the metric value 'm' through the
        MIB-module, then the OLSRv2 code can derive 'compressed_m'
        = M(a,b) according to the algorithm in RFC 5497 and
        'compressed_m' is the value represented in the OLSRv2 messages.
        But the value 'm' is persistently stored by the MIB-module.
        If the MIB-module is pulling this time parameter from some other
        source, i.e., the protocol instance, then this value is stored
        as is."
    SYNTAX  Unsigned32 (0..16776960)

Olsrv2TimeValueCompressedForm32TC ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "x"
```

STATUS           current  
DESCRIPTION

"OLSRv2 time values may be expressed in terms of a compressed form within the OLSRv2 protocol. This textual convention defines the syntax of the time objects defined in terms of an interger number of millisceonds, consistent with the definitions of the 8-bit exponent-mantissa compressed form defined in Section 5 of RFC 5497. Time values with this representation are defined in terms of a constant C which is represented in terms of seconds. The constant C (time granularity) is used as specified in RFC 5497. It MUST be the same as is used by the NHDP protocol RFC 6130.

The 8-bit compressed form of a time value uses a modified form of a representation with an 3-bit mantissa (denoted a) and a 5-bit exponent (denoted b). Note that if represented as the 8 bit value  $8b+a$  then the ordering of those 8 bit values is identical to the ordering of the represented values.

The minimum time-value that can be represented in this manner is C. The maximum time-value that can be represented in this manner is  $15 * 2^{28} * C$ ,  $15 * 268,435,456 * C$ ,  $4,026,531,840 * C$  or about 45 days if, for example,  $C = 1/1024$  second.

This TEXTUAL-CONVENTION limits the maximum value of the time granularity constant C to be no greater than 1/1024 seconds due to its use of the Unsigned32 syntax limiting the maximum number of milliseconds to no more than 3932160000.

When OLSRv2 uses this 8-bit exponent-mantissa compressed form, this object value MUST be translated from the integer form represented in this MIB-module into the exponent-mantissa form for the OLSRv2 protocol to use according to the algorithm defined in Section 5 of RFC 5497 for finding the next larger time value within the exponent-mantissa format.

If a network manager sets the time value 't' through the MIB-module, then the OLSRv2 code can derive 'compressed\_t' = T(a,b) according to the algorithm in RFC 5497 and 'compressed\_t' is the value represented in the OLSRv2 messages. But the value 't' is persistently stored by the MIB-module. If the MIB-module is pulling this time parameter from some other source which is using the compressed form, i.e., the protocol instance, then

this value is stored as is, after converting from  
number of time constants C into number of milliseconds."  
SYNTAX Unsigned32 (1..3932160000)

Olsrv2StatusTC ::= TEXTUAL-CONVENTION  
STATUS current  
DESCRIPTION  
"Controls the operation of the OLSRv2  
protocol on the device or a specific interface.  
For example, for an interface: 'enabled' indicates  
that OLSRv2 is permitted to operate,  
and 'disabled' indicates that it is not."  
SYNTAX INTEGER {  
enabled (1),  
disabled (2)  
}

WillingnessTC ::= TEXTUAL-CONVENTION  
DISPLAY-HINT "x"  
STATUS current  
DESCRIPTION  
"A willingness value which evaluates to the  
device's interest in participating in  
a particular function, process or behavior.  
  
The willingness ranges from a low value of  
WILL\_NEVER(0) to a high value of  
WILL\_ALWAYS(15). For each parameter x,  
there is an associated willingness value  
W(x) such that WILL\_NEVER < W(x) <= WILL\_ALWAYS."  
SYNTAX Unsigned32 (0..15)

--

-- Top-Level Object Identifier Assignments

--

olsrv2MIBNotifications OBJECT IDENTIFIER ::= { manetOlsrv2MIB 0 }  
olsrv2MIBObjects OBJECT IDENTIFIER ::= { manetOlsrv2MIB 1 }  
olsrv2MIBConformance OBJECT IDENTIFIER ::= { manetOlsrv2MIB 2 }

--

-- olsrv2ConfigurationGroup

--

-- Contains the OLSRv2 objects that configure specific  
-- options that determine the overall performance and operation  
-- of the OLSRv2 routing process.



olsrv2ConfigurationGroup OBJECT IDENTIFIER ::= {olsrv2MIBObjects 1}

olsrv2AdminStatus OBJECT-TYPE

SYNTAX Olsrv2StatusTC

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The configured status of the OLSRv2 process on this device. 'enabled(1)' means that OLSRv2 is configured to run on this device. 'disabled(2)' mean that the OLSRv2 process is configured off.

Operation of the OLSRv2 routing protocol requires the operation of the Neighborhood Discovery Protocol (RFC 6130). Hence, this object cannot have a status of 'enabled' unless at least one interface on the device is a MANET interface with NHDP enabled on that interface. If a network manager attempts to set this object to 'enabled' when no interfaces on this device have NHDP enabled, the device MUST fail the set with inconsistentValue. If all device interfaces running NHDP become disabled or removed, then the olsrv2AdminStatus MUST be 'disabled'.

If the network manager, or other means, sets this object to 'disabled', then the associated interface specific objects, i.e., the olsrv2InterfaceAdminStatus objects MUST all be 'disabled'.

This object is persistent and when written the entity SHOULD save the change to non-volatile storage."

DEFVAL { 2 }

::= { olsrv2ConfigurationGroup 1 }

olsrv2InterfaceTable OBJECT-TYPE

SYNTAX SEQUENCE OF Olsrv2InterfaceEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The olsrv2InterfaceTable describes the OLSRv2 status on the NHDP interfaces of this router. As such, this table augments the nhdpInterfaceTable

defined in the NHDP-MIB (RFC 6779). NHDP interfaces are explicitly defined by network management, CLI, or other means for interfaces on the device that are intended to run MANET protocols. The olsrv2InterfaceTable contains a single object, the olsrv2InterfaceAdminStatus object. This object is set by network management, or by other means, e.g., CLI.

A conceptual row in this table exists if and only if a corresponding entry in the nhdpInterfaceTable exists. If the corresponding entry with nhdpIfIndex value is deleted from the nhdpInterfaceTable, then the entry in this table is automatically deleted and OLSRv2 is disabled on this interface, and all configuration and state information related to this interface is to be removed from memory.

The olsrv2InterfaceAdminStatus can only be 'enabled' if the corresponding olsrv2AdminStatus object is also set to 'enabled'."

#### REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

::= { olsrv2ConfigurationGroup 2 }

#### olsrv2InterfaceEntry OBJECT-TYPE

SYNTAX Olsrv2InterfaceEntry

MAX-ACCESS not-accessible

STATUS current

#### DESCRIPTION

"The olsrv2InterfaceEntry describes one OLSRv2 local interface configuration as indexed by its nhdpIfIndex as defined in the NHDP-MIB (RFC 6779)."

The objects in this table are persistent and when written the device SHOULD save the change to non-volatile storage. For further information on the storage behavior for these objects, refer to the description for the nhdpIfRowStatus object in the NHDP-MIB (RFC6779)."

#### REFERENCE

"RFC 6779 - The Neighborhood Discovery Protocol MIB, Herberg, U., Cole, R.G. and I. Chakeres, October 2012"

```

    AUGMENTS { nhdpInterfaceEntry }
    ::= { olsrv2InterfaceTable 1 }

Olsrv2InterfaceEntry ::=
    SEQUENCE {
        olsrv2InterfaceAdminStatus
        Olsrv2StatusTC
    }

olsrv2InterfaceAdminStatus OBJECT-TYPE
    SYNTAX      Olsrv2StatusTC
    MAX-ACCESS  read-create
    STATUS      current
    DESCRIPTION
        "The OLSRv2 interface's administrative status.
        The value 'enabled(1)' denotes that the interface
        is permitted to participate in the OLSRv2 routing
        process. The value 'disabled(2)' denotes that
        the interface is not permitted to participate
        in the OLSRv2 routing process.

        The configuration objects for the OLSRv2 routing
        process, other than the administrative status objects,
        are common to all interfaces on this device.
        As such, the OLSRv2 configuration objects are globally
        defined for the device and are not contained within
        the olsrv2InterfaceTable."
    DEFVAL { 2 }
    ::= { olsrv2InterfaceEntry 1 }

olsrv2OrigIpAddrType OBJECT-TYPE
    SYNTAX      InetAddressType { ipv4(1) , ipv6(2) }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The type of the olsrv2OrigIpAddr, as defined
        in the InetAddress MIB module (RFC 4001).

        Only the values 'ipv4(1)' and
        'ipv6(2)' are supported."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
    ::= { olsrv2ConfigurationGroup 3 }

olsrv2OrigIpAddr OBJECT-TYPE
    SYNTAX      InetAddress (SIZE(4|16))

```

```
MAX-ACCESS    read-write
STATUS        current
DESCRIPTION
    "The router's originator address. An address that
    is unique (within the MANET) to this router.

    This object is persistent and when written
    the entity SHOULD save the change to
    non-volatile storage."
REFERENCE
    "RFC XXXX - The Optimized Link State Routing Protocol
    version 2, Clausen, T., Dearlove, C., Jacquet, P.
    and U. Herberg, March 2013."
 ::= { olsrv2ConfigurationGroup 4 }

--
-- Local History Times
--

olsrv2OHoldTime  OBJECT-TYPE
    SYNTAX      Unsigned32
    UNITS       "milliseconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "olsrv2OHoldTime corresponds to
        O_HOLD_TIME of OLSRv2 and represents the
        time for which a recently used and replaced
        originator address is used to recognize the router's
        own messages.

        Guidance for setting this object may be found
        in Section 5 of the OLSRv2 specification (RFC XXXX),
        which indicates that:
            o  olsrv2OHoldTime > 0

        This object is persistent and when written
        the entity SHOULD save the change to
        non-volatile storage."
    REFERENCE
        "Section 5 on Protocol Parameters.
        RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
    DEFVAL { 30000 }
 ::= { olsrv2ConfigurationGroup 5 }
```

```
--
-- Message intervals
--

olsrv2TcInterval OBJECT-TYPE
    SYNTAX      Olsrv2TimeValueCompressedForm32TC
    UNITS        "milliseconds"
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "olsrv2TcInterval corresponds to
        TC_INTERVAL of OLSRv2 and represents the
        maximum time between the transmission of
        two successive TC messages by this router.

        Guidance for setting this object may be found
        in Section 5 of the OLSRv2 specification (RFC XXXX),
        which indicates that:

            o olsrv2TcInterval > 0
            o olsrv2TcInterval >= olsrv2TcMinInterval

        This object is persistent and when written
        the entity SHOULD save the change to
        non-volatile storage."
    REFERENCE
        "Section 5 on Representing Time.
        RFC 5497 - Representing Multi-Value Time in
        Mobile Ad Hoc Networks (MANETs),
        Clausen, T. and C. Dearlove, March 2009.

        and

        Section 5 on Protocol Parameters.
        RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
    DEFVAL { 5000 }
 ::= { olsrv2ConfigurationGroup 6 }

olsrv2TcMinInterval OBJECT-TYPE
    SYNTAX      Olsrv2TimeValueCompressedForm32TC
    UNITS        "milliseconds"
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "olsrv2TcMinInterval corresponds to
        TC_MIN_INTERVAL of OLSRv2 and represents
```

the minimum interval between transmission of two successive TC messages by this router.

Guidance for setting this object may be found in Section 5 of the OLSRv2 specification (RFC XXXX), which indicates that:

```
o olsrv2TcInterval >= olsrv2TcMinInterval
```

The OLSRv2 protocol may choose to represent this time interval in terms of the 8-bit exponent-mantissa form defined in Section 5 of RFC 5497. When this is the case, this object value MUST be translated from the integer form represented in this MIB-module into the exponent-mantissa form for the OLSRv2 protocol to use according to the algorithm defined in Section 5 of RFC 5497 for finding the next larger time value within the exponent-mantissa format.

This object is persistent and when written the entity SHOULD save the change to non-volatile storage."

#### REFERENCE

"Section 5 on Representing Time.  
RFC 5497 - Representing Multi-Value Time in  
Mobile Ad Hoc Networks (MANETs),  
Clausen, T. and C. Dearlove, March 2009.

and

Section 5 on Protocol Parameters.  
RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

```
DEFVAL { 1250 }
 ::= { olsrv2ConfigurationGroup 7 }
```

```
--
-- Advertised information validity times
--
```

```
olsrv2THoldTime OBJECT-TYPE
    SYNTAX      Olsrv2TimeValueCompressedForm32TC
    UNITS       "milliseconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
```

"olsrv2THoldTime corresponds to T\_HOLD\_TIME of OLSRv2 and is used as the minimum value in the TLV with Type = VALIDITY\_TIME included in all TC messages sent by this router.

Guidance for setting this object may be found in Section 5 of the OLSRv2 specification (RFC XXXX), which indicates that:

- o olsrv2THoldTime >= olsrv2TcInterval
- o If TC messages can be lost, then olsrv2THoldTime SHOULD be significantly greater than olsrv2TcInterval; a value >= 3 x olsrv2TcInterval is RECOMMENDED.

This object is persistent and when written the entity SHOULD save the change to non-volatile storage."

#### REFERENCE

"Section 5 on Representing Time.  
RFC 5497 - Representing Multi-Value Time in Mobile Ad Hoc Networks (MANETs),  
Clausen, T. and C. Dearlove, March 2009.

and

Section 5 on Protocol Parameters.  
RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

DEFVAL { 15000 }

::= { olsrv2ConfigurationGroup 8 }

olsrv2AHoldTime OBJECT-TYPE

SYNTAX Olsrv2TimeValueCompressedForm32TC

UNITS "milliseconds"

MAX-ACCESS read-write

STATUS current

#### DESCRIPTION

"olsrv2AHoldTime corresponds to A\_HOLD\_TIME of OLSRv2 and represents the period during which TC messages are sent after they no longer have any advertised information to report, but are sent in order to accelerate outdated information removal by other routers.

Guidance for setting this object may be found

in Section 5 of the OLSRv2 specification (RFC XXXX), which indicates that:

- o If TC messages can be lost, then  
olsrv2AHoldTime SHOULD be  
significantly greater than olsrv2TcInterval;  
a value  $\geq 3 \times \text{olsrv2TcInterval}$  is  
RECOMMENDED.

This object is persistent and when written the entity SHOULD save the change to non-volatile storage."

REFERENCE

"Section 5 on Representing Time.  
RFC 5497 - Representing Multi-Value Time in  
Mobile Ad Hoc Networks (MANETs),  
Clausen, T. and C. Dearlove, March 2009.

and

Section 5 on Protocol Parameters.  
RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

DEFVAL { 15000 }  
::= { olsrv2ConfigurationGroup 9 }

--  
-- Received message validity times  
--

olsrv2RxHoldTime OBJECT-TYPE

SYNTAX Unsigned32  
UNITS "milliseconds"  
MAX-ACCESS read-write  
STATUS current  
DESCRIPTION

"olsrv2RxHoldTime corresponds to  
RX\_HOLD\_TIME of OLSRv2 and represents the period  
after receipt of a message by the appropriate OLSRv2  
interface of this router for which that information  
is recorded, in order that the message is recognized  
as having been previously received on this OLSRv2  
interface.

Guidance for setting this object may be found  
in Section 5 of the OLSRv2 specification (RFC XXXX),  
which indicates that:

- o olsrv2RxHoldTime > 0



- o This parameter SHOULD be greater than the maximum difference in time that a message may take to traverse the MANET, taking into account any message forwarding jitter as well as propagation, queuing, and processing delays.

This object is persistent and when written the entity SHOULD save the change to non-volatile storage."

REFERENCE

"Section 5 on Protocol Parameters.

RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

DEFVAL { 30000 }

::= { olsrv2ConfigurationGroup 10 }

olsrv2PHoldTime OBJECT-TYPE

SYNTAX Unsigned32

UNITS "milliseconds"

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"olsrv2PHoldTime corresponds to P\_HOLD\_TIME of OLSRv2 and represents the period after receipt of a message that is processed by this router for which that information is recorded, in order that the message is not processed again if received again.

Guidance for setting this object may be found in Section 5 of the OLSRv2 specification (RFC XXXX), which indicates that:

- o olsrv2PHoldTime > 0
- o This parameter SHOULD be greater than the maximum difference in time that a message may take to traverse the MANET, taking into account any message forwarding jitter as well as propagation, queuing, and processing delays.

This object is persistent and when written the entity SHOULD save the change to non-volatile storage."

REFERENCE

"Section 5 on Protocol Parameters.

RFC XXXX - The Optimized Link State Routing Protocol

```
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
    DEFVAL { 30000 }
 ::= { olsrv2ConfigurationGroup 11 }

olsrv2FHoldTime OBJECT-TYPE
    SYNTAX      Unsigned32
    UNITS       "milliseconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "olsrv2FHoldTime corresponds to
        F_HOLD_TIME of OLSRv2 and represents the period
        after receipt of a message that is forwarded by this
        router for which that information is recorded, in order
        that the message is not forwarded again if received again.

        Guidance for setting this object may be found
        in Section 5 of the OLSRv2 specification (RFC XXXX),
        which indicates that:
            o olsrv2FHoldTime > 0
            o This parameter SHOULD be greater
              than the maximum difference in time that a
              message may take to traverse the MANET,
              taking into account any message forwarding
              jitter as well as propagation, queuing,
              and processing delays.

        This parameter SHOULD be greater
        than the maximum difference in time that a
        message may take to traverse the MANET,
        taking into account any message forwarding
        jitter as well as propagation, queuing,
        and processing delays.

        This object is persistent and when written
        the entity SHOULD save the change to
        non-volatile storage."
    REFERENCE
        "Section 5 on Protocol Parameters.
        RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
    DEFVAL { 30000 }
 ::= { olsrv2ConfigurationGroup 12 }
```

--

```
-- Jitter times
--
```

```
olsrv2TpMaxJitter OBJECT-TYPE
```

```
SYNTAX      Unsigned32
UNITS       "milliseconds"
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION
```

```
"olsrv2TpMaxJitter corresponds to
TP_MAXJITTER of OLSRv2 and represents the value
of MAXJITTER used in RFC5148 for periodically
generated TC messages sent by this router.
```

```
For constraints on these parameters see RFC 5148.
```

```
This object is persistent and when written
the entity SHOULD save the change to
non-volatile storage."
```

```
REFERENCE
```

```
"Section 5 on Protocol Parameters.
RFC XXXX - The Optimized Link State Routing Protocol
version 2, Clausen, T., Dearlove, C., Jacquet, P.
and U. Herberg, March 2013."
```

```
DEFVAL { 500 }
```

```
::= { olsrv2ConfigurationGroup 13 }
```

```
olsrv2TtMaxJitter OBJECT-TYPE
```

```
SYNTAX      Unsigned32
UNITS       "milliseconds"
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION
```

```
"olsrv2TtMaxJitter corresponds to
TT_MAXJITTER of OLSRv2 and represents the value
of MAXJITTER used in RFC5148 for externally
triggered TC messages sent by this router.
```

```
For constraints on these parameters see RFC 5148.
```

```
This object is persistent and when written
the entity SHOULD save the change to
non-volatile storage."
```

```
REFERENCE
```

```
"Section 5 on Protocol Parameters.
RFC XXXX - The Optimized Link State Routing Protocol
version 2, Clausen, T., Dearlove, C., Jacquet, P.
and U. Herberg, March 2013."
```

```
    DEFVAL { 500 }  
 ::= { olsrv2ConfigurationGroup 14 }
```

olsrv2FMaxJitter OBJECT-TYPE

```
SYNTAX      Unsigned32  
UNITS       "milliseconds"  
MAX-ACCESS  read-write  
STATUS      current  
DESCRIPTION
```

"olsrv2FMaxJitter corresponds to  
F\_MAXJITTER of OLSRv2 and represents the  
default value of MAXJITTER used in RFC 5148 for  
messages forwarded by this router.

For constraints on these parameters see RFC 5148.

This object is persistent and when written  
the entity SHOULD save the change to  
non-volatile storage."

REFERENCE

"Section 5 on Protocol Parameters.  
RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

```
    DEFVAL { 500 }  
 ::= { olsrv2ConfigurationGroup 15 }
```

```
--  
-- Hop limits  
--
```

olsrv2TcHopLimit OBJECT-TYPE

```
SYNTAX      Unsigned32 (0..255)  
UNITS       "hops"  
MAX-ACCESS  read-write  
STATUS      current  
DESCRIPTION
```

"olsrv2TcHopLimit corresponds to  
TC\_HOP\_LIMIT of OLSRv2.

Guidance for setting this object may be found  
in Section 5 of the OLSRv2 specification (RFC XXXX),  
which indicates that:

- o The maximum value of  
olsrv2TcHopLimit >= the network diameter  
in hops, a value of 255 is RECOMMENDED.

```

    o All values of olsrv2TcHopLimit >= 2.

    This object is persistent and when written
    the entity SHOULD save the change to
    non-volatile storage."
REFERENCE
    "Section 5 on Protocol Parameters.
    RFC XXXX - The Optimized Link State Routing Protocol
    version 2, Clausen, T., Dearlove, C., Jacquet, P.
    and U. Herberg, March 2013."
    DEFVAL { 255 }
::= { olsrv2ConfigurationGroup 16 }

--
-- Willingness
--

olsrv2WillRouting OBJECT-TYPE
    SYNTAX      WillingnessTC
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "olsrv2WillRouting corresponds to
        WILL_ROUTING of OLSRv2.

        Guidance for setting this object may be found
        in Section 5 of the OLSRv2 specification (RFC XXXX),
        which indicates that:
            o WILL_NEVER (0) <= olsrv2WillRouting <=
              WILL_ALWAYS (15)

        This object is persistent and when written
        the entity SHOULD save the change to
        non-volatile storage."
REFERENCE
    "Section 5 on Protocol Parameters.
    RFC XXXX - The Optimized Link State Routing Protocol
    version 2, Clausen, T., Dearlove, C., Jacquet, P.
    and U. Herberg, March 2013."
    DEFVAL { 7 }
::= { olsrv2ConfigurationGroup 17 }

olsrv2WillFlooding OBJECT-TYPE
    SYNTAX      WillingnessTC
    MAX-ACCESS   read-write
    STATUS       current
```

## DESCRIPTION

"olsrv2WillFlooding corresponds to  
WILL\_FLOODING of OLSRv2.

Guidance for setting this object may be found  
in Section 5 of the OLSRv2 specification (RFC XXXX),  
which indicates that:

o WILL\_NEVER (0) <= olsrv2WillFlooding <=  
WILL\_ALWAYS (15)

This object is persistent and when written  
the entity SHOULD save the change to  
non-volatile storage."

## REFERENCE

"Section 5 on Protocol Parameters.  
RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

DEFVAL { 7 }

::= { olsrv2ConfigurationGroup 18 }

olsrv2LinkMetricType OBJECT-TYPE

SYNTAX IANAolsrv2LinkMetricTypeTC

MAX-ACCESS read-write

STATUS current

## DESCRIPTION

"olsrv2LinkMetricType corresponds to  
LINK\_METRIC\_TYPE of OLSRv2.

If olsrv2LinkMetricType changes, then all  
link metric information recorded by this router  
is invalid. The router MUST take the  
actions described in Section 5.5.  
'Parameter Change Constraints' and  
Section 17 'Information Base Changes'  
in RFC XXXX.

This object is persistent and when written  
the entity SHOULD save the change to  
non-volatile storage."

## REFERENCE

"Section 5 on Protocol Parameters.  
RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

DEFVAL { 0 }

::= { olsrv2ConfigurationGroup 19 }

```
--
-- olsrv2StateGroup
--

--
-- Contains information describing the current state of
-- the OLSRv2 process.
--

olsrv2StateGroup OBJECT IDENTIFIER ::= { olsrv2MIBObjects 2 }

--
-- Interface Information Base (IIB)
--

--
-- Link Set from RFC 6130, extended by L_in_metric,
-- L_out_metric, and L_mpr_selector entries for each tuple
--

olsrv2IibLinkSetTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Olsrv2IibLinkSetEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "A Link Set of an interface records all links
        from other routers which are, or recently
        were, 1-hop neighbors."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
    ::= { olsrv2StateGroup 1 }

olsrv2IibLinkSetEntry OBJECT-TYPE
    SYNTAX      Olsrv2IibLinkSetEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "A Link Set consists of Link Tuples, each
        representing a single link indexed by the
        local and remote interface pair. Each Link Set
        from NHDP is extended by OLSRv2 by the following
        fields:

        (L_in_metric (olsrv2IibLinkSetInMetricValue),
         L_out_metric (olsrv2IibLinkSetOutMetricValue),
         L_mpr_selector (olsrv2IibLinkSetMprSelector))"
```

```
REFERENCE
    "RFC XXXX - The Optimized Link State Routing Protocol
    version 2, Clausen, T., Dearlove, C., Jacquet, P.
    and U. Herberg, March 2013."
AUGMENTS { nhdpIibLinkSetEntry }
::= { olsrv2IibLinkSetTable 1 }

Olsrv2IibLinkSetEntry ::=
    SEQUENCE {
        olsrv2IibLinkSetInMetricValue
            Olsrv2MetricValueCompressedFormTC,
        olsrv2IibLinkSetOutMetricValue
            Olsrv2MetricValueCompressedFormTC,
        olsrv2IibLinkSetMprSelector
            TruthValue
    }

olsrv2IibLinkSetInMetricValue OBJECT-TYPE
    SYNTAX      Olsrv2MetricValueCompressedFormTC
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "olsrv2IibLinkSetInMetricValue is the metric of the link
        from the OLSRv2 interface with addresses
        L_neighbor_iface_addr_list to this OLSRv2 interface.
        The L_neighbor_iface_addr_list is identified by
        the nhdpDiscIfIndex which is an index to the
        nhdpIibLinkSetTable which this table augments."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
    ::= { olsrv2IibLinkSetEntry 1 }

olsrv2IibLinkSetOutMetricValue OBJECT-TYPE
    SYNTAX      Olsrv2MetricValueCompressedFormTC
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "olsrv2IibLinkSetOutMetricValue is the metric of the
        link to the OLSRv2 interface with addresses
        L_neighbor_iface_addr_list from this OLSRv2 interface.
        The L_neighbor_iface_addr_list is identified by
        the nhdpDiscIfIndex which is an index to the
        nhdpIibLinkSetTable which this table augments."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
```



```
        and U. Herberg, March 2013."
 ::= { olsrv2IibLinkSetEntry 2 }

olsrv2IibLinkSetMprSelector OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "olsrv2IibLinkSetMprSelector is a boolean flag,
        recording whether this neighbor has selected this router
        as a flooding MPR, i.e., is a flooding MPR selector
        of this router."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2IibLinkSetEntry 3 }

--
-- 2-Hop Set; from RFC 6130, extended by OLSRv2 by the
-- following fields: N2_in_metric, N2_out_metric
--

olsrv2Iib2HopSetTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Olsrv2Iib2HopSetEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "A 2-Hop Set of an interface records network
        addresses of symmetric 2-hop neighbors, and
        the symmetric links to symmetric 1-hop neighbors
        through which these symmetric 2-hop neighbors
        can be reached. It consists of 2-Hop Tuples."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2StateGroup 2 }

olsrv2Iib2HopSetEntry OBJECT-TYPE
    SYNTAX      Olsrv2Iib2HopSetEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "olsrv2Iib2HopSetTable consists of 2-Hop Tuples,
        each representing a single network address of
        a symmetric 2-hop neighbor, and a single MANET
        interface of a symmetric 1-hop neighbor."
```

Each 2-Hop Set from NHDP is extended by OLSRv2 by the following fields:

```
(N2_in_metric (olsrv2Iib2HopSetInMetricValue),
  N2_out_metric (olsrv2Iib2HopSetOutMetricValue))"
REFERENCE
  "RFC XXXX - The Optimized Link State Routing Protocol
  version 2, Clausen, T., Dearlove, C., Jacquet, P.
  and U. Herberg, March 2013."
AUGMENTS { nhdpIib2HopSetEntry }
::= { olsrv2Iib2HopSetTable 1 }

Olsrv2Iib2HopSetEntry ::=
SEQUENCE {
  olsrv2Iib2HopSetInMetricValue
    Olsrv2MetricValueCompressedFormTC,
  olsrv2Iib2HopSetOutMetricValue
    Olsrv2MetricValueCompressedFormTC
}

olsrv2Iib2HopSetInMetricValue OBJECT-TYPE
SYNTAX      Olsrv2MetricValueCompressedFormTC
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
  "olsrv2Iib2HopSetInMetricValue is the neighbor
  metric from the router with address
  N2_2hop_iface_addr to the router
  with OLSRv2 interface addresses
  N2_neighbor_iface_addr_list.

  The N2_2hop_iface_addr is identified by the
  (nhdpIib2HopSetIpAddressType,
  nhdpIib2HopSetIpAddress) pair from the
  nhdpIibLinkSetTable which this table augments.

  The N2_neighbor_iface_addr_list is defined by
  the nhdpDiscIfIndex which is an index of the
  nhdpIibLinkSetTable which this table augments."
REFERENCE
  "RFC XXXX - The Optimized Link State Routing Protocol
  version 2, Clausen, T., Dearlove, C., Jacquet, P.
  and U. Herberg, March 2013.

  and

  RFC 6779 - Definition of Managed Objects for the
  Neighborhood Discovery Process, Herberg, U.,
```

```

        Cole, R. and I. Chakeres, October 2012."
 ::= { olsrv2Iib2HopSetEntry 1 }

olsrv2Iib2HopSetOutMetricValue OBJECT-TYPE
    SYNTAX      Olsrv2MetricValueCompressedFormTC
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "olsrv2Iib2HopSetOutMetricValue is the neighbor metric
        to the router with address N2_2hop_iface_addr
        from the router with OLSRv2 interface addresses
        N2_neighbor_iface_addr_list.

        The N2_2hop_iface_addr is identified by the
        (nhdpIib2HopSetIpAddressType,
        nhdpIib2HopSetIpAddress) pair from the
        nhdpIibLinkSetTable which this table augments.

        The N2_neighbor_iface_addr_list is defined by
        the nhdpDiscIfIndex which is an index of the
        nhdpIibLinkSetTable which this table augments."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013.

        and

        RFC 6779 - Definition of Managed Objects for the
        Neighborhood Discovery Process, Herberg, U.,
        Cole, R. and I. Chakeres, October 2012."
 ::= { olsrv2Iib2HopSetEntry 2 }

--
-- Local Information Base - as defined in RFC 6130,
-- extended by the addition of an Originator Set,
-- defined in Section 6.1 and a Local Attached
-- Network Set, defined in Section 6.2.
--
--
-- Originator Set
--

olsrv2LibOrigSetTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Olsrv2LibOrigSetEntry
    MAX-ACCESS  not-accessible

```

```

STATUS      current
DESCRIPTION
    "A router's Originator Set records addresses
    that were recently used as originator addresses
    by this router."
REFERENCE
    "RFC XXXX - The Optimized Link State Routing Protocol
    version 2, Clausen, T., Dearlove, C., Jacquet, P.
    and U. Herberg, March 2013."
::= { olsrv2StateGroup 3 }

olsrv2LibOrigSetEntry OBJECT-TYPE
SYNTAX      Olsrv2LibOrigSetEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "A router's Originator Set consists of
    Originator Tuples:

    (O_orig_addr (olsrv2LibOrigSetIpAddressType
    and olsrv2LibOrigSetIpAddress),
    O_time (olsrv2LibOrigSetExpireTime))."
REFERENCE
    "RFC XXXX - The Optimized Link State Routing Protocol
    version 2, Clausen, T., Dearlove, C., Jacquet, P.
    and U. Herberg, March 2013."
INDEX { olsrv2LibOrigSetIpAddressType,
        olsrv2LibOrigSetIpAddress }
::= { olsrv2LibOrigSetTable 1 }

Olsrv2LibOrigSetEntry ::=
SEQUENCE {
    olsrv2LibOrigSetIpAddressType
        InetAddressType,
    olsrv2LibOrigSetIpAddress
        InetAddress,
    olsrv2LibOrigSetExpireTime
        TimeStamp
}

olsrv2LibOrigSetIpAddressType OBJECT-TYPE
SYNTAX      InetAddressType { ipv4(1) , ipv6(2) }
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The type of the olsrv2LibOrigSetIpAddress,
    as defined in the InetAddress MIB (RFC4001)."
```

Only the values 'ipv4(1)' and  
'ipv6(2)' are supported."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2LibOrigSetEntry 1 }

olsrv2LibOrigSetIpAddress OBJECT-TYPE

SYNTAX InetAddress (SIZE(4|16))

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"An originator address recently employed  
by this router."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2LibOrigSetEntry 2 }

olsrv2LibOrigSetExpireTime OBJECT-TYPE

SYNTAX TimeStamp

UNITS "centiseconds"

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"olsrv2LibOrigSetExpireTime specifies the value  
of sysUptime when this entry SHOULD expire and be  
removed from the olsrv2LibOrigSetTable. This time  
is determined at the time the entry is added,  
derived from the following expression:

O\_time := current time + O\_HOLD\_TIME

where O\_time is olsrv2LibOrigSetExpireTime,  
current\_time is current sysUptime and  
O\_HOLD\_TIME is a parameter of the OLSRv2  
protocol. In the event that the  
O\_HOLD\_TIME is changed, then the  
olsrv2LibOrigSetExpireTime needs to be  
recomputed for each of the entries in this Table."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2LibOrigSetEntry 3 }

```
--
-- Local Attached Network Set
--

olsrv2LibLocAttNetSetTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Olsrv2LibLocAttNetSetEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A router's Local Attached Network Set records
         its local non-OLSRv2 interfaces via which it
         can act as gateways to other networks."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
         version 2, Clausen, T., Dearlove, C., Jacquet, P.
         and U. Herberg, March 2013."
 ::= { olsrv2StateGroup 4 }

olsrv2LibLocAttNetSetEntry OBJECT-TYPE
    SYNTAX      Olsrv2LibLocAttNetSetEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The entries include the Local Attached
         Network Tuples:

        (AL_net_addr (olsrv2LibLocAttNetSetIpAddress),
         AL_dist (olsrv2LibLocAttNetSetDistance),
         AL_metric (olsrv2LibLocAttNetSetMetricValue)
         )

        where:

        AL_net_addr is the network address
        of an attached network which can
        be reached via this router. The
        AL_net_addr is defined in this MIB
        module by the tuple
        (olsrv2LibLocAttNetSetIpAddressType,
         olsrv2LibLocAttNetSetIpAddress,
         olsrv2LibLocAttNetSetIpAddressPrefixLen).

        AL_dist is the number of hops to
        the network with address AL_net_addr
        from this router. The AL_dist is
        defined in this MIB module by the
        olsrv2LibLocAttNetSetDistance object.
```

AL\_metric is the metric of the link to the attached network with address AL\_net\_addr from this router. The AL\_metric is defined in this MIB module by the olsrv2LibLocAttNetSetMetricValue object.

OLSRv2 (RFC XXXX) defines the rules for managing entries within this table, e.g., populating and purging entries. Specific instructions for the olsrv2LibLocAttNetSetEntry(s) are found in Section 7.2 and Section 17 of OLSRv2 (RFC XXXX)."

#### REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

INDEX { olsrv2LibLocAttNetSetIpAddressType,  
olsrv2LibLocAttNetSetIpAddress,  
olsrv2LibLocAttNetSetIpAddressPrefixLen }

::= { olsrv2LibLocAttNetSetTable 1 }

Olsrv2LibLocAttNetSetEntry ::=

```
SEQUENCE {
    olsrv2LibLocAttNetSetIpAddressType
        InetAddressType,
    olsrv2LibLocAttNetSetIpAddress
        InetAddress,
    olsrv2LibLocAttNetSetIpAddressPrefixLen
        InetAddressPrefixLength,
    olsrv2LibLocAttNetSetDistance
        Unsigned32,
    olsrv2LibLocAttNetSetMetricValue
        Olsrv2MetricValueCompressedFormTC
}
```

olsrv2LibLocAttNetSetIpAddressType OBJECT-TYPE

SYNTAX InetAddressType { ipv4(1) , ipv6(2) }

MAX-ACCESS not-accessible

STATUS current

#### DESCRIPTION

"The type of the olsrv2LibLocAttNetSetIpAddress, as defined in the InetAddress MIB (RFC 4001).

Only the values 'ipv4(1)' and 'ipv6(2)' are supported."

#### REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P.

```
        and U. Herberg, March 2013."
 ::= { olsrv2LibLocAttNetSetEntry 1 }

olsrv2LibLocAttNetSetIpAddress OBJECT-TYPE
    SYNTAX      InetAddress (SIZE(4|16))
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This is the network address of an attached
        network which can be reached via this router."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2LibLocAttNetSetEntry 2 }

olsrv2LibLocAttNetSetIpAddressPrefixLen OBJECT-TYPE
    SYNTAX      InetAddressPrefixLength
    UNITS        "bits"
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Indicates the number of leading one bits that form the
        mask to be logically ANDed with the destination address
        before being compared to the value in the
        olsrv2LibLocAttNetSetIpAddress field."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2LibLocAttNetSetEntry 3 }

olsrv2LibLocAttNetSetDistance OBJECT-TYPE
    SYNTAX      Unsigned32 (1..255)
    UNITS        "hops"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This object specifies the number of hops
        to the network with address
        olsrv2LibLocAttNetSetIpAddress from this router."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2LibLocAttNetSetEntry 4 }

olsrv2LibLocAttNetSetMetricValue OBJECT-TYPE
```



```

SYNTAX      Olsrv2MetricValueCompressedFormTC
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "This object specifies the metric of the
    link to the attached network with
    address AL_net_addr from this router.  The
    AL_net_addr is defined by the tuple
    (olsrv2LibLocAttNetSetIpAddressType,
     olsrv2LibLocAttNetSetIpAddress,
     olsrv2LibLocAttNetSetIpAddressPrefixLen)."
REFERENCE
    "RFC XXXX - The Optimized Link State Routing Protocol
    version 2, Clausen, T., Dearlove, C., Jacquet, P.
    and U. Herberg, March 2013."
 ::= { olsrv2LibLocAttNetSetEntry 5 }

--
-- Neighbor Information Base - as defined in RFC 6130,
-- extended by OLSRv2 by the addition of the following
-- elements to each Neighbor Tuple
--

--
-- Neighbor Set
--

olsrv2NibNeighborSetTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Olsrv2NibNeighborSetEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A router's Neighbor Set records all network
        addresses of each 1-hop neighbor. It consists
        of Neighbor Tuples, each representing a single
        1-hop neighbor. "
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
    ::= { olsrv2StateGroup 5 }

olsrv2NibNeighborSetEntry OBJECT-TYPE
    SYNTAX      Olsrv2NibNeighborSetEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION

```

"Each Neighbor Tuple in the Neighbor Set, defined in RFC 6130, has these additional elements:

```

    N_orig_addr (olsrv2NibNeighborSetNOrigIpAddrType,
                 olsrv2NibNeighborSetNOrigIpAddr)
    N_in_metric (olsrv2NibNeighborSetNInMetricValue)
    N_out_metric (olsrv2NibNeighborSetNOutMetricValue)
    N_will_flooding (olsrv2NibNeighborSetNWillFlooding)
    N_will_routing (olsrv2NibNeighborSetNWillRouting)
    N_flooding_mpr (olsrv2NibNeighborSetNFloodingMpr)
    N_routing_mpr (olsrv2NibNeighborSetNRoutingMpr)
    N_mpr_selector (olsrv2NibNeighborSetNMprSelector)
    N_advertised (olsrv2NibNeighborSetNAdvertised)
  
```

defined here as extensions."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

```

AUGMENTS { nhdpNibNeighborSetEntry }
::= { olsrv2NibNeighborSetTable 1 }
  
```

```
olsrv2NibNeighborSetEntry ::=
```

```

SEQUENCE {
    olsrv2NibNeighborSetNOrigIpAddrType
        InetAddressType,
    olsrv2NibNeighborSetNOrigIpAddr
        InetAddress,
    olsrv2NibNeighborSetNInMetricValue
        Olsrv2MetricValueCompressedFormTC,
    olsrv2NibNeighborSetNOutMetricValue
        Olsrv2MetricValueCompressedFormTC,
    olsrv2NibNeighborSetNWillFlooding
        WillingnessTC,
    olsrv2NibNeighborSetNWillRouting
        WillingnessTC,
    olsrv2NibNeighborSetNFloodingMpr
        TruthValue,
    olsrv2NibNeighborSetNRoutingMpr
        TruthValue,
    olsrv2NibNeighborSetNMprSelector
        TruthValue,
    olsrv2NibNeighborSetNAdvertised
        TruthValue
}
  
```

```

olsrv2NibNeighborSetNOrigIpAddrType  OBJECT-TYPE
SYNTAX      InetAddressType { ipv4(1) , ipv6(2) }
MAX-ACCESS  read-only
STATUS      current
  
```

## DESCRIPTION

"The type of the olsrv2NibNeighborSetNOrigIpAddress, as defined in the InetAddress MIB module (RFC4001).

Only the values 'ipv4(1)' and 'ipv6(2)' are supported."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

::= { olsrv2NibNeighborSetEntry 1 }

olsrv2NibNeighborSetNOrigIpAddress OBJECT-TYPE

SYNTAX InetAddress (SIZE(4|16))

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"This is the originator IP address of the neighbor represented by this table entry."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

::= { olsrv2NibNeighborSetEntry 2 }

olsrv2NibNeighborSetNInMetricValue OBJECT-TYPE

SYNTAX Olsrv2MetricValueCompressedFormTC

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"This object is the neighbor metric of any link from this neighbor to an OLSRv2 interface of this router, i.e., the minimum of all corresponding L\_in\_metric (olsrv2IibLinkSetInMetricValue) with L\_status = SYMMETRIC and L\_in\_metric (olsrv2IibLinkSetInMetricValue) != UNKNOWN\_METRIC, UNKNOWN\_METRIC if there are no such Link Tuples. UNKNOWN\_METRIC has a value of 0."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

::= { olsrv2NibNeighborSetEntry 3 }

olsrv2NibNeighborSetNOutMetricValue OBJECT-TYPE

SYNTAX Olsrv2MetricValueCompressedFormTC

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"This object is the neighbor metric of any link from an OLSRv2 interface of this router to this neighbor, i.e., the minimum of all corresponding L\_out\_metric (olsrv2IibLinkSetOutMetricValue) with L\_status = SYMMETRIC and L\_out\_metric (olsrv2IibLinkSetOutMetricValue) != UNKNOWN\_METRIC, UNKNOWN\_METRIC if there are no such Link Tuples. UNKNOWN\_METRIC has a value of 0."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

::= { olsrv2NibNeighborSetEntry 4 }

olsrv2NibNeighborSetNWillFlooding OBJECT-TYPE

SYNTAX WillingnessTC

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"This object is the neighbor's willingness to be selected as a flooding MPR, in the range from WILL\_NEVER to WILL\_ALWAYS, both inclusive, taking the value WILL\_NEVER if no OLSRv2 specific information is received from this neighbor."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

::= { olsrv2NibNeighborSetEntry 5 }

olsrv2NibNeighborSetNWillRouting OBJECT-TYPE

SYNTAX WillingnessTC

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"This object is the neighbor's willingness to be selected as a routing MPR, in the range from WILL\_NEVER to WILL\_ALWAYS, both inclusive, taking the value WILL\_NEVER if no OLSRv2 specific information is received from this neighbor."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

::= { olsrv2NibNeighborSetEntry 6 }

```
olsrv2NibNeighborSetNFFloodingMpr OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "This object is a boolean flag, recording whether
         this neighbor is selected as a flooding MPR
         by this router."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
         version 2, Clausen, T., Dearlove, C., Jacquet, P.
         and U. Herberg, March 2013."
 ::= { olsrv2NibNeighborSetEntry 7 }

olsrv2NibNeighborSetNRoutingMpr OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "This object is a boolean flag, recording whether
         this neighbor is selected as a routing MPR
         by this router."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
         version 2, Clausen, T., Dearlove, C., Jacquet, P.
         and U. Herberg, March 2013."
 ::= { olsrv2NibNeighborSetEntry 8 }

olsrv2NibNeighborSetNMprSelector OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "This object is a boolean flag,
         recording whether this neighbor has selected this router
         as a routing MPR, i.e., is a routing MPR
         selector of this router.

         When set to 'true', then this router is selected as
         a routing MPR by the neighbor router.
         When set to 'false',
         then this router is not selected by the neighbor
         as a routing MPR."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
         version 2, Clausen, T., Dearlove, C., Jacquet, P.
         and U. Herberg, March 2013."
 ::= { olsrv2NibNeighborSetEntry 9 }
```

```
olsrv2NibNeighborSetNAdvertised OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "This object, N_mpr_selector
        (olsrv2NibNeighborSetNMprSelector), is a boolean flag,
        recording whether this router has elected to
        advertise a link to this neighbor in its TC messages."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2NibNeighborSetEntry 10 }

olsrv2NibNeighborSetTableAnsn OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "Advertised Neighbor Sequence Number (ANSN), is
        a variable, whose value is included in TC messages to
        indicate the freshness of the information transmitted."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2StateGroup 6 }

--
-- Topology Information Base - this Information
-- Base is specific to OLSRv2, and is defined in
-- Section 10 of RFC XXXX.
--

--
-- Advertising Remote Router Set
--

olsrv2TibAdRemoteRouterSetTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Olsrv2TibAdRemoteRouterSetEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
        "A router's Advertising Remote Router Set records
        information describing each remote router in the
```

network that transmits TC messages."

REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

```
 ::= { olsrv2StateGroup 7 }
```

olsrv2TibAdRemoteRouterSetEntry OBJECT-TYPE

SYNTAX Olsrv2TibAdRemoteRouterSetEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A router's Advertised Neighbor Set Table entry consists of Advertising Remote Router Tuples:

```
(AR_orig_addr (olsrv2TibAdRemoteRouterSetIpAddressType,
                olsrv2TibAdRemoteRouterSetIpAddress),
  AR_seq_number (olsrv2TibAdRemoteRouterSetMaxSeqNo),
  AR_time (olsrv2TibAdRemoteRouterSetExpireTime).
```

Addresses associated with this router are found in the NHDP-MIB module's nhdpDiscIfSetTable.

OLSRv2 (RFC XXXX) defines the rules for managing entries within this table, e.g., populating and purging entries. Specific instructions for the olsrv2TibAdRemoteRouterSetEntry(s) are found in Section 10.1 and Section 17 of OLSRv2 (RFC XXXX)."

REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

INDEX { olsrv2TibAdRemoteRouterSetIpAddressType,
 olsrv2TibAdRemoteRouterSetIpAddress }

```
 ::= { olsrv2TibAdRemoteRouterSetTable 1 }
```

Olsrv2TibAdRemoteRouterSetEntry ::=

SEQUENCE {

```
  olsrv2TibAdRemoteRouterSetIpAddressType
    InetAddressType,
  olsrv2TibAdRemoteRouterSetIpAddress
    InetAddress,
  olsrv2TibAdRemoteRouterSetMaxSeqNo
    Unsigned32,
  olsrv2TibAdRemoteRouterSetExpireTime
    TimeStamp
}
```

```
olsrv2TibAdRemoteRouterSetIpAddressType OBJECT-TYPE
    SYNTAX      InetAddressType { ipv4(1) , ipv6(2) }
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The type of the olsrv2TibAdRemoteRouterSetIpAddress,
         as defined in the InetAddress MIB module (RFC4001).

         Only the values 'ipv4(1)' and
         'ipv6(2)' are supported."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
         version 2, Clausen, T., Dearlove, C., Jacquet, P.
         and U. Herberg, March 2013."
 ::= { olsrv2TibAdRemoteRouterSetEntry 1 }

olsrv2TibAdRemoteRouterSetIpAddress OBJECT-TYPE
    SYNTAX      InetAddress (SIZE(4|16))
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This is the originator address of a received
         TC message."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
         version 2, Clausen, T., Dearlove, C., Jacquet, P.
         and U. Herberg, March 2013."
 ::= { olsrv2TibAdRemoteRouterSetEntry 2 }

olsrv2TibAdRemoteRouterSetMaxSeqNo OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This is the greatest Assigned Neighbor Sequence
         Number (ANSN) in any TC message
         received which originated from the router
         with originator address
         olsrv2TibAdRemoteRouterSetIpAddress.

         Sequence numbers are used in the OLSRv2 protocol
         for the purpose of discarding 'old' information,
         i.e., messages received out of order. 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 this protocol,
```



OLSRv2 implementations observe the following when determining the ordering of sequence numbers.

In OLSRv2, MAXVALUE designates one more than the largest possible value for a sequence number. For a 16 bit sequence number MAXVALUE is 65536.

The sequence number S1 is said to be 'greater than' the sequence number S2 if:

- o S1 > S2 AND S1 - S2 < MAXVALUE/2 OR
- o S2 > S1 AND S2 - S1 > MAXVALUE/2

When sequence numbers S1 and S2 differ by MAXVALUE/2 their ordering cannot be determined. In this case, which should not occur, either ordering may be assumed.

Thus when comparing two messages, it is possible - even in the presence of wrap-around - to determine which message contains the most recent information."

#### REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

```
::= { olsrv2TibAdRemoteRouterSetEntry 3 }
```

```
olsrv2TibAdRemoteRouterSetExpireTime OBJECT-TYPE
```

```
SYNTAX      TimeStamp
UNITS       "centiseconds"
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
```

"olsrv2TibAdRemoteRouterSetExpireTime specifies the value of sysUptime when this entry SHOULD expire and be removed from the olsrv2TibAdRemoteRouterSetTable."

#### REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

```
::= { olsrv2TibAdRemoteRouterSetEntry 4 }
```

```
--
-- Router Topology Set
--
```

```

olsrv2TibRouterTopologySetTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Olsrv2TibTopologySetEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "A router's Router Topology Set records topology
        information about the network."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2StateGroup 8 }

olsrv2TibRouterTopologySetEntry OBJECT-TYPE
    SYNTAX      Olsrv2TibTopologySetEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "It consists of Router Topology Tuples:

        (TR_from_orig_addr
         (olsrv2TibRouterTopologySetFromOrigIpAddrType,
          olsrv2TibRouterTopologySetFromOrigIpAddr),
         TR_to_orig_addr
         (olsrv2TibRouterTopologySetToOrigIpAddrType,
          olsrv2TibRouterTopologySetToOrigIpAddr),
         TR_seq_number (olsrv2TibRouterTopologySetSeqNo),
         TR_metric (olsrv2TibRouterTopologySetMetricValue),
         TR_time (olsrv2TibRouterTopologySetExpireTime)).

        OLSRv2 (RFC XXXX) defines the rules for managing
        entries within this table, e.g., populating
        and purging entries.  Specific instructions for the
        olsrv2TibRouterTopologySetEntry(s) are found in
        Section 10.2 and Section 17 of OLSRv2 (RFC XXXX)."
```

REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

```

INDEX { olsrv2TibRouterTopologySetFromOrigIpAddrType,
        olsrv2TibRouterTopologySetFromOrigIpAddr,
        olsrv2TibRouterTopologySetToOrigIpAddrType,
        olsrv2TibRouterTopologySetToOrigIpAddr }
 ::= { olsrv2TibRouterTopologySetTable 1 }

Olsrv2TibTopologySetEntry ::=
    SEQUENCE {
        olsrv2TibRouterTopologySetFromOrigIpAddrType
```

```

        InetAddressType,
    olsrv2TibRouterTopologySetFromOrigIpAddress
        InetAddress,
    olsrv2TibRouterTopologySetToOrigIpAddressType
        InetAddressType,
    olsrv2TibRouterTopologySetToOrigIpAddress
        InetAddress,
    olsrv2TibRouterTopologySetSeqNo
        Unsigned32,
    olsrv2TibRouterTopologySetMetricValue
        Olsrv2MetricValueCompressedFormTC,
    olsrv2TibRouterTopologySetExpireTime
        TimeStamp
}

olsrv2TibRouterTopologySetFromOrigIpAddressType OBJECT-TYPE
    SYNTAX      InetAddressType { ipv4(1) , ipv6(2) }
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The type of the olsrv2TibRouterTopologySetFromOrigIpAddress,
        as defined in the InetAddress MIB module (RFC4001).

        Only the values 'ipv4(1)' and
        'ipv6(2)' are supported."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
    ::= { olsrv2TibRouterTopologySetEntry 1 }

olsrv2TibRouterTopologySetFromOrigIpAddress OBJECT-TYPE
    SYNTAX      InetAddress (SIZE(4|16))
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This is the originator address of a router which can
        reach the router with originator address TR_to_orig_addr
        in one hop."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
    ::= { olsrv2TibRouterTopologySetEntry 2 }

olsrv2TibRouterTopologySetToOrigIpAddressType OBJECT-TYPE
    SYNTAX      InetAddressType { ipv4(1) , ipv6(2) }
    MAX-ACCESS  not-accessible

```

```

STATUS      current
DESCRIPTION
    "The type of the olsrv2TibRouterTopologySetToOrigIpAddress,
    as defined in the InetAddress MIB module (RFC4001).

    Only the values 'ipv4(1)' and
    'ipv6(2)' are supported."
REFERENCE
    "RFC XXXX - The Optimized Link State Routing Protocol
    version 2, Clausen, T., Dearlove, C., Jacquet, P.
    and U. Herberg, March 2013."
::= { olsrv2TibRouterTopologySetEntry 3 }

olsrv2TibRouterTopologySetToOrigIpAddress OBJECT-TYPE
SYNTAX      InetAddress (SIZE(4|16))
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "This is the originator address of a router which can be
    reached by the router with originator address
    TR_to_orig_addr in one hop."
REFERENCE
    "RFC XXXX - The Optimized Link State Routing Protocol
    version 2, Clausen, T., Dearlove, C., Jacquet, P.
    and U. Herberg, March 2013."
::= { olsrv2TibRouterTopologySetEntry 4 }

olsrv2TibRouterTopologySetSeqNo OBJECT-TYPE
SYNTAX      Unsigned32 (0..65535)
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "This is the greatest Assigned Neighbor Sequence
    Number (ANSN) in any TC message
    received which originated from the router
    with originator address TR_from_orig_addr,
    i.e., which contributed to the information
    contained in this Tuple and is defined by the
    objects:
    (olsrv2TibRouterTopologySetFromOrigIpAddressType,
    olsrv2TibRouterTopologySetFromOrigIpAddress).

    Sequence numbers are used in the OLSRv2 protocol
    for the purpose of discarding 'old' information,
    i.e., messages received out of order. 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 this protocol, OLSRv2 implementations observe the following when determining the ordering of sequence numbers.

In OLSRv2, MAXVALUE designates one more than the largest possible value for a sequence number. For a 16 bit sequence number MAXVALUE is 65536.

The sequence number S1 is said to be 'greater than' the sequence number S2 if:

- o S1 > S2 AND S1 - S2 < MAXVALUE/2 OR
- o S2 > S1 AND S2 - S1 < MAXVALUE/2

When sequence numbers S1 and S2 differ by MAXVALUE/2 their ordering cannot be determined. In this case, which should not occur, either ordering may be assumed.

Thus when comparing two messages, it is possible - even in the presence of wrap-around - to determine which message contains the most recent information."

#### REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

```
::= { olsrv2TibRouterTopologySetEntry 5 }
```

olsrv2TibRouterTopologySetMetricValue OBJECT-TYPE

SYNTAX Olsrv2MetricValueCompressedFormTC

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This is the neighbor metric from the router with originator address TR\_from\_orig\_addr (olsrv2TibRouterTopologySetFromOrigIpAddressType, olsrv2TibRouterTopologySetFromOrigIpAddress) to the router with originator address TR\_to\_orig\_addr (olsrv2TibRouterTopologySetToOrigIpAddressType, olsrv2TibRouterTopologySetToOrigIpAddress)."

#### REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

```
::= { olsrv2TibRouterTopologySetEntry 6 }
```

```

olsrv2TibRouterTopologySetExpireTime OBJECT-TYPE
    SYNTAX      TimeStamp
    UNITS        "centiseconds"
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
        "olsrv2TibRouterTopologySetExpireTime specifies the value
         of sysUptime when this entry SHOULD expire and be
         removed from the olsrv2TibRouterTopologySetTable."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
         version 2, Clausen, T., Dearlove, C., Jacquet, P.
         and U. Herberg, March 2013."
 ::= { olsrv2TibRouterTopologySetEntry 7 }

--
-- Routable Address Topology Set
--

olsrv2TibRoutableAddressTopologySetTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Olsrv2TibRoutableAddressTopologySetEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
        "A router's Routable Address Topology Set records topology
         information about the routable addresses within the MANET,
         and via which routers they may be reached."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
         version 2, Clausen, T., Dearlove, C., Jacquet, P.
         and U. Herberg, March 2013."
 ::= { olsrv2StateGroup 9 }

olsrv2TibRoutableAddressTopologySetEntry OBJECT-TYPE
    SYNTAX      Olsrv2TibRoutableAddressTopologySetEntry
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
        "It consists of Router Topology Tuples:

        (TA_from_orig_addr
         (olsrv2TibRoutableAddressTopologySetFromOrigIpAddressType
          olsrv2TibRoutableAddressTopologySetFromOrigIpAddress),
         TA_dest_addr
         (olsrv2TibRoutableAddressTopologySetFromDestIpAddressType
          olsrv2TibRoutableAddressTopologySetFromDestIpAddress),
         TA_seq_number (olsrv2TibRoutableAddressTopologySetSeqNo)

```

```

    TA_metric (olsrv2TibRoutableAddressTopologySetMetricValue)
    TA_time (olsrv2TibRoutableAddressTopologySetExpireTime)
)

```

OLSRv2 (RFC XXXX) defines the rules for managing entries within this table, e.g., populating and purging entries. Specific instructions for the olsrv2TibRoutableAddressTopologySetEntry(s) are found in Section 10.3 and Section 17 of OLSRv2 (RFC XXXX)."

#### REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

```

INDEX { olsrv2TibRoutableAddressTopologySetFromOrigIpAddrType,
        olsrv2TibRoutableAddressTopologySetFromOrigIpAddr,
        olsrv2TibRoutableAddressTopologySetDestIpAddrType,
        olsrv2TibRoutableAddressTopologySetDestIpAddr }
 ::= { olsrv2TibRoutableAddressTopologySetTable 1 }

```

```

Olsrv2TibRoutableAddressTopologySetEntry ::=
SEQUENCE {
    olsrv2TibRoutableAddressTopologySetFromOrigIpAddrType
        InetAddressType,
    olsrv2TibRoutableAddressTopologySetFromOrigIpAddr
        InetAddress,
    olsrv2TibRoutableAddressTopologySetDestIpAddrType
        InetAddressType,
    olsrv2TibRoutableAddressTopologySetDestIpAddr
        InetAddress,
    olsrv2TibRoutableAddressTopologySetSeqNo
        Unsigned32,
    olsrv2TibRoutableAddressTopologySetMetricValue
        Olsrv2MetricValueCompressedFormTC,
    olsrv2TibRoutableAddressTopologySetExpireTime
        TimeStamp
}

```

olsrv2TibRoutableAddressTopologySetFromOrigIpAddrType OBJECT-TYPE

SYNTAX InetAddressType { ipv4(1) , ipv6(2) }

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The type of the  
olsrv2TibRoutableAddressTopologySetFromOrigIpAddr,  
as defined in the InetAddress MIB module (RFC 4001).

Only the values 'ipv4(1)' and  
'ipv6(2)' are supported."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2TibRoutableAddressTopologySetEntry 1 }

olsrv2TibRoutableAddressTopologySetFromOrigIpAddr OBJECT-TYPE

SYNTAX InetAddress (SIZE(4|16))

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"This is the originator address of a router which can  
reach the router with routable address TA\_dest\_addr  
in one hop."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2TibRoutableAddressTopologySetEntry 2 }

olsrv2TibRoutableAddressTopologySetDestIpAddrType OBJECT-TYPE

SYNTAX InetAddressType { ipv4(1) , ipv6(2) }

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"The type of the olsrv2TibRouterTopologySetToOrigIpAddr,  
as defined in the InetAddress MIB module (RFC 4001).

Only the values 'ipv4(1)' and  
'ipv6(2)' are supported."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2TibRoutableAddressTopologySetEntry 3 }

olsrv2TibRoutableAddressTopologySetDestIpAddr OBJECT-TYPE

SYNTAX InetAddress (SIZE(4|16))

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"This is a routable address of a router which can be  
reached by the router with originator address  
TA\_from\_orig\_addr in one hop. The TA\_from\_orig\_addr  
is defined by the tuple  
(olsrv2TibRoutableAddressTopologySetFromOrigIpAddrType  
olsrv2TibRoutableAddressTopologySetFromOrigIpAddr)."

## REFERENCE



```

    "RFC XXXX - The Optimized Link State Routing Protocol
    version 2, Clausen, T., Dearlove, C., Jacquet, P.
    and U. Herberg, March 2013."
 ::= { olsrv2TibRoutableAddressTopologySetEntry 4 }

olsrv2TibRoutableAddressTopologySetSeqNo OBJECT-TYPE
    SYNTAX      Unsigned32 (0..65535)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This is the greatest ANSN in any TC message
        received which originated from the router
        with originator address TA_from_orig_addr,
        i.e., which contributed to the information
        contained in this Tuple.  The TA_from_orig_addr
        is defined by the tuple
        (olsrv2TibRoutableAddressTopologySetFromOrigIpAddressType
         olsrv2TibRoutableAddressTopologySetFromOrigIpAddress)."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2TibRoutableAddressTopologySetEntry 5 }

olsrv2TibRoutableAddressTopologySetMetricValue OBJECT-TYPE
    SYNTAX      Olsrv2MetricValueCompressedFormTC
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This is the neighbor metric from the router
        with originator address TA_from_orig_addr (defined
        by the tuple
        (olsrv2TibRoutableAddressTopologySetFromOrigIpAddressType
         olsrv2TibRoutableAddressTopologySetFromOrigIpAddress))
        to the router with OLSRv2 interface address TA_dest_addr
        (defined by the tuple
        (olsrv2TibRoutableAddressTopologySetFromDestIpAddressType
         olsrv2TibRoutableAddressTopologySetFromDestIpAddress))."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2TibRoutableAddressTopologySetEntry 6 }

olsrv2TibRoutableAddressTopologySetExpireTime OBJECT-TYPE
    SYNTAX      TimeStamp
    UNITS       "centiseconds"
    MAX-ACCESS  read-only
```

```

STATUS      current
DESCRIPTION
    "olsrv2TibRoutableAddressTopologySetExpireTime
    specifies the value of sysUptime when this entry
    SHOULD expire and be removed from the
    olsrv2TibRoutableAddressTopologySetTable."
REFERENCE
    "RFC XXXX - The Optimized Link State Routing Protocol
    version 2, Clausen, T., Dearlove, C., Jacquet, P.
    and U. Herberg, March 2013."
 ::= { olsrv2TibRoutableAddressTopologySetEntry 7 }

--
-- Attached Network Set
--

olsrv2TibAttNetworksSetTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Olsrv2TibAttNetworksSetEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "A router's Attached Network Set records information
        about networks (which may be outside the MANET)
        attached to other routers and their routable addresses."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2StateGroup 10 }

olsrv2TibAttNetworksSetEntry OBJECT-TYPE
    SYNTAX      Olsrv2TibAttNetworksSetEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "It consists of Attached Network Tuples:

        (AN_orig_addr
         (olsrv2TibAttNetworksSetOrigIpAddressType,
          olsrv2TibAttNetworksSetOrigIpAddress),
         AN_net_addr
         (olsrv2TibAttNetworksSetNetIpAddressType,
          olsrv2TibAttNetworksSetNetIpAddress,
          olsrv2TibAttNetworksSetNetIpAddressPrefixLen),
         AN_seq_number (olsrv2TibAttNetworksSetSeqNo),
         AN_dist (olsrv2TibAttNetworksSetDist),
         AN_metric (olsrv2TibAttNetworksSetMetricValue),

```

```

    AN_time (olsrv2TibAttNetworksSetExpireTime)
)

OLSRv2 (RFC XXXX) defines the rules for managing
entries within this table, e.g., populating
and purging entries.  Specific instructions for the
olsrv2TibRoutableAddressTopologySetEntry(s) are found
in Section 10.4 and Section 17 of OLSRv2 (RFC XXXX)."
```

REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

```

INDEX { olsrv2TibAttNetworksSetOrigIpAddrType,
        olsrv2TibAttNetworksSetOrigIpAddr,
        olsrv2TibAttNetworksSetNetIpAddrType,
        olsrv2TibAttNetworksSetNetIpAddr,
        olsrv2TibAttNetworksSetNetIpAddrPrefixLen }
 ::= { olsrv2TibAttNetworksSetTable 1 }
```

Olsrv2TibAttNetworksSetEntry ::=

```

SEQUENCE {
    olsrv2TibAttNetworksSetOrigIpAddrType
        InetAddressType,
    olsrv2TibAttNetworksSetOrigIpAddr
        InetAddress,
    olsrv2TibAttNetworksSetNetIpAddrType
        InetAddressType,
    olsrv2TibAttNetworksSetNetIpAddr
        InetAddress,
    olsrv2TibAttNetworksSetNetIpAddrPrefixLen
        InetAddressPrefixLength,
    olsrv2TibAttNetworksSetSeqNo
        Unsigned32,
    olsrv2TibAttNetworksSetDist
        Unsigned32,
    olsrv2TibAttNetworksSetMetricValue
        Olsrv2MetricValueCompressedFormTC,
    olsrv2TibAttNetworksSetExpireTime
        TimeStamp
}
```

olsrv2TibAttNetworksSetOrigIpAddrType OBJECT-TYPE

```

SYNTAX      InetAddressType { ipv4(1) , ipv6(2) }
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "The type of the olsrv2TibAttNetworksSetOrigIpAddr,
    as defined in the InetAddress MIB module (RFC4001)."
```

Only the values 'ipv4(1)' and  
'ipv6(2)' are supported."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2TibAttNetworksSetEntry 1 }

olsrv2TibAttNetworksSetOrigIpAddress OBJECT-TYPE

SYNTAX InetAddress (SIZE(4|16))

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"This is the originator address, of type  
olsrv2TibAttNetworksSetOrigIpAddressType, of a  
router which can act as gateway to the  
network with address AN\_net\_addr. The  
AN\_net\_addr is defined by the tuple  
(olsrv2TibAttNetworksSetNetIpAddressType,  
olsrv2TibAttNetworksSetNetIpAddress,  
olsrv2TibAttNetworksSetNetIpAddressPrefixLen)."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2TibAttNetworksSetEntry 2 }

olsrv2TibAttNetworksSetNetIpAddressType OBJECT-TYPE

SYNTAX InetAddressType { ipv4(1) , ipv6(2) }

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"The type of the olsrv2TibAttNetworksSetNetIpAddress,  
as defined in the InetAddress MIB module (RFC 4001).

Only the values 'ipv4(1)' and  
'ipv6(2)' are supported."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2TibAttNetworksSetEntry 3 }

olsrv2TibAttNetworksSetNetIpAddress OBJECT-TYPE

SYNTAX InetAddress (SIZE(4|16))

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"This is is the network address, of type  
 olsrv2TibAttNetworksSetNetIpAddressType, of an  
 attached network, which may be reached via  
 the router with originator address AN\_orig\_addr.  
 The AN\_orig\_addr is defined by the tuple  
 (olsrv2TibAttNetworksSetOrigIpAddressType,  
 olsrv2TibAttNetworksSetOrigIpAddress)."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
 version 2, Clausen, T., Dearlove, C., Jacquet, P.  
 and U. Herberg, March 2013."

::= { olsrv2TibAttNetworksSetEntry 4 }

olsrv2TibAttNetworksSetNetIpAddressPrefixLen OBJECT-TYPE

SYNTAX InetAddressPrefixLength

UNITS "bits"

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"Indicates the number of leading one bits that form the  
 mask to be logically ANDed with the destination address  
 before being compared to the value in the  
 olsrv2TibAttNetworksSetNetIpAddress field."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
 version 2, Clausen, T., Dearlove, C., Jacquet, P.  
 and U. Herberg, March 2013."

::= { olsrv2TibAttNetworksSetEntry 5 }

olsrv2TibAttNetworksSetSeqNo OBJECT-TYPE

SYNTAX Unsigned32 (0..65535)

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"This is the greatest Assigned Neighbor Sequence  
 Number (ANSN) in any TC message received  
 which originated from the router  
 with originator address AN\_orig\_addr  
 (i.e., which contributed to the information  
 contained in this Tuple). The AN\_orig\_addr  
 is defined by the tuple  
 (olsrv2TibAttNetworksSetOrigIpAddressType,  
 olsrv2TibAttNetworksSetOrigIpAddress)."

Sequence numbers are used in the OLSRv2 protocol  
 for the purpose of discarding 'old' information,  
 i.e., messages received out of order. 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 this protocol, the following MUST be observed when determining the ordering of sequence numbers.

The term MAXVALUE designates in the following one more than the largest possible value for a sequence number. For a 16 bit sequence number (as are those defined in this specification) MAXVALUE is 65536.

The sequence number S1 is said to be 'greater than' the sequence number S2 if:

- o S1 > S2 AND S1 - S2 < MAXVALUE/2 OR
- o S2 > S1 AND S2 - S1 > MAXVALUE/2

When sequence numbers S1 and S2 differ by MAXVALUE/2 their ordering cannot be determined. In this case, which should not occur, either ordering may be assumed.

Thus when comparing two messages, it is possible - even in the presence of wrap-around - to determine which message contains the most recent information."

#### REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

```
::= { olsrv2TibAttNetworksSetEntry 6 }
```

olsrv2TibAttNetworksSetDist OBJECT-TYPE

SYNTAX Unsigned32 (0..255)

UNITS "hops"

MAX-ACCESS read-only

STATUS current

#### DESCRIPTION

"The number of hops to the network with address AN\_net\_addr from the router with originator address AN\_orig\_addr.

The AN\_orig\_addr is defined by the tuple (olsrv2TibAttNetworksSetOrigIpAddressType, olsrv2TibAttNetworksSetOrigIpAddress).

#### REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol

```
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2TibAttNetworksSetEntry 7 }

olsrv2TibAttNetworksSetMetricValue OBJECT-TYPE
    SYNTAX      Olsrv2MetricValueCompressedFormTC
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The metric of the link from the router with
        originator address AN_orig_addr to the attached
        network with address AN_net_addr.
        The AN_net_addr is defined by the tuple
        (olsrv2TibAttNetworksSetNetIpAddressType,
         olsrv2TibAttNetworksSetNetIpAddress,
         olsrv2TibAttNetworksSetNetIpAddressPrefixLen)."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2TibAttNetworksSetEntry 9 }

olsrv2TibAttNetworksSetExpireTime OBJECT-TYPE
    SYNTAX      TimeStamp
    UNITS       "centiseconds"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "olsrv2TibAttNetworksSetExpireTime
        specifies the value of sysUptime when this
        entry SHOULD expire and be removed from the
        olsrv2TibAttNetworksSetTable."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2TibAttNetworksSetEntry 10 }
```

```
--
-- Routing Set
--
```

```
olsrv2TibRoutingSetTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Olsrv2TibRoutingSetEntry
    MAX-ACCESS  not-accessible
    STATUS      current
```

## DESCRIPTION

"A router's Routing Set records the first hop along a selected path to each destination for which any such path is known."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

::= { olsrv2StateGroup 11 }

olsrv2TibRoutingSetEntry OBJECT-TYPE

SYNTAX Olsrv2TibRoutingSetEntry

MAX-ACCESS not-accessible

STATUS current

## DESCRIPTION

"It consists of Routing Tuples:

(R\_dest\_addr, R\_next\_iface\_addr,  
R\_local\_iface\_addr, R\_dist, R\_metric)"

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol version 2, Clausen, T., Dearlove, C., Jacquet, P. and U. Herberg, March 2013."

INDEX { olsrv2TibRoutingSetDestIpAddressType,  
olsrv2TibRoutingSetDestIpAddress,  
olsrv2TibRoutingSetDestIpAddressPrefixLen }

::= { olsrv2TibRoutingSetTable 1 }

Olsrv2TibRoutingSetEntry ::=

SEQUENCE {

olsrv2TibRoutingSetDestIpAddressType

InetAddressType,

olsrv2TibRoutingSetDestIpAddress

InetAddress,

olsrv2TibRoutingSetDestIpAddressPrefixLen

InetAddressPrefixLength,

olsrv2TibRoutingSetNextIfIpAddressType

InetAddressType,

olsrv2TibRoutingSetNextIfIpAddress

InetAddress,

olsrv2TibRoutingSetLocalIfIpAddressType

InetAddressType,

olsrv2TibRoutingSetLocalIfIpAddress

InetAddress,

olsrv2TibRoutingSetDist

Unsigned32,

olsrv2TibRoutingSetMetricValue

Olsrv2MetricValueCompressedFormTC



```
    }

    olsrv2TibRoutingSetDestIpAddressType OBJECT-TYPE
        SYNTAX      InetAddressType { ipv4(1) , ipv6(2) }
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
            "The type of the olsrv2TibRoutingSetDestIpAddress,
             as defined in the InetAddress MIB module (RFC 4001).

             Only the values 'ipv4(1)' and 'ipv6(2)' are
             supported."
        REFERENCE
            "RFC XXXX - The Optimized Link State Routing Protocol
             version 2, Clausen, T., Dearlove, C., Jacquet, P.
             and U. Herberg, March 2013."
    ::= { olsrv2TibRoutingSetEntry 1 }

    olsrv2TibRoutingSetDestIpAddress OBJECT-TYPE
        SYNTAX      InetAddress (SIZE(4|16))
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
            "This is the address of the destination,
             either the address of an interface of
             a destination router, or the network
             address of an attached network."
        REFERENCE
            "RFC XXXX - The Optimized Link State Routing Protocol
             version 2, Clausen, T., Dearlove, C., Jacquet, P.
             and U. Herberg, March 2013."
    ::= { olsrv2TibRoutingSetEntry 2 }

    olsrv2TibRoutingSetDestIpAddressPrefixLen OBJECT-TYPE
        SYNTAX      InetAddressPrefixLength
        UNITS        "bits"
        MAX-ACCESS   not-accessible
        STATUS       current
        DESCRIPTION
            "Indicates the number of leading one bits that form the
             mask to be logically ANDed with the destination address
             before being compared to the value in the
             olsrv2TibRoutingSetDestIpAddress field.

             Note: This definition needs to be consistent
             with the current forwarding table MIB module description.
             Specifically, it SHOULD allow for longest prefix
             matching of network addresses."
```

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2TibRoutingSetEntry 3 }

olsrv2TibRoutingSetNextIfIpAddressType OBJECT-TYPE

SYNTAX InetAddressType { ipv4(1) , ipv6(2) }

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The type of the olsrv2TibRoutingSetNextIfIpAddress,  
as defined in the InetAddress MIB module (RFC 4001).

Only the values 'ipv4(1)' and  
'ipv6(2)' are supported."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2TibRoutingSetEntry 4 }

olsrv2TibRoutingSetNextIfIpAddress OBJECT-TYPE

SYNTAX InetAddress (SIZE(4|16))

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"This object is the OLSRv2 interface address of the  
next hop on the selected path to the  
destination."

## REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2TibRoutingSetEntry 5 }

olsrv2TibRoutingSetLocalIfIpAddressType OBJECT-TYPE

SYNTAX InetAddressType { ipv4(1) , ipv6(2) }

MAX-ACCESS read-only

STATUS current

## DESCRIPTION

"The type of the olsrv2TibRoutingSetLocalIfIpAddress  
and olsrv2TibRoutingSetNextIfIpAddress,  
as defined in the InetAddress MIB module (RFC 4001).

Only the values 'ipv4(1)' and  
'ipv6(2)' are supported."

## REFERENCE

```
    "RFC XXXX - The Optimized Link State Routing Protocol
    version 2, Clausen, T., Dearlove, C., Jacquet, P.
    and U. Herberg, March 2013."
 ::= { olsrv2TibRoutingSetEntry 6 }

olsrv2TibRoutingSetLocalIfIpAddress OBJECT-TYPE
    SYNTAX      InetAddress (SIZE(4|16))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This object is the address of the local OLSRv2
        interface over which a packet must be
        sent to reach the destination by the
        selected path."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2TibRoutingSetEntry 7 }

olsrv2TibRoutingSetDist OBJECT-TYPE
    SYNTAX      Unsigned32 (0..255)
    UNITS       "hops"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This object is the number of hops on the selected
        path to the destination."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2TibRoutingSetEntry 8 }

olsrv2TibRoutingSetMetricValue OBJECT-TYPE
    SYNTAX      Olsrv2MetricValueCompressedFormTC
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This object is the metric of the route
        to the destination with address R_dest_addr."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
 ::= { olsrv2TibRoutingSetEntry 9 }
```

```
--
-- OLSRv2 Performance Group
--

--
--   Contains objects which help to characterize the
--   performance of the OLSRv2 routing process.
--

olsrv2PerformanceObjGrp  OBJECT IDENTIFIER ::= {olsrv2MIBObjects 3}

--
--   Objects per local interface
--

olsrv2InterfacePerfTable  OBJECT-TYPE
    SYNTAX      SEQUENCE OF Olsrv2InterfacePerfEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This table summarizes performance objects that are
        measured per each active local OLSRv2 interface.
        If the olsrv2InterfaceAdminStatus of the interface
        changes to 'disabled' then the row associated with this
        interface SHOULD be removed from this table."
    REFERENCE
        "RFC XXXX - The Optimized Link State Routing Protocol
        version 2, Clausen, T., Dearlove, C., Jacquet, P.
        and U. Herberg, March 2013."
    ::= { olsrv2PerformanceObjGrp 1 }

olsrv2InterfacePerfEntry  OBJECT-TYPE
    SYNTAX      Olsrv2InterfacePerfEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A single entry contains performance counters for
        each active local OLSRv2 interface."
    AUGMENTS { nhdpInterfacePerfEntry }
    ::= { olsrv2InterfacePerfTable 1 }

Olsrv2InterfacePerfEntry ::=
    SEQUENCE {
        olsrv2IfTcMessageXmits
            Counter32,
        olsrv2IfTcMessageRecvd
            Counter32,
        olsrv2IfTcMessageXmitAccumulatedSize
```

```
        Counter64,
    olsrv2IfTcMessageRecvdAccumulatedSize
        Counter64,
    olsrv2IfTcMessageTriggeredXmits
        Counter32,
    olsrv2IfTcMessagePeriodicXmits
        Counter32,
    olsrv2IfTcMessageForwardedXmits
        Counter32,
    olsrv2IfTcMessageXmitAccumulatedMPRSelectorCount
        Counter32
}

olsrv2IfTcMessageXmits OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "messages"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "A counter is incremented each time a TC
         message has been transmitted on that interface."
 ::= { olsrv2InterfacePerfEntry 1 }

olsrv2IfTcMessageRecvd OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "messages"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "A counter is incremented each time a
         TC message has been received on that interface.
         This excludes all messages that are ignored due to
         OLSRv2 protocol procedures."
 ::= { olsrv2InterfacePerfEntry 2 }

olsrv2IfTcMessageXmitAccumulatedSize OBJECT-TYPE
    SYNTAX      Counter64
    UNITS        "octets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "A counter is incremented by the number of octets in
         a TC message each time a TC message has been sent."
 ::= { olsrv2InterfacePerfEntry 3 }

olsrv2IfTcMessageRecvdAccumulatedSize OBJECT-TYPE
    SYNTAX      Counter64
    UNITS        "octets"
```

```
MAX-ACCESS    read-only
STATUS        current
DESCRIPTION
    "A counter is incremented by the number of octets in
    a TC message each time a TC message has been received.
    This excludes all messages that are ignored due to
    OLSRv2 protocol procedures."
 ::= { olsrv2InterfacePerfEntry 4 }

olsrv2IfTcMessageTriggeredXmits OBJECT-TYPE
    SYNTAX      Counter32
    UNITS       "messages"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "A counter is incremented each time a triggered
        TC message has been sent."
 ::= { olsrv2InterfacePerfEntry 5 }

olsrv2IfTcMessagePeriodicXmits OBJECT-TYPE
    SYNTAX      Counter32
    UNITS       "messages"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "A counter is incremented each time a periodic
        TC message has been sent."
 ::= { olsrv2InterfacePerfEntry 6 }

olsrv2IfTcMessageForwardedXmits OBJECT-TYPE
    SYNTAX      Counter32
    UNITS       "messages"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "A counter is incremented each time a
        TC message has been forwarded."
 ::= { olsrv2InterfacePerfEntry 7 }

olsrv2IfTcMessageXmitAccumulatedMPRSelectorCount OBJECT-TYPE
    SYNTAX      Counter32
    UNITS       "advertised MPR selectors"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "A counter is incremented by the number of advertised
        MPR selectors in a TC each time a TC
        message has been sent."
```

```
::= { olsrv2InterfacePerfEntry 8 }

--
-- Objects concerning the Routing set
--

olsrv2RoutingSetRecalculationCount OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "recalculations"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "This counter increments each time the Routing Set has
        been recalculated."
 ::= { olsrv2PerformanceObjGrp 2 }

--
-- Objects concerning the MPR set
--

olsrv2MPRSetRecalculationCount OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "recalculations"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "This counter increments each time the MPRs
        of this router have been recalculated for
        any of its interfaces."
 ::= { olsrv2PerformanceObjGrp 3 }

--
-- Notifications
--

olsrv2NotificationsObjects OBJECT IDENTIFIER ::=
    { olsrv2MIBNotifications 0 }
olsrv2NotificationsControl OBJECT IDENTIFIER ::=
    { olsrv2MIBNotifications 1 }
olsrv2NotificationsStates  OBJECT IDENTIFIER ::=
    { olsrv2MIBNotifications 2 }
```

```
-- olsrv2NotificationsObjects

olsrv2RouterStatusChange NOTIFICATION-TYPE
    OBJECTS { olsrv2OrigIpAddrType, -- The address type of
                                                -- the originator of
                                                -- the notification.
                olsrv2OrigIpAddr,    -- The originator of
                                                -- the notification.
                olsrv2AdminStatus    -- The new state.
            }
    STATUS      current
    DESCRIPTION
        "olsrv2RouterStatusChange is a notification generated
        when the OLSRv2 router changes its status.
        The router status is maintained in the
        olsrv2AdminStatus object."
 ::= { olsrv2NotificationsObjects 1 }

olsrv2OrigIpAddrChange NOTIFICATION-TYPE
    OBJECTS { olsrv2OrigIpAddrType, -- The address type of
                                                -- the originator of
                                                -- the notification.
                olsrv2OrigIpAddr,    -- The originator of
                                                -- the notification.
                olsrv2PreviousOrigIpAddrType, -- The address
                                                -- type of previous
                                                -- address of
                                                -- the originator of
                                                -- the notification.
                olsrv2PreviousOrigIpAddr -- The previous
                                                -- address of the
                                                -- originator of
                                                -- the notification.
            }
    STATUS      current
    DESCRIPTION
        "olsrv2OrigIpAddrChange is a notification generated when
        the OLSRv2 router changes its originator IP address.
        The notification includes the new and the previous
        originator IP address of the OLSRv2 router."
 ::= { olsrv2NotificationsObjects 2 }

olsrv2RoutingSetRecalculationCountChange NOTIFICATION-TYPE
    OBJECTS { olsrv2OrigIpAddrType, -- The address type of
                                                -- the originator of
                                                -- the notification.
                olsrv2OrigIpAddr,    -- The originator of
                                                -- the notification.
```



```
        olsrv2RoutingSetRecalculationCount  -- Number
                                           -- of the
                                           -- routing set
                                           -- recalculations.
    }
    STATUS          current
    DESCRIPTION
        "The olsrv2RoutingSetRecalculationCountChange
        notification is generated when a significant number of
        routing set recalculations have occurred in a short time.
        This notification SHOULD be generated no more than once
        per olsrv2RoutingSetRecalculationCountWindow.
        The network administrator SHOULD select
        appropriate values for 'significant number of
        routing set recalculations' and 'short time' through
        the settings of the
        olsrv2RoutingSetRecalculationCountThreshold
        and olsrv2RoutingSetRecalculationCountWindow objects."
    ::= { olsrv2NotificationsObjects 3 }

olsrv2MPRSetRecalculationCountChange NOTIFICATION-TYPE
    OBJECTS { olsrv2OrigIpAddrType, -- The address type of
                                           -- the originator of
                                           -- the notification.
              olsrv2OrigIpAddr,      -- The originator of
                                           -- the notification.
              olsrv2MPRSetRecalculationCount -- Number of
                                           -- MPR set
                                           -- recalculations.
    }
    STATUS          current
    DESCRIPTION
        "The olsrv2MPRSetRecalculationCountChange
        notification is generated when a significant
        number of MPR set recalculations occur in
        a short period of time. This notification
        SHOULD be generated no more than once
        per olsrv2MPRSetRecalculationCountWindow.
        The network administrator SHOULD select
        appropriate values for 'significant number of
        MPR set recalculations' and 'short period of
        time' through the settings of the
        olsrv2MPRSetRecalculationCountThreshold and
        olsrv2MPRSetRecalculationCountWindow objects."
    ::= { olsrv2NotificationsObjects 4 }

-- olsrv2NotificationsControl
```

```
olsrv2RoutingSetRecalculationCountThreshold OBJECT-TYPE
    SYNTAX      Integer32 (0..255)
    UNITS        "recalculations"
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "A threshold value for the
         olsrv2RoutingSetRecalculationCount object.
         If the number of occurrences exceeds this
         threshold within the previous
         olsrv2RoutingSetRecalculationCountWindow,
         then the olsrv2RoutingSetRecalculationCountChange
         notification is to be generated.

         It is RECOMMENDED that the value of this
         threshold be set to at least 20 and higher
         in dense topologies with frequent expected
         topology changes."
    DEFVAL { 20 }
 ::= { olsrv2NotificationsControl 1 }

olsrv2RoutingSetRecalculationCountWindow OBJECT-TYPE
    SYNTAX      TimeTicks
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "This object is used to determine whether to generate
         an olsrv2RoutingSetRecalculationCountChange notification.
         This object represents an interval from the present moment,
         extending into the past, expressed in hundredths of
         a second.  If the change in the value of the
         olsrv2RoutingSetRecalculationCount object during
         this interval has exceeded the value of
         olsrv2RoutingSetRecalculationCountThreshold, then
         an olsrv2RoutingSetRecalculationCountChange notification
         is generated.

         It is RECOMMENDED that the value for this
         window be set to at least 5 times the
         nhdpHelloInterval (whose default value is
         2 seconds."
    DEFVAL { 1000 }
 ::= { olsrv2NotificationsControl 2 }

olsrv2MPRSetRecalculationCountThreshold OBJECT-TYPE
    SYNTAX      Integer32 (0..255)
    UNITS        "recalculations"
    MAX-ACCESS   read-write
```

```
STATUS      current
DESCRIPTION
    "A threshold value for the
    olsrv2MPRSetRecalculationCount object.
    If the number of occurrences exceeds this
    threshold within the previous
    olsrv2MPRSetRecalculationCountWindow,
    then the
    olsrv2MPRSetRecalculationCountChange
    notification is to be generated.

    It is RECOMMENDED that the value of this
    threshold be set to at least 20 and higher
    in dense topologies with frequent expected
    topology changes."
    DEFVAL { 20 }
 ::= { olsrv2NotificationsControl 3 }

olsrv2MPRSetRecalculationCountWindow OBJECT-TYPE
SYNTAX      TimeTicks
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION
    "This object is used to determine whether to generate
    an olsrv2MPRSetRecalculationCountChange notification.
    This object represents an interval from the present moment,
    extending into the past, expressed in hundredths of
    a second.  If the change in the value of the
    olsrv2MPRSetRecalculationCount object during
    that interval has exceeded the value of
    olsrv2MPRSetRecalculationCountThreshold, then the
    an olsrv2MPRSetRecalculationCountChange notification
    is generated.

    It is RECOMMENDED that the value for this
    window be set to at least 5 times the
    nhdpHelloInterval."
    DEFVAL { 1000 }
 ::= { olsrv2NotificationsControl 4 }

olsrv2PreviousOrigIpAddressType OBJECT-TYPE
SYNTAX      InetAddressType { ipv4(1) , ipv6(2) }
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "The type of the olsrv2PreviousOrigIpAddress,
    as defined in the InetAddress MIB module (RFC 4001).
```

Only the values 'ipv4(1)' and  
'ipv6(2)' are supported.

This object MUST have the same persistence  
characteristics as olsrv2PreviousOrigIpAddress."

REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2NotificationsStates 1 }

olsrv2PreviousOrigIpAddress OBJECT-TYPE

SYNTAX InetAddress (SIZE(4|16))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The previous origination IP address  
of this OLSRv2 router.

This object SHOULD be updated each time  
the olsrv2OrigIpAddress is modified.

This object is persistent and when written  
the entity SHOULD save the change to  
non-volatile storage."

REFERENCE

"RFC XXXX - The Optimized Link State Routing Protocol  
version 2, Clausen, T., Dearlove, C., Jacquet, P.  
and U. Herberg, March 2013."

::= { olsrv2NotificationsStates 2 }

--

-- Compliance Statements

--

olsrv2Compliances OBJECT IDENTIFIER ::= { olsrv2MIBConformance 1 }

olsrv2MIBGroups OBJECT IDENTIFIER ::= { olsrv2MIBConformance 2 }

olsrv2BasicCompliance MODULE-COMPLIANCE

STATUS current

DESCRIPTION

"The basic implementation requirements for  
managed network entities that implement  
the OLSRv2 routing process."

MODULE -- this module

MANDATORY-GROUPS { olsrv2ConfigObjectsGroup }

```
::= { olsrv2Compliances 1 }

olsrv2FullCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION
        "The full implementation requirements for
        managed network entities that implement
        the OLSRv2 routing process."
    MODULE -- this module
    MANDATORY-GROUPS { olsrv2ConfigObjectsGroup,
                        olsrv2StateObjectsGroup,
                        olsrv2PerfObjectsGroup,
                        olsrv2NotificationsObjectsGroup,
                        olsrv2NotificationsGroup }
::= { olsrv2Compliances 2 }

--
-- Units of Conformance
--

olsrv2ConfigObjectsGroup OBJECT-GROUP
    OBJECTS {
        olsrv2AdminStatus,
        olsrv2InterfaceAdminStatus,
        olsrv2OrigIpAddrType,
        olsrv2OrigIpAddr,
        olsrv2OHoldTime,
        olsrv2TcInterval,
        olsrv2TcMinInterval,
        olsrv2THoldTime,
        olsrv2AHoldTime,
        olsrv2RxHoldTime,
        olsrv2PHoldTime,
        olsrv2FHoldTime,
        olsrv2TpMaxJitter,
        olsrv2TtMaxJitter,
        olsrv2FMaxJitter,
        olsrv2TcHopLimit,
        olsrv2WillFlooding,
        olsrv2WillRouting,
        olsrv2LinkMetricType
    }
    STATUS current
    DESCRIPTION
        "Objects to permit configuration of OLSRv2.
        All of these SHOULD be backed by non-volatile
        storage."
::= { olsrv2MIBGroups 1 }
```

```
olsrv2StateObjectsGroup  OBJECT-GROUP
  OBJECTS {
    olsrv2LibOrigSetExpireTime,
    olsrv2LibLocAttNetSetDistance,
    olsrv2LibLocAttNetSetMetricValue,
    olsrv2IibLinkSetInMetricValue,
    olsrv2IibLinkSetOutMetricValue,
    olsrv2IibLinkSetMprSelector,
    olsrv2Iib2HopSetInMetricValue,
    olsrv2Iib2HopSetOutMetricValue,
    olsrv2NibNeighborSetNOrigIpAddrType,
    olsrv2NibNeighborSetNOrigIpAddr,
    olsrv2NibNeighborSetNInMetricValue,
    olsrv2NibNeighborSetNOutMetricValue,
    olsrv2NibNeighborSetNWillFlooding,
    olsrv2NibNeighborSetNWillRouting,
    olsrv2NibNeighborSetNFloodingMpr,
    olsrv2NibNeighborSetNRoutingMpr,
    olsrv2NibNeighborSetNMprSelector,
    olsrv2NibNeighborSetNAdvertised,
    olsrv2NibNeighborSetTableAnsn,
    olsrv2TibAdRemoteRouterSetMaxSeqNo,
    olsrv2TibRouterTopologySetSeqNo,
    olsrv2TibRouterTopologySetMetricValue,
    olsrv2TibRoutableAddressTopologySetExpireTime,
    olsrv2TibRoutableAddressTopologySetSeqNo,
    olsrv2TibRoutableAddressTopologySetMetricValue,
    olsrv2TibAttNetworksSetSeqNo,
    olsrv2TibAttNetworksSetDist,
    olsrv2TibAttNetworksSetMetricValue,
    olsrv2TibAttNetworksSetExpireTime,
    olsrv2TibRoutingSetNextIfIpAddrType,
    olsrv2TibRoutingSetNextIfIpAddr,
    olsrv2TibRoutingSetLocalIfIpAddrType,
    olsrv2TibRoutingSetLocalIfIpAddr,
    olsrv2TibRoutingSetDist,
    olsrv2TibRoutingSetMetricValue
  }
  STATUS      current
  DESCRIPTION
    "Objects to permit monitoring of OLSRv2 state."
 ::= { olsrv2MIBGroups 2 }

olsrv2PerfObjectsGroup  OBJECT-GROUP
  OBJECTS {
    olsrv2IfTcMessageXmits,
    olsrv2IfTcMessageRecvd,
    olsrv2IfTcMessageXmitAccumulatedSize,
```

```
    olsrv2IfTcMessageRecvdAccumulatedSize,
    olsrv2IfTcMessageTriggeredXmits,
    olsrv2IfTcMessagePeriodicXmits,
    olsrv2IfTcMessageForwardedXmits,
    olsrv2IfTcMessageXmitAccumulatedMPRSelectorCount,
    olsrv2RoutingSetRecalculationCount,
    olsrv2MPRSetRecalculationCount
  }
  STATUS      current
  DESCRIPTION
    "Objects to support monitoring of OLSRv2 performance."
 ::= { olsrv2MIBGroups 3 }

olsrv2NotificationsObjectsGroup OBJECT-GROUP
  OBJECTS {
    olsrv2RoutingSetRecalculationCountThreshold,
    olsrv2RoutingSetRecalculationCountWindow,
    olsrv2MPRSetRecalculationCountThreshold,
    olsrv2MPRSetRecalculationCountWindow,
    olsrv2PreviousOrigIpAddressType,
    olsrv2PreviousOrigIpAddress
  }
  STATUS      current
  DESCRIPTION
    "Objects to support the notification types in the
     olsrv2NotificationsGroup.  Some of these appear in
     notification payloads, others serve to control
     notification generation."
 ::= { olsrv2MIBGroups 4 }

olsrv2NotificationsGroup NOTIFICATION-GROUP
  NOTIFICATIONS {
    olsrv2RouterStatusChange,
    olsrv2OrigIpAddressChange,
    olsrv2RoutingSetRecalculationCountChange,
    olsrv2MPRSetRecalculationCountChange
  }
  STATUS      current
  DESCRIPTION
    "Notification types to support management of OLSRv2."
 ::= { olsrv2MIBGroups 5 }

END
```

## 8. Security Considerations

This MIB module defines objects for the configuration, monitoring and notification of the Optimized Link State Routing protocol version 2 [OLSRv2]. OLSRv2 allows routers to acquire topological information of the routing domain by virtue of exchanging TC message, to calculate shortest paths to each destination router in the routing domain, to select relays for network-wide transmissions etc.

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations. These are the tables and objects and their sensitivity/vulnerability:

- o `olsrv2TcInterval`, `olsrv2TcMinInterval` - these writable objects control the rate at which TC messages are sent. If set at too high a rate, this could represent a form of DOS attack by overloading interface resources. If set low, OLSRv2 may not converge fast enough to provide accurate routes to all destinations in the routing domain.
- o `olsrv2TcHopLimit` - defines the hop limit for TC messages. If set too low, messages will not be forwarded beyond the defined scope, and thus routers further away from the message originator will not be able to construct appropriate topology graphs.
- o `olsrv2OHoldTime`, `olsrv2THoldTime`, `olsrv2AHoldTime`, `olsrv2RxHoldTime`, `olsrv2PHoldTime`, `olsrv2FHoldTime` - define hold times for tuples of different Information Bases of OLSRv2. If set too low, information will expire quickly, and may this harm a correct operation of the routing protocol.
- o `olsrv2WillFlooding` and `olsrv2WillRouting` - define the willingness of this router to become MPR. If this is set to `WILL_NEVER (0)`, the managed router will not forward any TC messages, nor accept a selection to become MPR by neighboring routers. If set to `WILL_ALWAYS (15)`, the router will be preferred by neighbors during MPR selection, and may thus attract more traffic.
- o `olsrv2TpMaxJitter`, `olsrv2TtMaxJitter`, `olsrv2FMaxJitter` - define jitter values for TC message transmission and forwarding. If set too low, control traffic may get lost if the channel is lossy.
- o `olsrv2LinkMetricType` - defines the type of the link metric that a router uses (e.g., ETX or hop-count). Whenever this value



changes, all link metric information recorded by the router is invalid, causing a reset of information acquired from other routers in the MANET. Moreover, if `olsrv2LinkMetricType` on a router is set to a value that is not known to other routers in the MANET, these routers will not be able to establish routes to that router or transiting that router. Existing routes to the router with a `olsrv2LinkMetricType` unknown to other routers in the MANET will be removed.

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to even encrypt the values of these objects when sending them over the network via SNMP. These are the tables and objects and their sensitivity/vulnerability:

- o `olsrv2TibRouterTopologySetTable` - The contains information on the topology of the MANET, specifically the IP address of the routers in the MANET (as identified by `olsrv2TibRouterTopologySetFromOrigIpAddr` and `olsrv2TibRouterTopologySetToOrigIpAddr` objects). This information provides an adversary broad information on the members of the MANET, located within this single table. This information can be used to expedite attacks on the other members of the MANET without having to go through a laborious discovery process on their own.

Some of the Tables in this MIB AUGMENT Tables defined in NHDP-MIB [RFC6779]. Hence, care must be taken in configuring access control here in order make sure that the permitted permissions granted for the AUGMENT-ing Tables here are consistent with the access controls permitted within the NHDP-MIB. The below list identifies the AUGMENT-ing Tables and their NHDP-MIB counterparts. It is recommend that access control policies for these Table pairs are consistently set.

- o The `olsrv2InterfaceTable` AUGMENTS the `nhdpInterfaceTable`.
- o The `olsrv2IibLinkSetTable` AUGMENTS the `nhdpIibLinkSetTable`.
- o The `olsrv2Iib2HopSetTable` AUGMENTS the `nhdpIib2HopSetTable`.
- o The `olsrv2NibNeighborSetTable` AUGMENTS the `nhdpNibNeighborSetTable`.
- o The `olsrv2InterfacePerfTable` AUGMENTS the `nhdpInterfacePerfTable`.

MANET technology is often deployed to support communications of

emergency services or military tactical applications. In these applications, it is imperative to maintain the proper operation of the communications network and to protect sensitive information related to its operation. Therefore, when implementing these capabilities, the full use of SNMPv3 cryptographic mechanisms for authentication and privacy is RECOMMENDED.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPsec), there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.

Implementations SHOULD provide the security features described by the SNMPv3 framework (see [RFC3410]), and implementations claiming compliance to the SNMPv3 standard MUST include full support for authentication and privacy via the User-based Security Model (USM) [RFC3414] with the AES cipher algorithm [RFC3826]. Implementations MAY also provide support for the Transport Security Model (TSM) [RFC5591] in combination with a secure transport such as SSH [RFC5592] or TLS/DTLS [RFC6353].

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

## 9. Applicability Statement

This document describes objects for configuring parameters of the Optimized Link State Routing version 2 (OLSRv2) Protocol [OLSRv2] process on a router. This MIB module, denoted OLSRv2-MIB, also reports state, performance information and notifications. The OLSRv2 protocol relies upon information gathered via the Neighborhood Discovery Protocol [RFC6130] in order to perform its operations. The NHDP protocol is managed via the NHDP-MIB [RFC6779].

MANET deployments can greatly differ in aspects of dynamics of the topology, capacity and loss rates of underlying channels, traffic flow directions, memory and CPU capacity of routers etc. SNMP and therefore this MIB module are only applicable for a subset of MANET deployments, in particular deployments:

- o In which routers have enough memory and CPU resources to run SNMP and expose the MIB module.

- o Where a network management station (NMS) is defined to which notifications are generated, and from which routers can be managed.
- o Where this NMS is reachable from routers in the MANET most of the time (as notifications to the NMS and management information from the NMS to the router will be lost when connectivity is temporarily lost). This requires that the topology of the MANET is only moderately dynamic.
- o Where the underlying wireless channel supports enough bandwidth to run SNMP, and where loss rates of the channel are not exhaustive.

Certain MANET deployments, such as community networks with non-mobile routers, dynamic topology because of changing link quality, and a pre-defined gateway (that could also serve as NMS), are examples of networks applicable for this MIB module. Other, more constrained deployments of MANETs may not be able to run SNMP and require different management protocols.

Some level of configuration, i.e., read-write objects, is desirable for OLSRv2 deployments. Topology related configuration such as the ability to enable OLSRv2 on new interfaces or initially configure OLSRv2 on a router's interfaces through the `olsrv2InterfaceAdminStatus` object is critical to initial system startup. The OLSRv2 protocol allows for some level of performance tuning through various protocol parameters and this MIB module allows for configuration of those protocol parameters through read-write objects such as the `olsrv2TcHopLimit` or the `olsrv2FMaxJitter`. Other read-write objects allow for the control of Notification behavior through this MIB module, e.g., the `olsrv2RoutingSetRecalculationCountThreshold` object. A fuller discussion of MANET network management applicability is to be provided elsewhere [USE-CASES].

## 10. IANA Considerations

The RFC editor should remove the specification of the `IANAolsrv2LinkMetricType-MIB` found in Appendix A upon publication of the OLSRv2-MIB. Further, IANA should take over the `IANAolsrv2LinkMetricType-MIB` and keep it synchronized with the registry identified below within the `IANAolsrv2LinkMetricType` TEXTUAL-CONVENTION.

The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER value recorded in the SMI Numbers registry:

Descriptor	OBJECT IDENTIFIER value
-----	-----
OLSRv2-MIB	{ mib-2 ZZZZ }
IANA EDITOR NOTE: please assign ZZZZ	

## 11. Acknowledgements

The authors would like to thank Randy Presuhn, Benoit Claise, Adrian Farrel, as well as the entire MANET WG for reviews of this document.

This MIB document uses the template authored by D. Harrington which is based on contributions from the MIB Doctors, especially Juergen Schoenwaelder, Dave Perkins, C.M.Heard and Randy Presuhn.

## 12. References

### 12.1. Normative References

- [OLSRv2] Clausen, T., Dearlove, C., Jacquet, P., and U. Herberg, "The Optimized Link State Routing Protocol version 2", draft-ietf-manet-olsrv2-19 (work in progress), March 2013.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.
- [RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.
- [RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.
- [RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.
- [RFC3414] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", December 2002.
- [RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62,

RFC 3418, December 2002.

- [RFC3826] Blumenthal, U., Maino, F., and K. McCloghrie, "The Advanced Encryption Standard (AES) Cipher Algorithm in the SNMP User-based Security Model", June 2004.
- [RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.
- [RFC5591] Harrington, D. and W. Hardaker, "Transport Security Model for the Simple Network Management Protocol (SNMP)", June 2009.
- [RFC5592] Harrington, D., Saloway, J., and W. Hardaker, "Secure Shell Transport Model for the Simple Network Management Protocol (SNMP)", June 2009.
- [RFC6130] Clausen, T., Dearlove, C., and J. Dean, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)", RFC 6130, April 2011.
- [RFC6353] Hardaker, W., "Transport Layer Security (TLS) Transport Model for the Simple Network Management Protocol (SNMP)", July 2011.
- [RFC6779] Herberg, U., Cole, R., and I. Chakeres, "Definition of Managed Objects for the Neighborhood Discovery Protocol", RFC 6779, May 2012.

## 12.2. Informative References

- [RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.
- [REPORT-MIB] Cole, R., Macker, J., and A. Bierman, "Definition of Managed Objects for Performance Reporting", draft-ietf-manet-report-mib-03 (work in progress), November 2012.
- [USE-CASES] Nguyen, J., Cole, R., Herberg, U., Yi, J., and J. Dean, "Network Management of Mobile Ad hoc Networks (MANET): Architecture, Use Cases, and Applicability", draft-nguyen-manet-management-00 (work in progress), February 2013.

## Appendix A. Appendix A:

This appendix contains the IANAolsrv2LinkMetricType-MIB module defined by this specification below. The RFC editor should remove this specification of the IANAolsrv2LinkMetricType-MIB upon publication of the OLSRv2-MIB. Further, IANA should take over the IANAolsrv2LinkMetricType-MIB and to keep it synchronized with the registry identified below within the IANAolsrv2LinkMetricType TEXTUAL-CONVENTION.

```
IANAolsrv2LinkMetricType-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY, mib-2
        FROM SNMPv2-SMI
    TEXTUAL-CONVENTION
        FROM SNMPv2-TC;

ianaolsrv2LinkMetricType MODULE-IDENTITY
    LAST-UPDATED "201306240000Z" -- June 24, 2013
    ORGANIZATION "IANA"
    CONTACT-INFO "Internet Assigned Numbers Authority

        Postal: ICANN
                4676 Admiralty Way, Suite 330
                Marina del Rey, CA 90292

        Tel:      +1 310 823 9358
        E-Mail:    iana@iana.org"
    DESCRIPTION "This MIB module defines the
        IANAolsrv2LinkMetricType Textual
        Convention, and thus the enumerated values of
        the olsrv2LinkMetricType object defined in
        the OLSRv2-MIB."
    REVISION "201306240000Z" -- June 24, 2013
    DESCRIPTION "Initial version of this MIB as published in
        RFC KKKK."

    ::= { mib-2 kkkk }

IANAolsrv2LinkMetricTypeTC ::= TEXTUAL-CONVENTION
    STATUS      current
    DESCRIPTION
        "This data type is used as the syntax of the
        olsrv2LinkMetricType object in the definition
        of the OLSRv2-MIB module.
```

The `olsrv2LinkMetricType` corresponds to `LINK_METRIC_TYPE` of OLSRv2 (RFC XXXX). OLSRv2 uses bidirectional additive link metrics to determine shortest distance routes (i.e., routes with smallest total of link metric values).

OLSRv2 has established a registry for the `LINK_METRIC_TYPES` (denoted 'LINK\_METRIC Address Block TLV Type Extensions'):

```
http://www.iana.org/assignments/manet-parameters/  
manet-parameters.xml#  
link-metric-address-block-tlv-type-extension
```

This is done in Section 24.5 in OLSRv2 (RFC XXXX). The `LINK_METRIC_TYPE` (which has as corresponding object in the MIB module `olsrv2LinkMetricType`) corresponds to the type extension of the `LINK_METRIC` TLV that is set up in the 'LINK\_METRIC Address Block TLV Type Extensions' registry. Whenever new link metric types are added to that registry, IANA MUST update this textual convention accordingly.

The definition of this textual convention with the addition of newly assigned values is published periodically by the IANA, in either the Assigned Numbers RFC, or some derivative of it specific to Internet Network Management number assignments. (The latest arrangements can be obtained by contacting the IANA.)

Requests for new values should be made to IANA via email ([iana@iana.org](mailto:iana@iana.org)).

```
SYNTAX  INTEGER {  
    unknown(0)      -- Link metric meaning assigned  
                    -- by administrative action  
                    -- 1-223 Unassigned  
                    -- 224-255 Reserved for  
                    -- Experimental Use  
}  
END
```

## Appendix B. Note to the RFC Editor

```
*****
* Note to the RFC Editor (to be removed prior to publication) *
*
* 1) The reference to RFCYYYY within the DESCRIPTION clauses *
* of the MIB module point to this draft and are to be *
* assigned by the RFC Editor. *
*
* 2) The reference to RFCXXXX throughout this document point *
* to the current draft-ietf-manet-olsrv2-xx.txt. This *
* needs to be replaced with the XXXX RFC number for the *
* OLSRv2 publication. *
*
* 3) Appendix A contains the IANAolsrv2LinkMetricType-MIB *
* module which is defined by this specification in the *
* appendix. The RFC editor should remove this specification *
* of the IANAolsrv2LinkMetricType-MIB upon publication of *
* the OLSRv2-MIB. Further, IANA should take over the *
* IANAolsrv2LinkMetricType-MIB and keep it synchronized *
* with the registry identified within the contained *
* IANAolsrv2LinkMetricType TEXTUAL-CONVENTION. *
*
*****
```

## Authors' Addresses

Ulrich Herberg  
Fujitsu Laboratories of America  
1240 East Arques Avenue  
Sunnyvale, CA 94085  
USA

EEmail: [ulrich@herberg.name](mailto:ulrich@herberg.name)  
URI: <http://www.herberg.name/>

Robert G. Cole  
US Army CERDEC  
6010 Frankford Road, Bldg 6010  
Aberdeen Proving Ground, Maryland 21005  
USA

Phone: +1 443 395 8744  
EEmail: [robert.g.cole@us.army.mil](mailto:robert.g.cole@us.army.mil)  
URI: <http://www.cs.jhu.edu/~rgcole/>



Thomas Heide Clausen  
LIX, Ecole Polytechnique  
Palaiseau Cedex, 91128  
France

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



Internet Engineering Task Force  
Internet-Draft  
Intended status: Experimental  
Expires: June 10, 2015

R. Cole  
US Army CERDEC  
J. Macker  
Naval Research Laboratory  
A. Bierman  
YumaWorks, Inc.  
December 7, 2014

Definition of Managed Objects for Performance Reporting  
draft-ietf-manet-report-mib-04

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes objects for configuring autonomous report generation on any device that supports MIBs containing objects that resolve to type Integer32 (i.e., Integer32, Counter, Gauge, or TimeTicks). to be used for performance monitoring. This allows a management station to instruct a device to build off-line reports to be collected either through notifications to the management station or queried asynchronously by the management station. Hence, this capability allows network operators to reduce the SNMP polling traffic burden on Mobile Ad-Hoc and Disruption Tolerant Networks which is problematic of SNMP performance management applications.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on June 10, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. The Internet-Standard Management Framework . . . . .	3
3. Conventions . . . . .	4
4. Overview . . . . .	4
4.1. reportSampledMIB Module Management Model . . . . .	4
4.2. Terms . . . . .	5
5. Structure of the MIB Module . . . . .	6
5.1. Textual Conventions . . . . .	7
5.2. Tables and Indexing . . . . .	7
6. Relationship to Other MIB Modules . . . . .	8
6.1. Relationship to the SNMPv2-MIB . . . . .	8
6.2. Relationship to the RMON2-MIB . . . . .	9
6.3. Relationship to the DISMAN-EVENT-MIB . . . . .	9
6.4. Relationship to the DISMAN-EXPRESSION-MIB . . . . .	10
6.5. MIB modules required for IMPORTS . . . . .	10
7. Definitions . . . . .	11
8. Security Considerations . . . . .	26
9. Applicability Statement . . . . .	29
10. IANA Considerations . . . . .	30
11. Contributors . . . . .	30
12. Acknowledgements . . . . .	30
13. References . . . . .	30
13.1. Normative References . . . . .	30
13.2. Informative References . . . . .	31
Appendix A. Change Log . . . . .	31
Appendix B. Open Issues . . . . .	33
Appendix C. . . . .	34

## 1. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes objects for configuring autonomous, off-line report generation for performance monitoring on any device supporting MIBs containing variables that resolve to type Integer32 (i.e., Integer32, Counter, Gauge, or TimeTicks). This reportSampledMIB module allows for the report generation to occur on the same device as containing the referenced counter object. This should be useful to devices or networks where efficient use of bandwidth is of concern or where intermittent connectivity is common. Hence, the reportSampledMIB module is useful for devices managed over some Mobile Ad-Hoc Networks (MANETs) or Disruption Tolerant Networks (DTNs).

This version of the reportSampledMIB module offers one type of off-line reporting. The MIB offers a means to collect sampled measurements related to defined MIB objects. This type of reporting is contained in the reportSampledMibObjects. Other types of report data are possible, including statistical data. However, it was felt wise to focus on a more limited scope off-line reporting capability and gain experimental use and application prior to expending energy developing a more extensive off line reporting capability.

The reportSampledMIB module relies upon the dismanEventMIB module RFC 2981 [RFC2981] to monitor the progress of reports being developed within the reportSampledMIB module and to trigger an events, i.e., notifications containing reports, at the appropriate times. This is discussed below in more detail in the section entitled 'Relationship to the DISMAN-EVENT-MIB'. Further, more sophisticated performance objects for monitoring from the reportSampledMIB module can be defined through the dismanExpressionMIB module RFC 2982 [RFC2982]

## 2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIV2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

### 3. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

### 4. Overview

The reportSampledMIB module references performance objects in other MIBs and generates off-line performance reports related to those referenced objects. The reportSampledMIB module can be coincident with the other MIB modules on the same device containing the referenced performance related object.

#### 4.1. reportSampledMIB Module Management Model

This section describes the management model for the reportSampledMIB module process.

The reportSampledMIB module objects are primarily contained within four tables. These are:

- o reportSampledControlTable - this is the control table within the reportSampledMIB module. This identifies the OIDs to be monitored which define the core of the Reports. The control table sets the sampling frequency for the Measurements and the number of Measurements that will define each Report.
- o reportSampledCurrentReportsStatusTable - This table tracks the status, i.e., the current number of collected Measurements per each Study. This information can be used by the dismanEventMIB module to determine when to trigger a notification to the Report owner containing the Measurements and associated data comprising the just completed Report.
- o reportSampledCurrentReportsTable - This table holds the Measurements for the Reports which are in current development.
- o reportSampledHistoricalReportsTable - This table holds the completed Reports for each Study for archival purposes, i.e., the Study owners can perform table walks to retrieve archived Reports or Studies.

The below figure illustrates the four main tables within the reportSampledMIB module. Further, if the dismanEventMIB module is so configured to generate triggered notifications, the below figure highlights the 'boolean trigger' and the notification generation. The figure further illustrates the movement of completed Reports from

the reportSampledCurrentReportsTable to the reportSampledHistoricalReportsTable upon Report Completion.



#### 4.2. Terms

The following definitions apply throughout this document:

- o Sampled - periodic measurement of target OIDs.
- o Measurement - a single instance of a sampling event.
- o Report - a collection of consecutive Measurements on the same Sampled target OID.

- o Study - a series of Reports on the same Sampled target OID.
- o Current Report - a Report which is in the process of being developed.
- o A Completed Report - contains a pre-defined number of Sampled Measurements.
- o Historical Report - a Report which has previously Completed, and is being stored locally for archival purposes.

## 5. Structure of the MIB Module

This section presents the structure of the reportSampledMIB module. The objects are arranged into the following groups:

- o reportSampledMibNotifications - defines the notifications associated with the reportSampledMIB module. These objects define notifications which track the behavior of the reportSampledMib module. A single notification is defined in the reportSampledMIB module which reports a series of failed measurement attempts in the process of building a Report. Associated with this notification is a control object which defines a threshold of failures which would initiate the notification. These notifications do not cover the triggered notifications which carry the performance Reports generated by the reportSampledMib module. These triggered notifications are defined through the use of the dismanEventMIB module.
- o reportSampledMibObjects - defines the objects forming the basis for the reportSampledMIB module. These objects are basically divided up by function into the following four tables:
  - \* reportSampledControlTable - This group contains the objects which support the generation (collection) of Studies comprising of Reports exposing sampled Measurement values.
  - \* reportSampledCurrentReportsStatusTable - This group contains the objects which track the collection of Measurements for current (in-progress) Reports. This table allows the dismanEventMIB module to set triggers for Completed Reports which it can then send to the report owner through triggered notifications.
  - \* reportSampledCurrentReportsTable - This group contains the objects which represent the Measurement data associated with Current (in-progress) Reports. Once the Report completes, it is moved to the reportSampledHistoricalReportsTable for



archival purposes.

- \* reportSampledHistoricalReportsTable - This group contains the objects which represent archived Completed Reports. This allows the report owners to asynchronously retrieve Reports via table walks if so desired.
- o reportSampledMibConformance - Defines a single basic conformance of implementations of this reportSampledMIB module.

#### 5.1. Textual Conventions

No textual conventions are defined in the reportSampledMIB module.

#### 5.2. Tables and Indexing

The reportSampledMIB module contains four tables which control and record data related to the creation, notification and storage of Reports. Specifically:

- o the control and generation of remote performance Reports, i.e., reportSampledControlTable
- o the status of the Current Reports' development, i.e., reportSampledCurrentReportsStatusTable,
- o the Current Reports development and interim data, i.e., reportSampledCurrentReportsTable, and
- o the historical storage of remote performance Reports, i.e. reportSampledHistoricalReportsTable.

The reportSampledMIB module's tables are indexed via the following constructs:

- o reportSampledStudyIndex - an index that uniquely identifies a particular Study. The Study is comprised of multiple Reports, the number of Reports being stored is defined by the reportSampledStudyMaximumNumberOfHistoricalReports object.
- o reportSampledCurrentMeasurementIndex - an index that uniquely identifies an atomic Measurement associated with a Report.
- o reportSampledHistoricalReportIndex - an index that uniquely identifies an archived Completed Report resident within the reportSampledHistoricalReportsTable.

- o reportSampledHistoricalMeasurementIndex - an index that uniquely identifies an atomic Measurement comprising an archived Completed Report.

These tables and their indexing are:

- o reportSampledControlTable - this table contains a list of data-collection configuration entries defining aspects of the studies and their reports to be generated. These include, e.g., number of reports per study, the number Reports to be archived, etc. This table has 'INDEX { reportSampledStudyIndex }'.
- o reportSampledCurrentReportsStatusTable - this table contains objects which track the development of current Reports, e.g., the number of current Measurements collected for each Report under development. This table has 'INDEX { reportSampledStudyIndex }'. For each (active) Study, there exists only one Current Report under development.
- o reportSampledCurrentReportsTable - this table contains the Measurements which are developing the Current Reports. This table has 'INDEX { reportSampledStudyIndex, reportSampledCurrentMeasurementIndex }'.
- o reportSampledHistoricalReportsTable - this table contains the Reports which have completed. This table has 'INDEX { reportSampledStudyIndex, reportSampledHistoricalReportIndex, reportSampledHistoricalMeasurementIndex }'.

## 6. Relationship to Other MIB Modules

The text of this section specifies the relationship of the MIB modules contained in this document to other standards, particularly to standards containing other MIB modules. Definitions imported from other MIB modules and other MIB modules that SHOULD be implemented in conjunction with the MIB module contained within this document are identified in this section.

### 6.1. Relationship to the SNMPv2-MIB

The 'system' group in the SNMPv2-MIB [RFC3418] is defined as being mandatory for all systems, and the objects apply to the entity as a whole. The 'system' group provides identification of the management entity and certain other system-wide data. The reportSampledMIB module does not duplicate those objects.

## 6.2. Relationship to the RMON2-MIB

The reportSampledMIB module is closely related to and was inspired by the the RMON2-MIB module [RFC2021] usrHistoryGroup. The use of control tables to establish the periodic collection of measurement data for creation of performance reports was pulled from earlier work on the RMON2-MIB module.

## 6.3. Relationship to the DISMAN-EVENT-MIB

The reportSampledMIB module was developed to fundamentally work with the dismanEventMIB module RFC 2981 [RFC2981] in order to offer a complete and efficient off-line reporting capability for bandwidth challenged networks such as Mobile Ad-Hoc Networks (MANETs). This is accomplished through defining trigger test and associated notification actions indexed by mteOwner, mteTriggerName, mteObjectsName and mteEventName within the dismanEventMIB module. Specifically (within the dismanEventMIB module):

In the mteTriggerTable and specifically by setting

- o 'mteTriggerTest == boolean(1)',
- o 'mteTriggerSampleType == absoluteValue(1)',
- o 'mteTriggerValueID ==  
reportSampledNumberOfMeasurementsForCurrentReport',
- o 'mteTriggerValueIDWildcard == false(1)',
- o 'mteTriggerFrequency == 0.5\*reportSampledStudySamplingInterval',  
and
- o 'mteTriggerEnabled == true'.

In the mteTriggerBooleanTable and specifically by setting

- o 'mteTriggerBooleanComparison == equal(2)',
- o 'mteTriggerBooleanValue == value of  
reportSampledStudyNumberReportMeasurements', and
- o 'mteTriggerBooleanStartup == false'.

In the mteObjectsTable and specifically by setting

- o 'mteObjectsID == reportSampledCurrentMeasurementValue' and reportSampledCurrentMeasurementTime' and reportSampledCurrentMeasurementStatus' with
- o 'mteObjectsIDWildcard == true' // for each.

In the mteEventTable and specifically by setting

- o 'mteEventActions == notification(0)' and
- o 'mteEventEnabled == true' // for each.

In the mteEventNotificationTable and specifically by setting

- o 'mteEventNotification == mteTriggerFired' and the appropriate names for the
- o 'mteEventNotificationObjectOwner == mteOwner' and
- o 'mteEventNotificationObjects == mteObjectsName'.

These settings within the dismanEventMIB module will result in notifications generated by the dismanEventMIB module which will carry the recently completed reportSampledMIB module reports.

Set up properly, the dismanEventMIB module will trigger a notification each time the reportSampledCurrentTable contains a completed Report. This Report will be sent in a notification containing three columns of the reportSampledCurrentTable, i.e., the Value, the Time and the Status, due to the use of wildcarding within the dismanEventMIB module.

Simultaneously, the reportSampledMIB module will move the completed Current Report into the reportSampledHistoricalReportsTable and restart collection for the next Report within the reportSampledCurrentReportsTable.

#### 6.4. Relationship to the DISMAN-EXPRESSION-MIB

In conjunction with the dismanExpressionMIB module RFC 2982 [RFC2982], the reportSampledMIB module can be used to develop reports on relatively sophisticated object expressions.

#### 6.5. MIB modules required for IMPORTS

Citations are not permitted within a MIB module, but any module mentioned in an IMPORTS clause or document mentioned in a REFERENCE clause is a Normative reference, and must be cited someplace within

the narrative sections. Therefore, the imported items in this MIB module, such as Textual Conventions, that are not already cited, are cited in this section. Since relationships to other MIB modules should be described in the narrative text, this section will cite modules from which Textual Conventions are imported.

The reportSampledMIB module IMPORTS objects from SNMPv2-SMI [RFC2578], SNMPv2-TC [RFC2579], SNMPv2-CONF [RFC2580], SNMP-FRAMEWORK-MIB [RFC3411], and SNMPv2-MIB [RFC3418].

## 7. Definitions

REPORT-SAMPLED-MIB DEFINITIONS ::= BEGIN

IMPORTS

```
MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE,
Gauge32, Integer32, experimental
    FROM SNMPv2-SMI                                -- [RFC2578]

TimeStamp
    FROM SNMPv2-TC                                -- [RFC2579]

sysUpTime
    FROM SNMPv2-MIB                                -- [RFC3418]

SnmpAdminString
    FROM SNMP-FRAMEWORK-MIB                        -- [RFC3411]

MODULE-COMPLIANCE, OBJECT-GROUP,
NOTIFICATION-GROUP
    FROM SNMPv2-CONF                                -- [RFC2580]
;
```

```
reportSampledMIB MODULE-IDENTITY
    LAST-UPDATED "201412011300Z" -- December 01, 2014
    ORGANIZATION "IETF MANET Working Group"
    CONTACT-INFO
        "WG E-Mail: manet@ietf.org

        WG Chairs: sratliff@cisco.com
                   jmacker@nrl.navy.mil

        Editors:  Robert G. Cole
                   US Army CERDEC
```

6010 Frankford Road  
Aberdeen Proving Ground, MD 21005  
USA  
+1 443 395-8744  
robert.g.cole@us.army.mil

Joseph Macker  
Naval Research Laboratory  
Washington, D.C. 20375  
USA  
macker@itd.nrl.navy.mil

Andy Bierman  
YumaWorks, Inc.  
andy@yumaworks.com"

## DESCRIPTION

"This MIB module contains managed object definitions for the autonomous reporting of performance object counters. Copyright (C) The IETF Trust (2009). This version of this MIB module is part of RFC xxxx; see the RFC itself for full legal notices."

## -- Revision History

REVISION "201412011300Z" -- December 01, 2014

## DESCRIPTION

"The ninth draft of this MIB module published as draft-ietf-manet-report-mib-04.txt.

Revisions to this draft include

- a) A major restructuring of the MIB module in order to leverage the dismanEventMIB module for the automatic notification of Completed Reports.
- b) Efforts to incorporate this MIB module into the DISMAN management architecture.

"

REVISION "201211051300Z" -- November 05, 2012

## DESCRIPTION

"The seventh draft of this MIB module published as draft-ietf-manet-report-mib-03.txt.

Revisions to this draft include

- a) Added a 'Tables and Indexing' section to the body of this document.
- b) Added an 'Applicability Statement' section to the body of this document."

REVISION "201201311300Z" -- January 31, 2012

## DESCRIPTION

"The sixth draft of this MIB module published as

draft-ietf-manet-report-mib-02.txt.

Revisions to this draft include

- a) Pulled the statistical and historical reporting from the MIB module and left only the sampled reporting, in order to greatly simplify the first instance of this reporting MIB module.
- b) Renamed the module, the reportSampledMIB module.
- c) Leveraged the RMON2-MIB module more effectively through the use of the AUGMENTS clause.
- d) Changed the module to 'experimental'."

REVISION "201102171300Z" -- February 17, 2011  
DESCRIPTION

"The fifth draft of this MIB module published as draft-ietf-manet-report-mib-01.txt. This document has been promoted to a MANET Working Group draft.

Revisions to this draft include

- a) Proposed changes to the statsReport table to simplify communications between device and mgmt application,
- b) Added Notifications,
- c) Changed the reporting structure of the Sampled and the History reporting to align with the structure of the Statistics reports for the purpose of allowing for efficient notification and collection of data reports.
- d) Ran through smilint to clean up all errors and most warning. A few still remain."

REVISION "201007051300Z" -- July 05, 2010  
DESCRIPTION

"The fourth draft of this MIB module published as draft-ietf-manet-report-mib-00.txt. This document has been promoted to a MANET Working Group draft.

Significant revisions to this draft include

- a) added support for proxy configurations through the addition of address objects associated with the referenced counter objects associated with the performance reports."

REVISION "201003021300Z" -- March 02, 2010  
DESCRIPTION

"The third draft of this MIB module published as draft-cole-manet-report-mib-02.txt. Significant revisions to this draft include a) changed naming

```
    of usrHistoryGroup to sampledGroup and b) added
    a historyGroup."
REVISION      "200910251300Z"    -- October 25, 2009
DESCRIPTION
    "The second draft of this MIB module published as
    draft-cole-manet-report-mib-01.txt. Significant
    revisions to this draft include a) the inclusion of
    raw data collection borrow blatantly from the
    usrHistory Group within RMON2, b) the deletion of
    the CurrentHistoryTable from version -00,
    c) modifications to the overall structure of the
    MIB, and d) the definition of various Compliance
    options for implementations related to this MIB."
REVISION      "200904281300Z"    -- April 28, 2009
DESCRIPTION
    "Initial draft of this MIB module published as
    draft-cole-manet-report-mib-00.txt."
-- RFC-Editor assigns XXXX
::= { experimental 998 }    -- to be assigned by IANA


-- TEXTUAL CONVENTIONS
-- None


--
-- Top-Level Object Identifier Assignments
--
reportSampledMibNotifications OBJECT IDENTIFIER
                               ::= { reportSampledMIB 0 }
reportSampledMibObjects       OBJECT IDENTIFIER
                               ::= { reportSampledMIB 1 }
reportSampledMibConformance   OBJECT IDENTIFIER
                               ::= { reportSampledMIB 2 }


-- The reportSampledMibObjects assignments are :
--     reportSampledControlTable           - 1
--     reportSampledCurrentReportsStatusTable - 2
--     reportSampledCurrentReportsTable     - 3
--     reportSampledHistoricalReportsTable  - 4


--
-- The Control Table
--
reportSampledControlTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF reportSampledControlEntry
    MAX-ACCESS   not-accessible
```



```

STATUS      current
DESCRIPTION
    "A table to configure measurement Studies which
    are comprised of multiple Reports."
REFERENCE
    "tbd."
 ::= { reportSampledMibObjects 1 }

reportSampledControlEntry OBJECT-TYPE
    SYNTAX      ReportSampledControlEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A list of parameters that control the creation of
        off-line performance Studies.

        The objects in this table are persistent and when
        written the device SHOULD save the change to
        non-volatile storage.  For further information
        on the storage behavior for these objects, refer
        to the description for the reportSampledStudyEntryStatus
        object."
    INDEX       { reportSampledStudyIndex }
    ::= { reportSampledControlTable 1 }

ReportSampledControlEntry ::= SEQUENCE {
    reportSampledStudyIndex          Integer32,
    reportSampledStudyOwner          SnmpAdminString,
    reportSampledStudyName           SnmpAdminString,
    reportSampledStudyOid            Integer32,
    reportSampledStudySamplingInterval Integer32,
    reportSampledStudyNumberReportMeasurements Integer32,
    reportSampledStudyMaximumNumberOfHistoricalReports Integer32,
    reportSampledStudyEntryStatus    RowStatus
}

reportSampledStudyIndex OBJECT-TYPE
    SYNTAX      Integer32 (1..127)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A unique index that identifies a specific performace
        Study.  Each Study is comprised of multiple
        Reports.  Each Report is comprised of multiple
        atomic Measurements on a specified object."
    ::= { reportSampledControlEntry 1 }

reportSampledStudyOwner OBJECT-TYPE

```

```
SYNTAX      SnmpAdminString (SIZE (0..32))
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
    "The owner of the Study."
DEFVAL      { 'H }
::= { reportSampledControlEntry 2 }

reportSampledStudyName OBJECT-TYPE
SYNTAX      SnmpAdminString (SIZE (0..32))
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
    "The name of the Study."
DEFVAL      { 'H }
::= { reportSampledControlEntry 3 }

reportSampledStudyOid OBJECT-TYPE
SYNTAX      OBJECT IDENTIFIER
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
    "The target OID of the Study.  Each Study makes
    periodic Measurements on a specified object
    which is local to this device.  Currently, the
    objects of study are limited to objects that
    resolve to Integer32 (i.e., Integer32, Counter,
    Gauge, or TimeTicks)."
::= { reportSampledControlEntry 4 }

reportSampledStudySamplingInterval OBJECT-TYPE
SYNTAX      Integer32 (1..2147483647)
UNITS       "seconds"
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
    "The time (in seconds) between sampled Measurement
    instances."
DEFVAL      { 10 }
::= { reportSampledControlEntry 5 }

reportSampledStudyNumberReportMeasurements OBJECT-TYPE
SYNTAX      Integer32 (1..2147483647)
UNITS       "count"
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
    "The number of Measurements per Report for this Study."
```

```
DEFVAL      { 10 }  
::= { reportSampledControlEntry 6 }
```

reportSampledStudyMaximumNumberOfHistoricalReports OBJECT-TYPE

```
SYNTAX      Integer32 (1..2147483647)  
UNITS       "count"  
MAX-ACCESS  read-create  
STATUS      current  
DESCRIPTION  
    "The number of Historical Reports to archive locally  
    for this specific Study. The Historical Reports are  
    archived locally in the  
    reportSampledHistoricalReportsTable (below)."  
DEFVAL      { 10 }  
::= { reportSampledControlEntry 7 }
```

reportSampledStudyEntryStatus OBJECT-TYPE

```
SYNTAX      RowStatus  
MAX-ACCESS  read-create  
STATUS      current  
DESCRIPTION  
    "This object permits management of this table  
    by facilitating actions such as row creation,  
    construction, and destruction. The value of  
    this object has no effect on whether other  
    objects in this conceptual row can be  
    modified.
```

An entry may not exist in the 'active' state unless all objects in the entry have a defined appropriate value. For objects with DEFVAL clauses, the management station does not need to specify the value of these objects in order for the row to transit to the 'active' state; the default value for these objects is used. For objects that do not have DEFVAL clauses, then the network manager MUST specify the value of these objects prior to this row transitioning to the 'active' state.

When this object transitions to 'active', all objects in this row SHOULD be written to non-volatile (stable) storage. Read-create objects in this row MAY be modified. When an object in a row with smfCfgIfRowStatus of 'active' is changed, then the updated value MUST be reflected in SMF and this new object value MUST be written to non-volatile storage.

If this object is not equal to 'active', all associated entries in the reportSampledCurrentReportsStatusTable,

```
        the reportSampledCurrentReportsTable, and the
        reportSampledHistoricalReportsTable MUST be deleted."
 ::= { reportSampledControlEntry 8 }

--
-- the Current Reports Status Table
--
reportSampledCurrentReportsStatusTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF reportSampledCurrentReportsStatusEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table to tracking the progress of measurements of current
        reports in development.  Of particular note is the object
        reportSampledNumberOfMeasurementsForCurrentReport which
        can be compared to the value of the object
        reportSampledStudyNumberReportMeasurements by the
        dismanEventMIB module and generate triggered
        notifications to the Study owner containing the
        recently Completed Reports."
    REFERENCE
        "tbd."
    ::= { reportSampledMibObjects 2 }

reportSampledCurrentReportsStatusEntry OBJECT-TYPE
    SYNTAX      ReportSampledCurrentReportsStatusEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A list of parameters that track the status of
        current Reports in development."
    INDEX       { reportSampledStudyIndex }
    ::= { reportSampledCurrentReportsStatusTable 1 }

ReportSampledCurrentReportsStatusEntry ::= SEQUENCE {
    reportSampledNumberOfCurrentReport      Integer32,
    reportSampledNumberOfMeasurementsForCurrentReport Integer32
}

reportSampledNumberOfCurrentReport OBJECT-TYPE
    SYNTAX      Integer32 (1..2147483647)
    UNITS       "count"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number within the Study series of this
        current Report.  For each new Report within
```

```

        the Study, this value MUST increment by
        one.  For the first Report in this Study,
        the initial value of this object MUST be
        set to one.  The value MUST wrap back to one
        when the value has reached the maximum."
 ::= { reportSampledCurrentReportsStatusEntry 1 }

reportSampledNumberOfMeasurementsForCurrentReport OBJECT-TYPE
    SYNTAX      Integer32 (1..2147483647)
    UNITS       "count"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of Measurements collected so far for
        for this specific Report.  The initial value
        of this object MUST be one.  The value MUST
        increment be one for each attempted Measurement.
        The maximum value for this object is
        reportSampledStudyNumberReportMeasurements.
        Once this value is reached and the next
        Measurement is attempted, the Current Report is
        considered Completed, the agent MUST copy
        the Completed Report's data from the
        reportSampledCurrentReportsTable into the
        reportSampledHistoricalReportsTable, and the
        next Measurement (strating the next Report in the
        Study series) MUST be numbered with the value of
        this object as one."
 ::= { reportSampledCurrentReportsStatusEntry 2 }

--
-- the Current Reports Table
--
reportSampledCurrentReportsTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF ReportSampledCurrentReportsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A list of measurements being collected for active
        Reports."
    REFERENCE
        " TBD."
 ::= { reportSampledMibObjects 3 }

reportSampledCurrentReportsEntry OBJECT-TYPE
    SYNTAX      ReportSampledCurrentReportsEntry
    MAX-ACCESS  not-accessible

```

```

STATUS      current
DESCRIPTION
    "A list of entries storing the measurements from
    active Reports.  Once an active, current Report
    completes (when the value of the associated
    reportSampledNumberOfMeasurementsForCurrentReport
    equals the value of the associated
    reportSampledStudyNumberReportMeasurements), the
    agent MUST move the Report's data from the
    reportSampledCurrentReportsTable to the
    reportSampledHistoricalReportTable."
INDEX       { reportSampledStudyIndex,
              reportSampledCurrentMeasurementIndex }
 ::= { reportSampledCurrentReportsTable 1 }

ReportSampledCurrentReportsEntry ::= SEQUENCE {
    reportSampledCurrentMeasurementIndex  Integer32,
    reportSampledCurrentMeasurementValue  Integer32,
    reportSampledCurrentMeasurementTime   sysUpTime,
    reportSampledCurrentMeasurementStatus INTEGER
}

reportSampledCurrentMeasurementIndex OBJECT-TYPE
    SYNTAX      Integer32 (1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "An index for this table which represents
        the number of Measurements collected so far for
        for this current Report.  The initial value
        of this object MUST be one.  The value MUST
        increment be one for each attempted Measurement.
        The maximum value for this object is
        reportSampledStudyNumberReportMeasurements.
        Once this value is reached and the next
        Measurement is attempted, the Current Report is
        considered Completed, the agent MUST copy
        the Completed Report's data from the
        reportSampledCurrentReportsTable into the
        reportSampledHistoricalReportsTable, and the
        next Measurement (strating the next Report in the
        Study series) MUST be numbered with the value of
        this object as one."
    ::= { reportSampledCurrentReportsEntry 1 }

reportSampledCurrentMeasurementValue OBJECT-TYPE
    SYNTAX      Integer32 (1..2147483647)
    MAX-ACCESS  read-only

```

```

STATUS      current
DESCRIPTION
    "A single measurement for this Study for
    this Current Report.  The objects identifying
    the measurement MUST resolve to type Integer32
    (i.e., Integer32, Counter, Gauge, or TimeTicks).
    to be used for performance monitoring on this device."
 ::= { reportSampledCurrentReportsEntry 2 }

reportSampledCurrentMeasurementTime OBJECT-TYPE
    SYNTAX      sysUpTime
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The systemUpTime of the device on which the
        measurement was made for this Measurement."
    ::= { reportSampledCurrentReportsEntry 3 }

reportSampledCurrentMeasurementStatus OBJECT-TYPE
    SYNTAX INTEGER {
        valueNotAvailable(1),
        valuePositive(2),
        valueNegative(3)
    }
    MAX-ACCESS read-only
    STATUS      current
    DESCRIPTION
        "This object indicates the validity and sign of the
        data in the associated value recorded by the
        reportSampledCurrentMeasurementValue object."
    ::= { reportSampledCurrentReportsEntry 4 }

--
-- Historical Reports Table
--
reportSampledHistoricalReportsTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF ReportSampledHistoricalReportsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table archiving non-active Reports for each
        defined Study up to a maximum number of Reports
        per Study."
    ::= { reportSampledMibObjects 4 }

reportSampledHistoricalReportsEntry OBJECT-TYPE
    SYNTAX      ReportSampledHistoricalReportsEntry

```

```

MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
    "A list of entries storing the measurements from
    Completed Reports. Once an active, current Report
    completes (when the value of the associated
    reportSampledNumberOfMeasurementsForCurrentReport
    equals the value of the associated
    reportSampledStudyNumberReportMeasurements), the
    agent MUST move the Report's data from the
    reportSampledCurrentReportsTable to this
    reportSampledHistoricalReportTable."
REFERENCE
    " TBD. "
INDEX      { reportSampledStudyIndex,
              reportSampledHistoricalReportIndex,
              reportSampledHistoricalMeasurementIndex }
 ::= { reportSampledHistoricalReportsTable 1 }

ReportSampledHistoricalReportsEntry ::= SEQUENCE {
    reportSampledHistoricalReportIndex      Integer32,
    reportSampledHistoricalMeasurementIndex Integer32,
    reportSampledHistoricalMeasurementValue Integer32,
    reportSampledHistoricalMeasurementTime  sysUpTime,
    reportSampledHistoricalMeasurementStatus INTEGER
}

reportSampledHistoricalReportIndex OBJECT-TYPE
    SYNTAX      Integer32 (1..2147483647)
    MAX-ACCESS not-accessible
    STATUS      current
    DESCRIPTION
        "An index that uniquely identifies the particular Report
        archived in this table for the specific Study
        (identified by the reportSampledStudyIndex).
        ::= { reportSampledHistoricalReportsEntry 1 }

reportSampledHistoricalMeasurementIndex OBJECT-TYPE
    SYNTAX      Integer32 (1..2147483647)
    MAX-ACCESS read-only
    STATUS      current
    DESCRIPTION
        "An index that uniquely identifies a Measurement for
        a specific Report archived in this table for a
        Specific Study."
        ::= { reportSampledHistoricalReportsEntry 2 }

reportSampledHistoricalReportsValue OBJECT-TYPE

```



```
SYNTAX      Integer32(1..2147483647)
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "A single measurement for this Study for
    this Completed Report.  The objects identifying
    the measurement MUST resolve to type Integer32
    (i.e., Integer32, Counter, Gauge, or TimeTicks).
    to be used for performance monitoring on this device."
 ::= { reportSampledHistoriclReportsEntry 3 }

reportSampledHistoricalMeasurementTime OBJECT-TYPE
    SYNTAX      sysUpTime
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The systemUpTime of the device on which the
        measurement was made for this Measurement."
    ::= { reportSampledHistoriclReportsEntry 5 }

reportSampledHistoricalMeasurementStatus OBJECT-TYPE
    SYNTAX      INTEGER {
        valueNotAvailable(1),
        valuePositive(2),
        valueNegative(3)
    }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This object indicates the validity and sign of the
        data in the associated value recorded by the
        reportSampledHistoricalMeasurementValue object."
    ::= { reportSampledHistoriclReportsEntry 5 }

--
-- Notifications
--

-- The following notification objects to define issues with making
-- and storing measurements.

-- Actions which report data, i.e., Reports, are to be handled by
-- the dismanEventMIB module.

reportSampledNotificationObjects OBJECT IDENTIFIER
    ::= {reportSampledMibNotifications 0}
reportSampledNotificationControl OBJECT IDENTIFIER
```

```

::= {reportSampledMibNotifications 1}

--
-- reportSampledNotificationObjects
--
reportSampledDataCollectionFailure NOTIFICATION-TYPE
    OBJECTS      { reportSampledStudyOwner, -- The entity that
                  reportSampledStudyName,   -- configured this Study
                  reportSampledStudyOid     -- The name of the Study
                  }                         -- that is failing to
                  -- collect measurement data
    STATUS       current
    DESCRIPTION   "The reportSampledDataCollectionFailure is a notification
                  sent when the number of consecutive measurement failures
                  within a Current Report, as indicated by consecutive values
                  of the reportSampledCurrentMeasurementStatus being set
                  to 'valueNotAvailable(1)', exceeds the value of the
                  threshold value defined in the
                  reportSampledDataCollectionFailureThreshold object."
    ::= { reportSampledNotificationObjects 1 }

--
-- nhdpNotificationsControl
--
reportSampledDataCollectionFailureThreshold OBJECT-TYPE
    SYNTAX       Integer32 (1..255)
    UNITS        "count"
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION   "A threshold value for the number of
                  consecutive measurement failures within a
                  Current Report as indicated by consecutive values
                  of the reportSampledCurrentMeasurementStatus being
                  being set to 'valueNotAvailable(1)' which
                  exceed the value of this threshold. A value of
                  '255' for this threshold indicates that the
                  reportSampledDataCollectionFailure notification
                  is never to be sent. "
    DEFVAL       { 10 }
    ::= { nhdpNotificationsControl 1 }

```

```
--
-- Compliance Statements
--

-- Mandatory compliance for the reportSampledMIB module will
-- include all objects defined within the module.
reportSampledCompliances OBJECT IDENTIFIER
    ::= { reportSampledMIBConformance 1 }
reportSampledMIBGroups OBJECT IDENTIFIER
    ::= { reportSampledMIBConformance 2 }

reportSampledCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION "The reportSampled basic implementation requirements
        for managed network entities that implement
        the REPORT Sampled process."
    MODULE -- this module
    MANDATORY-GROUPS { reportSampledLocalGroup }
    ::= { reportSampledCompliances 1 }

--
-- Units of Conformance
--

reportSampledLocalGroup OBJECT-GROUP
    OBJECTS {
        reportSampledStudyOwner,
        reportSampledStudyName,
        reportSampledStudyOid,
        reportSampledStudySamplingInterval,
        reportSampledStudyNumberReportMeasurements,
        reportSampledStudyMaximumNumberOfHistoricalReports,
        reportSampledStudyEntryStatus,

        reportSampledNumberOfCurrentReport,
        reportSampledNumberOfMeasurementsForCurrentReport,

        reportSampledCurrentMeasurementValue,
        reportSampledCurrentMeasurementTime,
        reportSampledCurrentMeasurementStatus,

        reportSampledHistoricalMeasurementValue,
        reportSampledHistoricalMeasurementTime,
        reportSampledHistoricalMeasurementStatus,

        reportSampledDataCollectionFailure,
        reportSampledDataCollectionFailureThreshold
    }
}
```

```
STATUS    current
DESCRIPTION
    "The basic set of objects in thie reportSampledMIB module
    to be implemented in order to meet the minimal compliance
    conditions."
::= { reportSampledMIBGroups 1 }

END
```

## 8. Security Considerations

This reportSampledMIB module defines a capability where the local device may poll other MIB modules on the device to collect performance data. These capabilities defined within the reportSampledMIB module are control-able by a network management application through SNMP. As such, a network management application could potentially use the reportSampledMIB module as a mechanism to implement a limited Distributed Denial-of-Service (DDoS) attack against remote devices by overloading their SNMP processing. Care should be taken to secure access to the reportSampledMIB module agent. Specifically, access control mechanisms and authentication mechanisms (via SNMPv3) should always be used for SNMP SET operations. Further, some objects may contain data deemed sensitive and authentication and encryption mechanisms (via SNMPv3) should be used for SNMP GET operations.

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations.

These are the tables and objects and their sensitivity/vulnerability:

- o The reportSampledControlTable is a writable table whose columnar objects are read-create. The following objects with MAX ACCESS of read-create and their security sensitivities are:
  - o
    - \* reportSampledStudyOwner - this object is an administrative string which identifies the the owner of the Study.
    - \* reportSampledStudyName - this object is an administrative string which gives a name to the Study defined by the objects in this table.

- \* reportSampledStudyOid - this object identifies the the Object ID from another MIB module resident on this local device which forms the measurement basis for the Study. Depending upon the object and the associated process responsible for its maintenance, polling this object too frequently may place an undo burden on the process resulting in diminishing its overall ability to perform its operation.
- \* reportSampledStudySamplingInterval - this object identifies the time interval being sampling events. If set too low, the device may not be able to sample the object on remote devices fast enough to satisfy the requested interval. Further, setting this value too low could be used to overwhelm the processing capabilities of the remote agent, resulting in a form of Denial-of-Service (DoS) attack.
- \* reportSampledStudyNumberReportsMeasurements - this object identifies the requested number of measurements (and associated storage/memory) for each identified object for each Study instance. As such, this related to the total device memory necessary to hold the collected data for the identified reports. The device must determine whether it has the necessary storage. If not, the device can reject this value when requested by returning the .... to protect itself against memory overruns.
- \* reportSampledStudyMaximumNumberOfHistoricalReports - this object identifies the requested number of Reports for each identified object for each Study instance. As such, this related to the total device memory necessary to hold the collected data for the identified reports. The device must determine whether it has the necessary storage. If not, the device can reject this value when requested by returning the .... to protect itself against memory overruns.
- \* reportSampledStudyEntryStatus - this is the RowStatus object controlling the configuration of this table row.
- o The reportSampledCurrentReportsStatusTable is a read-only table containing state information. The information in this Table relates to performance measurements on the underlying Mobile Ad-Hoc Network (MANET). As such, some of this information may be deemed sensitive to the overall performance of the MANET and to the organization to which the MANET belongs.
- o The reportSampledCurrentReportsTable is a read-only table containing state information. The information in this Table relates to performance measurements on the underlying Mobile Ad-

Hoc Network (MANET). As such, some of this information may be deemed sensitive to the overall performance of the MANET and to the organization to which the MANET belongs.

- o The reportSampledHistoricalReportsStatusTable is a read-only table containing state information. The information in this Table relates to performance measurements on the underlying Mobile Ad-Hoc Network (MANET). As such, some of this information may be deemed sensitive to the overall performance of the MANET and to the organization to which the MANET belongs.
- o The reportSampledDataCollectionFailure notification object reports information regarding the inability of the reportSampledMIB module from completing its configured reporting mission, including the specific information that the module is failing to collect. To some organizations this information may be deemed sensitive to its mission and may want to protect this information through encryption mechanisms.
- o The reportSampledDataCollectionFailureThreshold object controls the ability of the device to report information regarding the inability of the reportSampledMIB module from completing its configured reporting mission. If set to low, notifications may be produced too frequently causing processor burdens to the agent and the collecting manager, and generating too much management traffic for a bandwidth constrained MANET to support. If set too high, notifications may not be produced frequently enough for the collecting manager to properly track the MANET system performance.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPSec), even then, there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.

It is RECOMMENDED that implementers consider the security features as provided by the SNMPv3 framework (see [RFC3410], section 8), including full support for the SNMPv3 cryptographic mechanisms (for authentication and privacy).

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

## 9. Applicability Statement

This document describes objects for configuring parameters of the remote report generation process on a router or other device. This MIB module, denoted reportSampledMIB module, also reports performance information and notifications. The reportSampledMIB module provides for the remote control, collection and notification of performance reports on devices. As such, it eliminates the need for periodic polling for counters from remote management stations as a means for generating performance reports. This is hoped to greatly reduce management overhead on the MANET. This sections provides some examples of how this MIB module can be used in MANET network deployments. A fuller discussion of MANET network management use cases and challenges will be provided elsewhere.

In the following, two scenarios are identified where this MIB module is useful. This list is not complete and other scenarios are possible.

- o For Mobile vehicles with Low Bandwidth Satellite Link to a Fixed Network Operations Center (NOC) - Here the vehicles carrying the MANET routers carry multiple wireless interfaces, one of which is a relatively low-bandwidth on-the-move satellite connection which interconnects a fix NOC to the nodes of the MANET. Standards-based methods for monitoring and fault management from the fixed NOC are necessary for this deployment option. However, to reduce polling overhead over the low bandwidth communications links, the reportSampledMIB module can be deployed the remote MANET nodes for the remote generation of performance reports.
- o For Fixed NOC and Mobile Local Manager in Larger Vehicles - for larger vehicles, a hierarchical network management arrangement is useful. Centralized network management is performed from a fixed NOC while local management, as provided by this reportSampledMIB module is performed locally from within the vehicles. Standards-based methods for configuration, monitoring, fault and performance management are necessary for this deployment option.

## 10. IANA Considerations

The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER values recorded in the SMI Numbers registry:

Descriptor -----	OBJECT IDENTIFIER value -----
reportSampledMIB	{ experimental XXX }

## 11. Contributors

This MIB document uses the template authored by D. Harrington which is based on contributions from the MIB Doctors, especially Juergen Schoenwaelder, Dave Perkins, C.M.Heard and Randy Presuhn.

## 12. Acknowledgements

We would like to thank Bert Wijnen for pointing out the existence of the usrHistory group within RMON2 and in answering our numerous questions on the usrHistory group. Further, we wish to thank U. Herberg for promoting additions to this MIB through his thoughtful consideration of performance monitoring requirements for other MIBs within the MANET WG, e.g., NHDP and OLSR MIBs.

## 13. References

### 13.1. Normative References

- [RFC2021] Waldbusser, S., "Remote Network Monitoring Management Information Base Version 2 using SMIV2", RFC 2021, January 1997.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIV2)", STD 58, RFC 2578, April 1999.
- [RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIV2", STD 58, RFC 2579, April 1999.
- [RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIV2", STD 58, RFC 2580, April 1999.



- [RFC2981] Kavasseri, R., "Event MIB", RFC 2981, October 2000.
- [RFC2982] Kavasseri, R., "Distributed Management Expression MIB", RFC 2982, October 2000.
- [RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.
- [RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.

### 13.2. Informative References

- [RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

### Appendix A. Change Log

Changes from draft-ietf-manet-report-mib-03 to draft-ietf-manet-report-mib-04 draft.

1. This version 04 of the reportSampledMIB module represents a fairly significant restructuring of the MIB module. This restructuring was necessary in order to align the remote distributed reporting capabilities of this MIB module with the prior DISMAN Working Group (WG) efforts at the IETF.
2. The reportSampledCurrentReportsStatusTable was added to allow a network manager to automate notifications of completed reports from this MIB module using the dismanEventMIB module.
3. The notifications in this reportSampledMIB module were reduced to only notifications related to the operation of the data collections and no longer addressing the reporting of the performance data itself. This later data is now carried in notifications under the control of the dismanEventMIB module.

Changes from draft-ietf-manet-report-mib-01 to draft-ietf-manet-report-mib-02 draft.

1. Stripped the Statistical and the Historical Reports from this draft in order to greatly simplify the initial development and experiments of this MIB module.

2. Changed the RFC category to Experimental.
3. Completed the Security section.
4. Relied upon the AUGMENTS statement to simplify further this MIB definition.

Changes from draft-ietf-manet-report-mib-00 to draft-ietf-manet-report-mib-01 draft.

1. Proposed additions to the statsReports in order to potentially simplify data transmission to management applications.
2. Added some Notification definitions and their relationship to the three reports' structure, i.e., statsReports, sampledReports, and historyReports.
3. In the process of adding notifications for the Sampled and the History reports, decided to restructure the reports from their previously rolling storage model to the fixed interval reporting used all along in the Statistics reporting. This allows the agent to notify the management application that a report has completed and that it is ready to be pulled from the agent storage.
4. Ran MIB through smilint checker and cleaned up all errors and most warnings. A few warnings remain to be addressed.
5. Cleaned up textual material.

Changes from draft-cole-manet-report-mib-02 to draft-ietf-manet-report-mib-00 draft.

1. Major change was the incorporation of the IP address objects associated with all objects of type 'OBJECT IDENTIFIER'. This allows the reportSampledMIB module to exist as a proxy report generation capability on a device separate but in close proximity to the device monitoring the referenced object.
2. Cleaned up the up front text, reducing the repetition with the object descriptions in the MIB.
3. Worked on and added sections discussing the relationship to other MIBs.

Changes from draft-cole-manet-report-mib-01 to draft-cole-manet-report-mib-02 draft.

1. Restructured the MIB somewhat to now offer the three reporting capabilities in increasing order of detail: a) statistical reports, b) sampled reports, and c) historical reports.
2. Renamed the usrHistoryGroup and elements to samplingGroup. This is in line with its actual capabilities.
3. Added a new historyGroup which provides a history of change events.
4. Updated the4 Conformance section to reflect the above changes and additions. But did not yet run smilint to check MIB syntax.

Changes from draft-cole-manet-report-mib-00 to draft-cole-manet-report-mib-01 draft.

1. Added (copied) the usrHistory group from RMON2 into the reportSampledMIB module.
2. Restructured the MIB to account for the inclusion of the reportSampledMibObjects.
3. Dropped the reportCurReportsTable as this did not make sense within the context of the reportSampledMIB module.
4. Added the Compliance and Conformance material. Defined several Compliance Groups to all for base implementations of the reportSampledMIB module for only statistical reports, for only historical reports or for both. Allow for enhanced implementations to address higher capacity issues and extension to metric reporting for statistical reporting.
5. Ran the MIB through the smilint checker and in the process corrected numerous typos, omissions, TEXTUAL CONVENTIONS, IMPORTS, etc.
6. Updated main text to reflect changes.

#### Appendix B. Open Issues

This section contains the set of open issues related to the development and design of the reportSampledMIB module. This section will not be present in the final version of the MIB and will be removed once all the open issues have been resolved.

1. Provide references within the REFERENCE clauses in the MIB module.

2. Identify all objects requiring non-volatile storage in their DESCRIPTION clauses.
3. Request an initial review of this MIB module by a MIB Doctor familiar with the work of the DISMAN WG, preferably an author of the dismanEventMIB module.

#### Appendix C.

```
*****
* Note to the RFC Editor (to be removed prior to publication) *
*
* 1) The reference to RFCXXXX within the DESCRIPTION clauses *
* of the MIB module point to this draft and are to be *
* assigned by the RFC Editor. *
*
* 2) The reference to RFCXXX2 throughout this document point *
* to the current draft-ietf-manet-report-xx.txt. This *
* need to be replaced with the XXX RFC number. *
*
*****
```

#### Authors' Addresses

Robert G. Cole  
US Army CERDEC  
6010 Frankford Road  
Aberdeen Proving Ground, Maryland 21005  
USA

Phone: +1 443 395 8744  
EMail: robert.g.cole@us.army.mil

Joseph Macker  
Naval Research Laboratory  
Washington, D.C. 20375  
USA

EMail: macker@itd.nrl.navy.mil

Internet-Draft

The REPORT-SAMPLED-MIB

December 2014

Andy Bierman  
YumaWorks, Inc.  
Redwood City, CA 94065

EMail: [andy@yumaworks.com](mailto:andy@yumaworks.com)



Internet Engineering Task Force  
Internet-Draft  
Intended status: Experimental  
Expires: February 13, 2015

R. Cole  
US Army CERDEC  
J. Macker  
B. Adamson  
Naval Research Laboratory  
August 12, 2014

Definition of Managed Objects for the Manet Simplified Multicast  
Framework Relay Set Process  
draft-ietf-manet-smf-mib-13

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes objects for configuring aspects of the Simplified Multicast Forwarding (SMF) process for Mobile Ad-Hoc Networks (MANETs). The SMF-MIB module also reports state information, performance information, and notifications. In addition to configuration, the additional state and performance information is useful to operators troubleshooting multicast forwarding problems.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

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

This Internet-Draft will expire on February 13, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. The Internet-Standard Management Framework . . . . .	3
3. Conventions . . . . .	3
4. Overview . . . . .	3
4.1. SMF Management Model . . . . .	4
4.2. Terms . . . . .	5
5. Structure of the MIB Module . . . . .	5
5.1. Textual Conventions . . . . .	6
5.2. The Capabilities Group . . . . .	6
5.3. The Configuration Group . . . . .	7
5.4. The State Group . . . . .	7
5.5. The Performance Group . . . . .	7
5.6. The Notifications Group . . . . .	8
5.7. Tables and Indexing . . . . .	8
6. Relationship to Other MIB Modules . . . . .	9
6.1. Relationship to the SNMPv2-MIB . . . . .	9
6.2. Relationship to the IP-MIB . . . . .	9
6.3. Relationship to the IPMCAST-MIB . . . . .	10
6.4. MIB modules required for IMPORTS . . . . .	10
6.5. Relationship to the Future RSSA-MIB Moduleless . . . . .	10
7. SMF-MIB Definitions . . . . .	11
8. IANA-SMF-MIB Definitions . . . . .	52
9. Security Considerations . . . . .	56
10. Applicability Statement . . . . .	60
11. IANA Considerations . . . . .	62
12. Contributors . . . . .	63
13. Acknowledgements . . . . .	63
14. References . . . . .	63
14.1. Normative References . . . . .	63
14.2. Informative References . . . . .	64
Appendix A. . . . .	65



## 1. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes objects for configuring aspects of a process implementing Simplified Multicast Forwarding (SMF) [RFC6621] for Mobile Ad-Hoc Networks (MANETs). SMF provides multicast Duplicate Packet Detection (DPD) and supports algorithms for constructing an estimate of a MANET Minimum Connected Dominating Set (MCDS) for efficient multicast forwarding. The SMF-MIB module also reports state information, performance information, and notifications. In addition to configuration, this additional state and performance information is useful to operators troubleshooting multicast forwarding problems.

## 2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIv2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

## 3. Conventions

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 RFC 2119 [RFC2119].

## 4. Overview

SMF provides methods for implementing Duplicate Packet Detection (DPD)-based multicast forwarding with the optional use of Connected Dominating Set (CDS)-based relay sets. The CDS provides a complete connected coverage of the nodes comprising the MANET. The Minimum CDS (MCDS) is the smallest set of MANET nodes (comprising a connected cluster) which cover all the nodes in the cluster with their transmissions. As the density of the MANET nodes increase, the fraction of nodes required in an MCDS decreases. Using the MCDS as a multicast forwarding set then becomes an efficient multicast

mechanism for MANETs.

Various algorithms for the construction of estimates of the MCDS exist. The Simplified Multicast Framework [RFC6621] describes some of these. It further defines various operational modes for a node which is participating in the collective creation of the MCDS estimates. These modes depend upon the set of related MANET routing and discovery protocols and mechanisms in operation in the specific MANET node.

A SMF router's MIB module contains SMF process configuration parameters (e.g. specific CDS algorithm), state information (e.g., current membership in the CDS), performance counters (e.g., packet counters), and notifications.

#### 4.1. SMF Management Model

This section describes the management model for the SMF node process.

Figure 1 (reproduced from Figure 1 of [RFC6621]) shows the relationship between the SMF Relay Set selection algorithm and the related algorithms, processes and protocols running in the MANET nodes. The Relay Set Selection Algorithm (RSSA) can rely upon topology information gotten from the MANET Neighborhood Discovery Protocol (NHDP), from the specific MANET routing protocol running on the node, or from Layer 2 information passed up to the higher layer protocol processes.

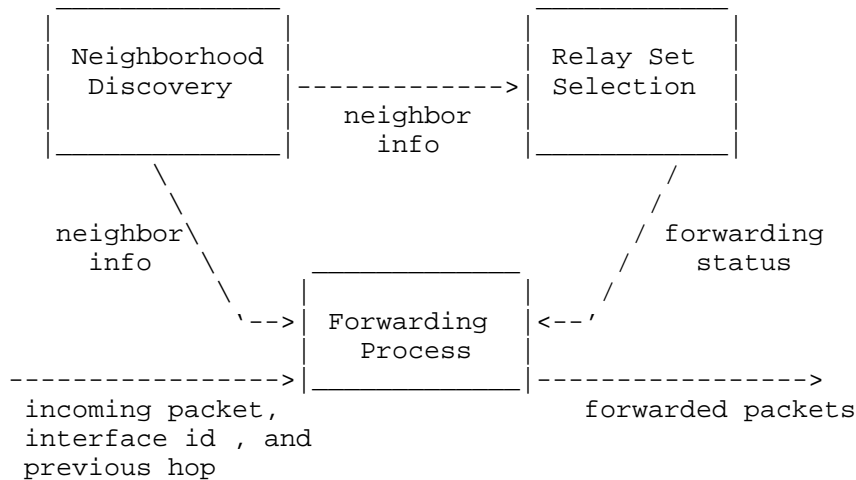


Figure 1: SMF Router Architecture

#### 4.2. Terms

The following definitions apply throughout this document:

- o Configuration Objects - switches, tables, objects which are initialized to default settings or set through the management interfaces such as defined by this MIB module.
- o Tunable Configuration Objects - objects whose values affect timing or attempt bounds on the SMF Relay Set (RS) process.
- o State Objects - automatically generated values which define the current operating state of the SMF RS process in the router.
- o Performance Objects - automatically generated values which help an administrator or automated tool to assess the performance of the CDS multicast process on the router and the overall multicast performance within the MANET routing domain.

#### 5. Structure of the MIB Module

This section presents the structure of the SMF-MIB module. The objects are arranged into the following groups:

- o smfMIBNotifications - defines the notifications associated with the SMF process.

- o smfMIBObjects - defines the objects forming the basis for the SMF-MIB module. These objects are divided up by function into the following groups:
  - \* Capabilities Group - This group contains the SMF objects that the device uses to advertise its local capabilities with respect to, e.g., the supported RSSAs.
  - \* Configuration Group - This group contains the SMF objects that configure specific options that determine the overall operation of the SMF process and the resulting multicast performance.
  - \* State Group - Contains information describing the current state of the SMF process such as the Neighbor Table.
  - \* Performance Group - Contains objects which help to characterize the performance of the SMF process, typically counters for statistical computations.
- o smfMIBConformance - defines two, i.e., minimal and full, conformance implementations for the SMF-MIB module.

#### 5.1. Textual Conventions

The textual conventions defined within the SMF-MIB module are:

- o The SmfStatus is defined within the SMF-MIB module. This contains the current operational status of the SMF process on an interface.

The textual conventions defined for the SMF-MIB module and maintained by IANA are:

- o The IANAsmfOpModeIdTC represents an index that identifies a specific SMF operational mode. This textual convention is maintained by IANA in the IANA-SMF-MIB.
- o The IANAsmfRssaIdTC represents an index that identifies, through reference, a specific RSSA available for operation on the device. This textual convention is maintained by IANA also in the IANA-SMF-MIB.

#### 5.2. The Capabilities Group

The SMF device supports a set of capabilities. The list of capabilities which the device can advertise are:

- o Operational Mode - topology information from NHDP, CDS-aware unicast routing or Cross-layer from Layer 2.
- o SMF RSSA - the specific RSSA operational on the device. Note that configuration, state and performance objects related to a specific RSSA must be defined within a separate MIB module.

### 5.3. The Configuration Group

The SMF device is configured with a set of controls. Some of the prominent configuration controls for the SMF device are:

- o Operational Mode - determines where topology information is derived from, e.g., NHDP, CDS-aware unicast routing or Cross-layer from Layer 2.
- o SMF RSSA - the specific RSSA operational on the device.
- o Duplicate Packet detection for IPv4 - Identification-based or Hash-based DPD.
- o Duplicate Packet detection for IPv6 - Identification-based or Hash-based DPD.
- o SMF Type Message TLV - if NHDP mode is selected, then the SMF Type Message TLV MAY be included in the NHDP exchanges.
- o SMF Address Block TLV - if NHDP mode is selected, then the SMF Address Block TLV SHOULD be included in the NHDP exchanges.
- o SMF Address Forwarding Table - a table identifying configured multicast addresses to be forwarded by the SMF process.

### 5.4. The State Group

The State sub-tree reports current state information, e.g.,

- o Node RSSA State - identifies whether the node is currently in or out of the Relay Set.
- o Neighbors Table - a table containing current one-hop neighbors and their operational RSSA.

### 5.5. The Performance Group

The Performance sub-tree reports primarily counters that relate to SMF RSSA performance. The SMF performance counters consists of per node and per interface objects:

- o Total multicast packets received.
- o Total multicast packets forwarded.
- o Total duplicate multicast packets detected.
- o Per interface statistics table with the following entries:
  - \* Multicast packets received.
  - \* Multicast packets forwarded.
  - \* Duplicate multicast packets detected.

#### 5.6. The Notifications Group

The Notifications Sub-tree contains the list of notifications supported within the SMF-MIB module and their intended purpose and utility.

#### 5.7. Tables and Indexing

The SMF-MIB module contains a number of tables which record data related to:

- o configuration and operation of packet forwarding on the local router,
- o configuration and operation of local MANET interfaces on the router, and
- o configuration and operation of various RSSA algorithms for packet forwarding.

The SMF-MIB module's tables are indexed via the following constructs:

- o `smfCapabilitiesIndex` - the index identifying the combination of SMF mode and SMF RSSA available on this device.
- o `smfCfgAddrForwardingIndex` - the index to configured multicast addresses lists which are forwarded by the SMF process.
- o `smfCfgIfIndex` - the `IfIndex` of the interface on the local router on which SMF is configured.
- o `smfStateNeighborIpAddressType`, `smfStateNeighborIpAddress`, and `smfStateNeighborPrefixLen` - the interface index set of specific one-hop neighbor nodes to this local router.

These tables and their associated indexing are:

- o smfCapabilitiesTable - identifies the resident set of (SMF Operational Modes, SMF RSSA algorithms) available on this router. This table has 'INDEX { smfCapabilitiesIndex }'.
- o smfCfgAddrForwardingTable - contains information on multicast addresses which are to be forwarded by the SMF process on this device. This table has 'INDEX { smfCfgAddrForwardingIndex }'.
- o smfCfgInterfaceTable - describes the SMF interfaces on this device that are participating in the SMF packet forwarding process. This table has 'INDEX { smfCfgIfIndex }'.
- o smfStateNeighborTable - describes the current neighbor nodes, their addresses and the SMF RSSA and the interface on which they can be reached. This table has 'INDEX { smfStateNeighborIpAddressType, smfStateNeighborIpAddress, smfStateNeighborPrefixLen }'.
- o smfPerfIpv4InterfacePerfTable - contains the IPv4 related SMF statistics per each SMF interface on this device. This table has 'INDEX { smfCfgIfIndex }'.
- o smfPerfIpv6InterfacePerfTable - contains the IPv6 related SMF statistics per each SMF interface on this device. This table has 'INDEX { smfCfgIfIndex }'.

## 6. Relationship to Other MIB Modules

### 6.1. Relationship to the SNMPv2-MIB

The 'system' group in the SNMPv2-MIB module [RFC3418] is defined as being mandatory for all systems, and the objects apply to the entity as a whole. The 'system' group provides identification of the management entity and certain other system-wide data. The SMF-MIB module does not duplicate those objects.

### 6.2. Relationship to the IP-MIB

It is an expectation that SMF devices will implement the standard IP-MIB module [RFC4293]. Exactly how to integrate SMF packet handling and management into the standard IP-MIB module management are part of the experiment.

The SMF-MIB module counters within the smfPerformanceGroup count packets handled by the system and interface local SMF process (as discussed above). Not all IP (unicast and multicast) packets on a

device interface are handled by the SMF process. So the counters are tracking different packet streams in the IP-MIB and SMF-MIB modules.

#### 6.3. Relationship to the IPMCAST-MIB

The `smfCfgAddrForwardingTable` is essentially a filter table (if populated) that identifies addresses/packets to be forwarded via the local SMF flooding process. The RFC 5132 IP Multicast MIB module [RFC5132] manages objects related to standard IP multicast, which could be running in parallel to SMF on the device.

RFC 5132 manages traditional IP-based multicast (based upon multicast routing mechanisms). The SMF-MIB module provides management for a MANET subnet-based flooding mechanism which, may be used for multicast transport (through SMF broadcast) depending upon the MANET dynamics and other factors regarding the MANET subnet. Further, they may co-exist in certain MANET deployments using the `smfCfgAddrForwardingTable` to hand certain IP multicast addresses to the SMF process and other IP multicast packets to be forwarded by other IP routed-based multicast mechanisms. SMF and the associated SMF-MIB module are experimental and these are some of the experiments to be had with SMF and the SMF-MIB module.

#### 6.4. MIB modules required for IMPORTS

The textual conventions imported for use in the SMF-MIB module are as follows. The `MODULE-IDENTITY`, `OBJECT-TYPE`, `NOTIFICATION-TYPE`, `Counter32`, `Unsigned32`, `Integer32` and `mib-2` textual conventions are imported from RFC 2578 [RFC2578]. The `TEXTUAL-CONVENTION`, `RowStatus` and `TruthValue` textual conventions are imported from RFC 2579 [RFC2579]. The `MODULE-COMPLIANCE`, `OBJECT-GROUP` and `NOTIFICATION-GROUP` textual conventions are imported from RFC 2580 [RFC2580]. The `InterfaceIndexOrZero` textual convention is imported from RFC 2863 [RFC2863]. The `SnmpAdminString` textual convention is imported from RFC 3411 [RFC3411]. The `InetAddress`, `InetAddressType` and `InetAddressPrefixLength` textual conventions are imported from RFC 4001 [RFC4001].

#### 6.5. Relationship to the Future RSSA-MIB Modules

In a sense, the SMF-MIB module is a general front-end to a set of, yet to be developed, RSSA-specific MIB modules. These RSSA-specific MIB modules will define the objects for the configuration, state, performance and notification required for the operation of these specific RSSAs. The SMF-MIB module Capabilities Group allows the remote management station the ability to query the router to discover the set of supported RSSAs.



## 7. SMF-MIB Definitions

```
SMF-MIB DEFINITIONS ::= BEGIN

IMPORTS

    MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE,
    Counter32, Integer32, TimeTicks, experimental
        FROM SNMPv2-SMI
        -- [RFC2578]

    TEXTUAL-CONVENTION, RowStatus, TruthValue
        FROM SNMPv2-TC
        -- [RFC2579]

    MODULE-COMPLIANCE, OBJECT-GROUP,
    NOTIFICATION-GROUP
        FROM SNMPv2-CONF
        -- [RFC2580]

    InterfaceIndexOrZero, ifName
        FROM IF-MIB
        -- [RFC2863]

    SnmpAdminString
        FROM SNMP-FRAMEWORK-MIB
        -- [RFC3411]

    InetAddress, InetAddressType,
    InetAddressPrefixLength
        FROM INET-ADDRESS-MIB
        -- [RFC4001]

    IANA-smfOpModeIdTC
        FROM IANA-SMF-MIB

    IANA-smfRssaIdTC
        FROM IANA-SMF-MIB
;

smfMIB MODULE-IDENTITY
    LAST-UPDATED "201408121300Z" -- August 12, 2014
    ORGANIZATION "IETF MANET Working Group"
    CONTACT-INFO
        "WG E-Mail: manet@ietf.org

        WG Chairs: sratliff@cisco.com
                  jmacker@nrl.navy.mil

        Editors:  Robert G. Cole
                  US Army CERDEC
```

Space and Terrestrial Communications  
6010 Frankford Road  
Aberdeen Proving Ground, MD 21005  
USA  
+1 443 395-8744  
robert.g.cole@us.army.mil

Joseph Macker  
Naval Research Laboratory  
Washington, D.C. 20375  
USA  
macker@itd.nrl.navy.mil

Brian Adamson  
Naval Research Laboratory  
Washington, D.C. 20375  
USA  
adamson@itd.nrl.navy.mil"

## DESCRIPTION

"This MIB module contains managed object definitions for  
the Manet SMF RSSA process defined in:

Macker, J.(ed.),  
Simplified Multicast Forwarding, RFC 6621,  
May 2012.

Copyright (C) The IETF Trust (2014). This version  
of this MIB module is part of RFC xxxx; see the RFC  
itself for full legal notices."

## -- Revision History

REVISION "201408121300Z" -- August 12, 2014

## DESCRIPTION

"The first version of this MIB module,  
published as RFC xxxx.  
"

-- RFC-Editor assigns xxxx

::= { experimental xxxx } -- to be assigned by IANA

--

-- TEXTUAL CONVENTIONS

--

SmfStatus ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

```
"An indication of the operability of a SMF
function or feature.  For example, the status
of an interface: 'enabled' indicates that
this interface is performing SMF functions,
and 'disabled' indicates that it is not.
Similarly for the status of the device:
'enabled' indicates that the device has
enabled the SMF functions on the device and
'disabled' means that the device and all interfaces
have disabled all SMF functions."
SYNTAX  INTEGER {
    enabled (1),
    disabled (2)
}

--
-- Top-Level Object Identifier Assignments
--

smfMIBNotifications OBJECT IDENTIFIER ::= { smfMIB 0 }
smfMIBObjects        OBJECT IDENTIFIER ::= { smfMIB 1 }
smfMIBConformance    OBJECT IDENTIFIER ::= { smfMIB 2 }

--
-- smfMIBObjects Assignments:
--     smfCapabilitiesGroup - 1
--     smfConfigurationGroup - 2
--     smfStateGroup - 3
--     smfPerformanceGroup - 4
--
--
-- smfCapabilitiesGroup
--
--     This group contains the SMF objects that identify specific
--     capabilities within this device related to SMF functions.
--
smfCapabilitiesGroup OBJECT IDENTIFIER ::= { smfMIBObjects 1 }

--
-- SMF Capabilities Table
--

smfCapabilitiesTable OBJECT-TYPE
```

SYNTAX           SEQUENCE OF SmfCapabilitiesEntry  
 MAX-ACCESS   not-accessible  
 STATUS        current  
 DESCRIPTION  
     "The smfCapabilitiesTable identifies the  
     resident set of SMF Operational Modes and  
     RSSA combinations that can run on this  
     forwarder."

REFERENCE  
     "See Section 7.2. 'Reduced Relay Set Forwarding',  
     Section 8.1.1. 'SMF Message TLV Type', and  
     the Appendices A, B and C in  
     RFC 6621 - Simplified Multicast Forwarding  
     (SMF), Macker, J., May 2012."  
 ::= { smfCapabilitiesGroup 1 }

smfCapabilitiesEntry OBJECT-TYPE  
 SYNTAX       SmfCapabilitiesEntry  
 MAX-ACCESS   not-accessible  
 STATUS       current  
 DESCRIPTION  
     "Information about a particular operational  
     mode and RSSA combination."  
 INDEX        { smfCapabilitiesIndex }  
 ::= { smfCapabilitiesTable 1 }

SmfCapabilitiesEntry ::= SEQUENCE {  
     smfCapabilitiesIndex                   Integer32,  
     smfCapabilitiesOpModeID               IANA-smfOpModeIdTC,  
     smfCapabilitiesRssaID                 IANA-smfRssaIdTC  
 }

smfCapabilitiesIndex       OBJECT-TYPE  
 SYNTAX       Integer32 (1..2147483647)  
 MAX-ACCESS   not-accessible  
 STATUS       current  
 DESCRIPTION  
     "The index for this entry; a unique value,  
     greater than zero, for each combination of  
     a particular operational mode and RSSA  
     algorithm available on this device.  
     It is recommended that values are assigned  
     contiguously starting from 1.  
  
     Rows in this table are automatically  
     populated by the entity's management system  
     on initialization.

By default, the agent should support at least the Classical Flooding 'cF' algorithm. All compliant SMF forwarders must support Classical Flooding. Hence, the first entry in this table MUST exist and MUST be defined as:

```
smfCapabilitiesIndex i '1'
smfCapabilitiesOpModeID i 'cfOnly(1)'
smfCapabilitiesRssaID i 'cF(1)'
```

The value for each combination MUST remain constant at least from one re-initialization of the entity's management system to the next re-initialization."

```
::= { smfCapabilitiesEntry 1 }
```

```
smfCapabilitiesOpModeID      OBJECT-TYPE
    SYNTAX      IANAsmfOpModeIdTC
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "This object identifies
         the particular operational mode for this device."
    ::= { smfCapabilitiesEntry 2 }
```

```
smfCapabilitiesRssaID      OBJECT-TYPE
    SYNTAX      IANAsmfRssaIdTC
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "This object identifies
         the particular RSSA algorithm in this MIB
         module. Example RSSAs are found in the
         appendix of RFC 6621."
    REFERENCE
        "See, e.g., Section 8.1.1. 'SMF Message TLV Type',
         and the Appendices A, B and C in
         RFC 6621 - Simplified Multicast Forwarding
         (SMF), Macker, J., May 2012."
    ::= { smfCapabilitiesEntry 3 }
```

```
--
-- smfConfigurationGroup
--
-- This group contains the SMF objects that configure specific
-- options that determine the overall performance and operation
-- of the multicast forwarding process for the router device
```

```
--      and its interfaces.
--

smfConfigurationGroup  OBJECT IDENTIFIER ::= { smfMIBObjects 2 }

smfCfgAdminStatus  OBJECT-TYPE
    SYNTAX      SmfStatus
    MAX-ACCESS   read-write
    STATUS      current
    DESCRIPTION
        "The configured status of the SMF process
        on this device.  'enabled(1)' means that
        SMF is configured to run on this device.
        'disabled(2)' mean that the SMF process
        is configured off.

        Prior to SMF functions being performed over
        specific interfaces, this object must first
        be 'enabled'.  If this object is 'disabled',
        then no SMF functions are being performed on
        the device and all smfCfgIfAdminStatus objects
        MUST also be set to 'disabled'.  When this
        object is changed from 'enabled' to 'disabled'
        by the manager, then all smfCfgIfAdminStatus
        objects MUST also be automatically set to
        'disabled' by the agent.

        The default value for this object SHOULD be
        'enabled'.

        This object is persistent and when written
        the entity SHOULD save the change to
        non-volatile storage."
    DEFVAL { enabled }
 ::= { smfConfigurationGroup 1 }

smfCfgSmfSysUpTime OBJECT-TYPE
    SYNTAX      TimeTicks
    MAX-ACCESS   read-only
    STATUS      current
    DESCRIPTION
        "The time (in hundredths of a second) since the
        system SMF process was last re-initialized.
        The SMF process is re-initialized when the
        value of the 'smfCfgAdminStatus' object
        transitions to 'enabled' from either a prior
        value of 'disabled' or upon initialization
        of this device."
```

```
::= { smfConfigurationGroup 2 }

smfCfgRouterIDAddrType OBJECT-TYPE
    SYNTAX      InetAddressType { ipv4(1), ipv6(2) }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The address type of the address used for
        SMF ID of this router as specified
        in the 'smfCfgRouterID' next.

        Only the values ipv4(1) and ipv6(2)
        are supported.

        This object is persistent and when written
        the entity SHOULD save the change to
        non-volatile storage."
    DEFVAL { ipv4 }
::= { smfConfigurationGroup 3 }

smfCfgRouterID OBJECT-TYPE
    SYNTAX      InetAddress (SIZE(4|16))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The IP address used as the SMF router ID.
        This can be set by the management station.
        If not explicitly set, then the device
        SHOULD select a routable IP address
        assigned to this router for use as
        the 'smfCfgRouterID'.

        The smfCfgRouterID is a logical identification
        that MUST be consistent across interoperable
        SMF neighborhoods and it is RECOMMENDED to be
        chosen as the numerically largest address
        contained in a node's 'Neighbor Address List'
        as defined in NHDP. A smfCfgRouterID MUST be
        unique within the scope of the operating
        MANET network regardless of the method used
        for selecting it.

        This object is persistent and when written
        the entity SHOULD save the change to
        non-volatile storage."
    REFERENCE
        "See, e.g.,
```

Appendix Section A.1. 'E-CDS Relay Set  
Selection Overview' and

Appendix Section C.1. 'MPR-CDS Relay  
Set Selection Overview'

in RFC 6621 - Simplified Multicast Forwarding  
(SMF), Macker, J., May 2012."

::= { smfConfigurationGroup 4 }

smfCfgOperationalMode OBJECT-TYPE

SYNTAX Integer32 (1..2147483647)

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The SMF RSS node operational mode and  
RSSA algorithm combination active on this  
local forwarder. This object is defined  
to be equal to the smfCapabilitiesIndex  
which identifies the specific active  
operational mode and RSSA.

The default value for this object is  
'1' which corresponds to:

smfCapabilitiesOpModeID i 'cfOnly(1)'

smfCapabilitiesRssaID i 'cF(1)'

This object is persistent and when written  
the entity SHOULD save the change to  
non-volatile storage."

REFERENCE

"See Section 7.2. 'Reduced Relay Set Forwarding',  
and the Appendices A, B and C in  
RFC 6621 - Simplified Multicast Forwarding  
(SMF), Macker, J., May 2012."

DEFVAL { 1 }

::= { smfConfigurationGroup 5 }

smfCfgRssaMember OBJECT-TYPE

SYNTAX INTEGER {  
potential(1),  
always(2),  
never(3)  
}

MAX-ACCESS read-write

STATUS current

DESCRIPTION



"The RSSA downselects a set of forwarders for multicast forwarding. Sometimes it is useful to force an agent to be included or excluded from the resulting RSS. This object is a switch to allow for this behavior.

The value 'potential(1)' allows the selected RSSA to determine if this agent is included or excluded from the RSS.

The value 'always(2)' forces the selected RSSA include this agent in the RSS.

The value 'never(3)' forces the selected RSSA to exclude this agent from the RSS.

The default setting for this object is 'potential(1)'. Other settings could pose operational risks under certain conditions.

This object is persistent and when written the entity SHOULD save the change to non-volatile storage."

#### REFERENCE

"See Section 7. 'Relay Set Selection' in RFC 6621 - Simplified Multicast Forwarding (SMF), Macker, J., May 2012."

DEFVAL { potential }

::= { smfConfigurationGroup 6 }

smfCfgIpv4Dpd OBJECT-TYPE

SYNTAX INTEGER {  
hashBased(1),  
identificationBased(2)  
}

MAX-ACCESS read-write

STATUS current

#### DESCRIPTION

"The current method for IPv4 duplicate packet detection.

The value 'hashBased(1)' indicates that the routers duplicate packet detection is based upon comparing a hash over the packet fields. This is the default setting for this object.

The value 'identificationBased(2)' indicates that the duplicate packet

detection relies upon header information in the multicast packets to identify previously received packets.

This object is persistent and when written the entity SHOULD save the change to non-volatile storage."

#### REFERENCE

"See Section 6.2. 'IPv4 Duplicate Packet Detection' in RFC 6621 - Simplified Multicast Forwarding (SMF), Macker, J., May 2012."

```
DEFVAL { hashBased }
 ::= { smfConfigurationGroup 7 }

smfCfgIpv6Dpd OBJECT-TYPE
    SYNTAX      INTEGER {
                        hashBased(1),
                        identificationBased(2)
                    }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The current method for IPv6 duplicate packet
        detection.
```

The values indicate the type of method used for duplicate packet detection as described the previous description for the object 'smfCfgIpv4Dpd'.

The default value for this object is 'hashBased(1)'.

This object is persistent and when written the entity SHOULD save the change to non-volatile storage."

#### REFERENCE

"See Section 6.1. 'IPv6 Duplicate Packet Detection' in RFC 6621 - Simplified Multicast Forwarding (SMF), Macker, J., May 2012."

```
DEFVAL { hashBased }
 ::= { smfConfigurationGroup 8 }
```

```
smfCfgMaxPktLifetime OBJECT-TYPE
    SYNTAX      Integer32 (0..65535)
    UNITS       "Seconds"
```

```
MAX-ACCESS    read-write
STATUS        current
DESCRIPTION
    "The estimate of the network packet
    traversal time.

    This object is persistent and when written
    the entity SHOULD save the change to
    non-volatile storage."
REFERENCE
    "See Section 6. 'SMF Duplicate Packet
    Detection' in RFC 6621 - Simplified
    Multicast Forwarding (SMF), Macker, J.,
    May 2012."
DEFVAL { 60 }
 ::= { smfConfigurationGroup 9 }

smfCfgDpdEntryMaxLifetime OBJECT-TYPE
    SYNTAX      Integer32 (0..65525)
    UNITS        "Seconds"
    MAX-ACCESS   read-write
    STATUS       current
    DESCRIPTION
        "The maximum lifetime of a cached DPD
        record in the local device storage.

        If the memory is running low prior to the
        MaxLifetime being exceeded, the local SMF
        devices should purge the oldest records first.

        This object is persistent and when written
        the entity SHOULD save the change to
        non-volatile storage."
    REFERENCE
        "See Section 6. 'SMF Duplicate Packet
        Detection' in RFC 6621 - Simplified
        Multicast Forwarding (SMF), Macker, J.,
        May 2012."
    DEFVAL { 600 }
 ::= { smfConfigurationGroup 10 }

--
-- Configuration of messages to be included in
-- NHDP message exchanges in support of SMF
-- operations.
--
```

smfCfgNhdpRssaMesgTLVIncluded OBJECT-TYPE  
SYNTAX TruthValue  
MAX-ACCESS read-write  
STATUS current  
DESCRIPTION  
"Indicates whether the associated NHDP messages  
include the RSSA Message TLV, or not. This  
is an optional SMF operational setting.  
The value 'true(1)' indicates that this TLV is  
included; the value 'false(2)' indicates that it  
is not included.  
  
It is RECOMMENDED that the RSSA Message TLV  
be included in the NHDP messages.  
  
This object is persistent and when written  
the entity SHOULD save the change to  
non-volatile storage."  
REFERENCE  
"See Section 8.1.1. 'SMF Message TLV Type' in  
RFC 6621 - Simplified Multicast Forwarding  
(SMF), Macker, J., May 2012."  
DEFVAL { true }  
 ::= { smfConfigurationGroup 11 }

smfCfgNhdpRssaAddrBlockTLVIncluded OBJECT-TYPE  
SYNTAX TruthValue  
MAX-ACCESS read-write  
STATUS current  
DESCRIPTION  
"Indicates whether the associated NHDP messages  
include the RSSA Address Block TLV, or not.  
This is an optional SMF operational setting.  
The value 'true(1)' indicates that this TLV is  
included; the value 'false(2)' indicates that it  
is not included.  
  
The smfCfgNhdpRssaAddrBlockTLVIncluded is optional  
in all cases as it depends on the existence of  
an address block which may not be present.  
If this SMF device is configured with NHDP,  
then this object SHOULD be set to 'true(1)'.  
  
This object is persistent and when written  
the entity SHOULD save the change to  
non-volatile storage."  
REFERENCE  
"See Section 8.1.2. 'SMF Address Block TLV

```

        Type' in RFC 6621 - Simplified Multicast
        Forwarding (SMF), Macker, J., May 2012."
    DEFVAL { true }
    ::= { smfConfigurationGroup 12 }

```

```

--
-- Table identifying configured multicast addresses to be forwarded.
--

```

```

smfCfgAddrForwardingTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF SmfCfgAddrForwardingEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The smfCfgAddrForwardingTable is essentially a filter
        table (if populated) that identifies addresses/packets
        to be forwarded via the local SMF flooding process.
        The RFC 5132 IP Multicast MIB module manages objects
        related to standard IP multicast, which could be running
        in parallel to SMF on the device.

        RFC 5132 manages traditional IP-based multicast (based
        upon multicast routing mechanisms). The SMF-MIB module
        provides management for a MANET subnet-based flooding
        mechanism which, may be used for multicast transport
        (through SMF broadcast) depending upon the MANET dynamics
        and other factors regarding the MANET subnet. Further,
        they may co-exist in certain MANET deployments
        using the smfCfgAddrForwardingTable to hand certain IP
        multicast addresses to the SMF process and other IP
        multicast packets to be forwarded by other IP
        routed-based multicast mechanisms. SMF and the
        associated SMF-MIB module are experimental and these
        are some of the experiments to be had with SMF and
        the SMF-MIB module.

```

This is the (conceptual) table containing information on multicast addresses which are to be forwarded by the SMF process. This table represents an IP filters table for forwarding (or not) packets based upon their IP multicast address.

The SMF process can be configured to forward only those multicast addresses found within the smfCfgAddrForwardingTable. As such, addresses which are to be forwarded by the SMF process MUST be found within

the address ranges configured within this table, unless this table is empty.

Each row is associated with a range of multicast addresses, and ranges for different rows must be disjoint. Different rows MAY share a common smfCfgAddrForwardingGroupName to administratively associate different rows.

The objects in this table are persistent and when written the entity SHOULD save the change to non-volatile storage."

#### REFERENCE

"See Section 9.1. 'Forwarded Multicast Groups' in RFC 6621 - Simplified Multicast Forwarding (SMF), Macker, J., May 2012."

::= { smfConfigurationGroup 13 }

#### smfCfgAddrForwardingEntry OBJECT-TYPE

SYNTAX SmfCfgAddrForwardingEntry

MAX-ACCESS not-accessible

STATUS current

#### DESCRIPTION

"An entry (conceptual row) containing the information on a particular multicast scope."

INDEX { smfCfgAddrForwardingIndex }

::= { smfCfgAddrForwardingTable 1 }

#### SmfCfgAddrForwardingEntry ::= SEQUENCE {

smfCfgAddrForwardingIndex Integer32,

smfCfgAddrForwardingGroupName SnmpAdminString,

smfCfgAddrForwardingAddrType InetAddressType,

smfCfgAddrForwardingAddress InetAddress,

smfCfgAddrForwardingAddrPrefixLength

InetAddressPrefixLength,

smfCfgAddrForwardingStatus RowStatus

}

#### smfCfgAddrForwardingIndex OBJECT-TYPE

SYNTAX Integer32 (1..2147483647)

MAX-ACCESS not-accessible

STATUS current

#### DESCRIPTION

"This object identifies an unique entry for a forwarding group. The index for this entry is a unique value, greater than zero, for each row. It is recommended that values are assigned contiguously starting from 1."

The value for each row index MUST remain constant from one re-initialization of the entity's management system to the next re-initialization."

```
::= { smfCfgAddrForwardingEntry 1 }
```

smfCfgAddrForwardingGroupName OBJECT-TYPE

SYNTAX SnmpAdminString

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object identifies a group name for a set of row entries in order to administratively associate a set of address ranges.

If there is no group name or this object is otherwise not applicable, then this object contains a zero-length string.

This object is persistent and when written the entity SHOULD save the change to non-volatile storage."

```
::= { smfCfgAddrForwardingEntry 2 }
```

smfCfgAddrForwardingAddrType OBJECT-TYPE

SYNTAX InetAddressType { ipv4(1), ipv6(2) }

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The type of the addresses in the multicast forwarding ranges identified by this table.

Only the values ipv4(1) and ipv6(2) are supported.

This object is persistent and when written the entity SHOULD save the change to non-volatile storage."

```
::= { smfCfgAddrForwardingEntry 3 }
```

smfCfgAddrForwardingAddress OBJECT-TYPE

SYNTAX InetAddress (SIZE(4|16))

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The multicast group address which, when combined with smfCfgAddrForwardingAddrPrefixLength, gives the group prefix for this forwarding range.

The InetAddressType is given by  
smfCfgAddrForwardingAddrType.

This address object is only significant up to  
smfCfgAddrForwardingAddrPrefixLength bits. The  
remaining address bits are set to zero. This is  
especially important for this index field,  
Any non-zero bits would signify an entirely  
different entry.

Legal values correspond to the subset of address  
families for which multicast address allocation  
is supported.

This object is persistent and when written  
the entity SHOULD save the change to  
non-volatile storage."

```
::= { smfCfgAddrForwardingEntry 4 }
```

smfCfgAddrForwardingAddrPrefixLength OBJECT-TYPE

SYNTAX InetAddressPrefixLength

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The length in bits of the mask which, when  
combined with smfCfgAddrForwardingAddress,  
gives the group prefix for this forwarding  
range.

This object is persistent and when written  
the entity SHOULD save the change to  
non-volatile storage."

```
::= { smfCfgAddrForwardingEntry 5 }
```

smfCfgAddrForwardingStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The status of this row, by which new entries may be  
created, or old entries deleted from this table."

```
::= { smfCfgAddrForwardingEntry 6 }
```

```
--  
-- SMF Interfaces Configuration Table  
--
```



**smfCfgInterfaceTable OBJECT-TYPE****SYNTAX** SEQUENCE OF SmfCfgInterfaceEntry**MAX-ACCESS** not-accessible**STATUS** current**DESCRIPTION**

"The SMF Interface Table describes the SMF interfaces that are participating in the SMF packet forwarding process. The ifIndex is from the interfaces group defined in the Interfaces Group MIB module (RFC 2863). As such, this table 'sparse augments' the ifTable specifically when SMF is to be configured to operate over this interface.

A conceptual row in this table exists if and only if either a manager has explicitly created the row or there is an interface on the managed device that automatically supports and runs SMF as part of the device's initialization process.

The manager creates a row in this table by setting rowStatus to 'createAndGo' or 'createAndWait'. Row objects having associated DEFVAL clauses are automatically defined by the agent with these values during row creation, unless the manager explicitly defines these object values during the row creation.

As the smfCfgInterfaceTable sparsely augments the IfTable. Hence,

- + an entry cannot exist in smfCfgInterfaceTable without a corresponding entry in the ifTable.
- + if an entry in the ifTable is removed, the corresponding entry (if it exists) in the smfCfgInterfaceTable MUST be removed.
- + the smfCfgIfStatus can have a value of 'enabled' or 'disabled' independent of the current value of the ifAdminStatus of the corresponding entry in the ifTable.

The values of the objects smfCfgAdminStatus and smfCfgIfAdminStatus reflect the up-down status of the SMF process running on the device and on the specific interfaces respectively. Hence,

- + the value of the smfCfgAdminStatus can be 'enabled' or 'disabled' reflecting the current running status of the SMF process on the device.
- + the value of the smfCfgIfAdminStatus can be 'enabled' or 'disabled' if the value of the smfCfgAdminStatus is set to 'enabled'.
- + if the value of the smfCfgAdminStatus is 'disabled', then the corresponding smfCfgIfAdminStatus objects MUST be set to 'disabled' in the smfCfgInterfaceTable.
- + once the value of the smfCfgAdminStatus changes from 'disabled' to 'enabled', it is up to the management system to make the corresponding changes to the smfCfgIfAdminStatus values back to 'enabled'.

"

## REFERENCE

"RFC 2863 - The Interfaces Group MIB, McCloghrie, K., and F. Kastenholz, June 2000."

```
::= { smfConfigurationGroup 14 }
```

## smfCfgInterfaceEntry OBJECT-TYPE

```
SYNTAX      SmfCfgInterfaceEntry
```

```
MAX-ACCESS  not-accessible
```

```
STATUS      current
```

## DESCRIPTION

"The SMF interface entry describes one SMF interface as indexed by its ifIndex.

The objects in this table are persistent and when written the device SHOULD save the change to non-volatile storage. For further information on the storage behavior for these objects, refer to the description for the smfCfgIfRowStatus object."

```
INDEX { smfCfgIfIndex }
```

```
::= { smfCfgInterfaceTable 1 }
```

```
SmfCfgInterfaceEntry ::=
```

```
SEQUENCE {
    smfCfgIfIndex      InterfaceIndexOrZero,
    smfCfgIfAdminStatus SmfStatus,
    smfCfgIfSmfUpTime  TimeTicks,
    smfCfgIfRowStatus   RowStatus
}
```

smfCfgIfIndex OBJECT-TYPE  
SYNTAX InterfaceIndexOrZero  
MAX-ACCESS not-accessible  
STATUS current  
DESCRIPTION  
    "The ifIndex for this SMF interface. This value  
    MUST correspond to an ifIndex referring  
    to a valid entry in The Interfaces Table.  
    If the manager attempts to create a row  
    for which the ifIndex does not exist on the  
    local device, then the agent SHOULD issue  
    a return value of 'inconsistentValue' and  
    the operation SHOULD fail."  
REFERENCE  
    "RFC 2863 - The Interfaces Group MIB, McCloghrie,  
    K., and F. Kastenholz, June 2000."  
 ::= { smfCfgInterfaceEntry 1 }

smfCfgIfAdminStatus OBJECT-TYPE  
SYNTAX SmfStatus  
MAX-ACCESS read-create  
STATUS current  
DESCRIPTION  
    "The SMF interface's administrative status.  
    The value 'enabled' denotes that the interface  
    is running the SMF forwarding process.  
    The value 'disabled' denotes that the interface is  
    currently external to the SMF forwarding process.  
  
    When the value of the smfCfgAdminStatus is  
    'disabled', then the corresponding smfCfgIfAdminStatus  
    objects MUST be set to 'disabled' in the  
    smfCfgInterfaceTable.  
  
    The default value for this object is 'enabled(1)'.  
  
    This object SHOULD be persistent and when  
    written the device SHOULD save the change to  
    non-volatile storage."  
DEFVAL { enabled }  
 ::= { smfCfgInterfaceEntry 2 }

smfCfgIfSmfUpTime OBJECT-TYPE  
SYNTAX TimeTicks  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
    "The time (in hundredths of a second) since

this interface SMF process was last re-initialized. The interface SMF process is re-initialized when the corresponding 'smfCfgIfRowStatus' object transits to the 'active' state."  
 ::= { smfCfgInterfaceEntry 3 }

smfCfgIfRowStatus OBJECT-TYPE

SYNTAX RowStatus  
MAX-ACCESS read-create  
STATUS current  
DESCRIPTION

"This object permits management of this table by facilitating actions such as row creation, construction, and destruction. The value of this object has no effect on whether other objects in this conceptual row can be modified.

An entry may not exist in the 'active' state unless all objects in the entry have a defined appropriate value. For objects with DEFVAL clauses, the management station does not need to specify the value of these objects in order for the row to transit to the 'active' state; the default value for these objects is used. For objects that do not have DEFVAL clauses, then the network manager MUST specify the value of these objects prior to this row transitioning to the 'active' state.

When this object transitions to 'active', all objects in this row SHOULD be written to non-volatile (stable) storage. Read-create objects in this row MAY be modified. When an object in a row with smfCfgIfRowStatus of 'active' is changed, then the updated value MUST be reflected in SMF and this new object value MUST be written to non-volatile storage.

If this object is not equal to 'active', all associated entries in the smfPerfIpv4InterfacePerfTable and the smfPerfIpv6InterfacePerfTable MUST be deleted."  
 ::= { smfCfgInterfaceEntry 4 }

--  
-- smfStateGroup  
--  
-- Contains information describing the current state of the SMF  
-- process such as the current inclusion in the RS or not.

--

smfStateGroup OBJECT IDENTIFIER ::= { smfMIBObjects 3 }

smfStateNodeRsStatusIncluded OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The current status of the SMF node in the context of the MANETs relay set. A value of 'true(1)' indicates that the node is currently part of the MANET Relay Set. A value of 'false(2)' indicates that the node is currently not part of the MANET Relay Set."

REFERENCE

"See Section 7. 'Relay Set Selection' in RFC 6621 - Simplified Multicast Forwarding (SMF), Macker, J., May 2012."

::= { smfStateGroup 1 }

smfStateDpdMemoryOverflow OBJECT-TYPE

SYNTAX Counter32

UNITS "DPD Records"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of DPD records that had to be flushed to prevent memory overruns for caching of these records. The number of records to be flushed upon a buffer overflow is an implementation specific decision."

There is the potential for a counter discontinuity in this object if the system SMF process had been disabled and later enabled. In order to check for the occurrence of such a discontinuity when monitoring this counter object, it is recommended that the smfCfgSmfSysUpTime object also be monitored."

REFERENCE

"See Section 6. 'SMF Duplicate Packet Detection' in RFC 6621 - Simplified Multicast Forwarding (SMF), Macker, J., May 2012."

::= { smfStateGroup 2 }

--

-- SMF Neighbor Table

--

```

smfStateNeighborTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF SmfStateNeighborEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "The SMF StateNeighborTable describes the
        current one-hop neighbor nodes, their address
        and SMF RSSA and the interface on which
        they can be reached."
    REFERENCE
        "See Section 7. 'SMF Neighborhood Discovery' and
        Section 8.1. 'SMF Relay Algorithm TLV
        Types' in RFC 6621 - Simplified Multicast
        Forwarding (SMF), Macker, J., May 2012."
    ::= { smfStateGroup 3 }

smfStateNeighborEntry OBJECT-TYPE
    SYNTAX      SmfStateNeighborEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "The SMF Neighbor Table contains the
        set of one-hop neighbors, the interface
        they are reachable on and the SMF RSSA
        they are currently running."
    INDEX { smfStateNeighborIpAddressType,
            smfStateNeighborIpAddress,
            smfStateNeighborPrefixLen }
    ::= { smfStateNeighborTable 1 }

SmfStateNeighborEntry ::=
    SEQUENCE {
        smfStateNeighborIpAddressType      InetAddressType,
        smfStateNeighborIpAddress          InetAddress,
        smfStateNeighborPrefixLen           InetAddressPrefixLength,
        smfStateNeighborRSSA                IANASmfRssaIdTC,
        smfStateNeighborNextHopInterface   InterfaceIndexOrZero
    }

smfStateNeighborIpAddressType OBJECT-TYPE
    SYNTAX      InetAddressType { ipv4(1), ipv6(2) }
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "The one-hop neighbor IP address type.

        Only the values 'ipv4(1)' and
        'ipv6(2)' are supported."

```

```
::= { smfStateNeighborEntry 1 }

smfStateNeighborIpAddress OBJECT-TYPE
    SYNTAX      InetAddress (SIZE(4|16))
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
        "The one-hop neighbor Inet IPv4 or IPv6
        address.

        Only IPv4 and IPv6 addresses
        are supported."
::= { smfStateNeighborEntry 2 }

smfStateNeighborPrefixLen OBJECT-TYPE
    SYNTAX      InetAddressPrefixLength
    UNITS        "bits"
    MAX-ACCESS   not-accessible
    STATUS       current
    DESCRIPTION
        "The prefix length. This is a decimal value that
        indicates the number of contiguous, higher-order
        bits of the address that make up the network
        portion of the address."
::= { smfStateNeighborEntry 3 }

smfStateNeighborRSSA OBJECT-TYPE
    SYNTAX      IANA-smfRssaIdTC
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The current RSSA running on the neighbor."
::= { smfStateNeighborEntry 4 }

smfStateNeighborNextHopInterface OBJECT-TYPE
    SYNTAX      InterfaceIndexOrZero
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The interface ifIndex over which the
        neighbor is reachable in one-hop."
::= { smfStateNeighborEntry 6 }

--
-- SMF Performance Group
--
-- Contains objects which help to characterize the
```

```

--      performance of the SMF RSSA process, such as statistics
--      counters. There are two types of SMF RSSA statistics:
--      global counters and per interface counters.
--
--      It is an expectation that SMF devices will
--      implement the standard IP-MIB module RFC4293.
--      Exactly how to integrate SMF packet handling and
--      management into the standard IP-MIB module management
--      are part of the experiment.
--
--      The SMF-MIB module counters within the
--      smfPerformanceGroup count packets handled by the
--      system and interface local SMF process (as discussed
--      above). Not all IP (unicast and multicast) packets
--      on a device interface are handled by the SMF process.
--      So the counters are tracking different packet streams
--      in the IP-MIB and SMF-MIB modules.
--
smfPerformanceGroup  OBJECT IDENTIFIER ::= { smfMIBObjects 4 }

smfPerfGobalGroup  OBJECT IDENTIFIER ::= { smfPerformanceGroup 1 }

--
-- IPv4 packet counters
--

smfPerfIpv4MultiPktsRecvTotal  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "Packets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "A counter of the total number of
        multicast IPv4 packets received by the
        device and delivered to the SMF process.

        There is the potential for a counter discontinuity
        in this object if the system SMF process had been
        disabled and later enabled. In order to check for
        the occurrence of such a discontinuity when monitoring
        this counter object, it is recommended that the
        smfCfgSmfSysUpTime object also be monitored."
    ::= { smfPerfGobalGroup 1 }

smfPerfIpv4MultiPktsForwardedTotal  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "Packets"

```



```

MAX-ACCESS    read-only
STATUS        current
DESCRIPTION
    "A counter of the total number of
    multicast IPv4 packets forwarded by the
    device.

    There is the potential for a counter discontinuity
    in this object if the system SMF process had been
    disabled and later enabled. In order to check for
    the occurrence of such a discontinuity when monitoring
    this counter object, it is recommended that the
    smfCfgSmfSysUpTime object also be monitored."
 ::= { smfPerfGobalGroup 2 }

smfPerfIpv4DuplMultiPktsDetectedTotal  OBJECT-TYPE
SYNTAX          Counter32
UNITS           "Packets"
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "A counter of the total number of duplicate
    multicast IPv4 packets detected by the
    device.

    There is the potential for a counter discontinuity
    in this object if the system SMF process had been
    disabled and later enabled. In order to check for
    the occurrence of such a discontinuity when monitoring
    this counter object, it is recommended that the
    smfCfgSmfSysUpTime object also be monitored."
REFERENCE
    "See Section 6.2. 'IPv4 Duplicate Packet
    Detection' in RFC 6621 - Simplified Multicast
    Forwarding (SMF), Macker, J., May 2012."
 ::= { smfPerfGobalGroup 3 }

smfPerfIpv4DroppedMultiPktsTTLExceededTotal  OBJECT-TYPE
SYNTAX          Counter32
UNITS           "Packets"
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "A counter of the total number of dropped
    multicast IPv4 packets by the
    device due to TTL exceeded.

    There is the potential for a counter discontinuity

```

in this object if the system SMF process had been disabled and later enabled. In order to check for the occurrence of such a discontinuity when monitoring this counter object, it is recommended that the smfCfgSmfSysUpTime object also be monitored."

## REFERENCE

"See Section 5. 'SMF Packet Processing and Forwarding' in RFC 6621 - Simplified Multicast Forwarding (SMF), Macker, J., May 2012."

```
::= { smfPerfGobalGroup 4 }
```

```
smfPerfIpv4TTLLargerThanPreviousTotal OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
UNITS "Packets"
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

## DESCRIPTION

"A counter of the total number of IPv4 packets recieved which have a TTL larger than that of a previously received identical packet.

There is the potential for a counter discontinuity in this object if the system SMF process had been disabled and later enabled. In order to check for the occurrence of such a discontinuity when monitoring this counter object, it is recommended that the smfCfgSmfSysUpTime object also be monitored."

## REFERENCE

"See Section 5. 'SMF Packet Processing and Forwarding' in RFC 6621 - Simplified Multicast Forwarding (SMF), Macker, J., May 2012."

```
::= { smfPerfGobalGroup 5 }
```

```
--
```

```
-- IPv6 packet counters
```

```
--
```

```
smfPerfIpv6MultiPktsRecvTotal OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
UNITS "Packets"
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

## DESCRIPTION

"A counter of the total number of multicast IPv6 packets received by the device and delivered to the SMF process.

There is the potential for a counter discontinuity in this object if the system SMF process had been disabled and later enabled. In order to check for the occurrence of such a discontinuity when monitoring this counter object, it is recommended that the smfCfgSmfSysUpTime object also be monitored."

```
::= { smfPerfGobalGroup 6 }
```

smfPerfIpv6MultiPktsForwardedTotal OBJECT-TYPE

SYNTAX Counter32

UNITS "Packets"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A counter of the total number of multicast IPv6 packets forwarded by the device.

There is the potential for a counter discontinuity in this object if the system SMF process had been disabled and later enabled. In order to check for the occurrence of such a discontinuity when monitoring this counter object, it is recommended that the smfCfgSmfSysUpTime object also be monitored."

```
::= { smfPerfGobalGroup 7 }
```

smfPerfIpv6DuplMultiPktsDetectedTotal OBJECT-TYPE

SYNTAX Counter32

UNITS "Packets"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A counter of the total number of duplicate multicast IPv6 packets detected by the device.

There is the potential for a counter discontinuity in this object if the system SMF process had been disabled and later enabled. In order to check for the occurrence of such a discontinuity when monitoring this counter object, it is recommended that the smfCfgSmfSysUpTime object also be monitored."

REFERENCE

"See Section 6.1. 'IPv6 Duplicate Packet Detection' in RFC 6621 - Simplified Multicast Forwarding (SMF), Macker, J., May 2012."

```
::= { smfPerfGobalGroup 8 }
```

```
smfPerfIpv6DroppedMultiPktsTTLExceededTotal  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "Packets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "A counter of the total number of dropped
        multicast IPv6 packets by the
        device due to TTL exceeded.

        There is the potential for a counter discontinuity
        in this object if the system SMF process had been
        disabled and later enabled.  In order to check for
        the occurrence of such a discontinuity when monitoring
        this counter object, it is recommended that the
        smfCfgSmfSysUpTime object also be monitored."
    REFERENCE
        "See Section 5. 'SMF Packet Processing and
        Forwarding' in RFC 6621 - Simplified
        Multicast Forwarding (SMF), Macker, J.,
        May 2012."
 ::= { smfPerfGobalGroup 9 }

smfPerfIpv6TTLLargerThanPreviousTotal  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "Packets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "A counter of the total number of IPv6 packets
        recieved which have a TTL larger than that
        of a previously recieved identical packet.

        There is the potential for a counter discontinuity
        in this object if the system SMF process had been
        disabled and later enabled.  In order to check for
        the occurrence of such a discontinuity when monitoring
        this counter object, it is recommended that the
        smfCfgSmfSysUpTime object also be monitored."
    REFERENCE
        "See Section 5. 'SMF Packet Processing and
        Forwarding' in RFC 6621 - Simplified Multicast
        Forwarding (SMF), Macker, J., May 2012."
 ::= { smfPerfGobalGroup 10 }

smfPerfIpv6HAVAssistsReqdTotal  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "Packets"
```

```

MAX-ACCESS    read-only
STATUS        current
DESCRIPTION
    "A counter of the total number of IPv6 packets
    received which required the HAV assist for DPD.

    There is the potential for a counter discontinuity
    in this object if the system SMF process had been
    disabled and later enabled. In order to check for
    the occurrence of such a discontinuity when monitoring
    this counter object, it is recommended that the
    smfCfgSmfSysUpTime object also be monitored."
REFERENCE
    "See Section 6.1.1. 'IPv6 SMF DPD Option Header'
    in RFC 6621 - Simplified Multicast Forwarding
    (SMF), Macker, J., May 2012."
 ::= { smfPerfGobalGroup 11 }

smfPerfIpv6DpdHeaderInsertionsTotal OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "Packets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "A counter of the total number of IPv6 packets
        recieved which the device inserted the
        DPD header option.

        There is the potential for a counter discontinuity
        in this object if the system SMF process had been
        disabled and later enabled. In order to check for
        the occurrence of such a discontinuity when monitoring
        this counter object, it is recommended that the
        smfCfgSmfSysUpTime object also be monitored."
    REFERENCE
        "See Section 6.1.2. 'IPv6 Identification-Based
        DPD' in RFC 6621 - Simplified Multicast
        Forwarding (SMF), Macker, J., May 2012."
 ::= { smfPerfGobalGroup 12 }

--
-- Per SMF Interface Performance Table
--

smfPerfInterfaceGroup OBJECT IDENTIFIER ::= { smfPerformanceGroup 2 }

smfPerfIpv4InterfacePerfTable OBJECT-TYPE

```

```

SYNTAX          SEQUENCE OF SmfPerfIpv4InterfacePerfEntry
MAX-ACCESS      not-accessible
STATUS          current
DESCRIPTION
    "The SMF Interface Performance Table
    describes the SMF counters per
    interface."
 ::= { smfPerfInterfaceGroup 1 }

smfPerfIpv4InterfacePerfEntry OBJECT-TYPE
SYNTAX          SmfPerfIpv4InterfacePerfEntry
MAX-ACCESS      not-accessible
STATUS          current
DESCRIPTION
    "The SMF Interface Performance entry
    describes the statistics for a particular
    node interface."
INDEX { smfCfgIfIndex }
 ::= { smfPerfIpv4InterfacePerfTable 1 }

SmfPerfIpv4InterfacePerfEntry ::=
SEQUENCE {
    smfPerfIpv4MultiPktsRecvPerIf          Counter32,
    smfPerfIpv4MultiPktsForwardedPerIf     Counter32,
    smfPerfIpv4DuplMultiPktsDetectedPerIf  Counter32,
    smfPerfIpv4DroppedMultiPktsTTLExceededPerIf Counter32,
    smfPerfIpv4TTLLargerThanPreviousPerIf  Counter32
}

smfPerfIpv4MultiPktsRecvPerIf OBJECT-TYPE
SYNTAX          Counter32
UNITS           "Packets"
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "A counter of the number of multicast IP
    packets received by the SMF process on
    this device on this interface.

    There is the potential for a counter discontinuity
    in this object if the system SMF process had been
    disabled and later enabled on this interface.
    In order to check for the occurrence of such a
    discontinuity when monitoring this counter object,
    it is recommended that the smfCfgIfSmfUpTime
    object also be monitored."
 ::= { smfPerfIpv4InterfacePerfEntry 1 }

```

```
smfPerfIpv4MultiPktsForwardedPerIf  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "Packets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "A counter of the number of
        multicast IP packets forwarded by the
        SMF process on this device
        on this interface.

        There is the potential for a counter discontinuity
        in this object if the system SMF process had been
        disabled and later enabled on this interface.
        In order to check for the occurrence of such a
        discontinuity when monitoring this counter object,
        it is recommended that the smfCfgrIfSmfUpTime
        object also be monitored."
 ::= { smfPerfIpv4InterfacePerfEntry 2 }

smfPerfIpv4DuplMultiPktsDetectedPerIf  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "Packets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "A counter of the number of duplicate
        multicast IP packets detected by the
        SMF process on this device
        on this interface.

        There is the potential for a counter discontinuity
        in this object if the system SMF process had been
        disabled and later enabled on this interface.
        In order to check for the occurrence of such a
        discontinuity when monitoring this counter object,
        it is recommended that the smfCfgrIfSmfUpTime
        object also be monitored."
 ::= { smfPerfIpv4InterfacePerfEntry 3 }

smfPerfIpv4DroppedMultiPktsTTLExceededPerIf  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "Packets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "A counter of the total number of dropped
        multicast IPv4 packets by the SMF process
```

on this device on this interface  
due to TTL exceeded.

There is the potential for a counter discontinuity  
in this object if the system SMF process had been  
disabled and later enabled on this interface.

In order to check for the occurrence of such a  
discontinuity when monitoring this counter object,  
it is recommended that the smfCfgIfSmfUpTime  
object also be monitored."

::= { smfPerfIpv4InterfacePerfEntry 4 }

smfPerfIpv4TTLLargerThanPreviousPerIf OBJECT-TYPE

SYNTAX Counter32

UNITS "Packets"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A counter of the total number of IPv4 packets  
received by the SMF process on this device  
on this interface which have a TTL larger than  
that of a previously received identical packet.

There is the potential for a counter discontinuity  
in this object if the system SMF process had been  
disabled and later enabled on this interface.

In order to check for the occurrence of such a  
discontinuity when monitoring this counter object,  
it is recommended that the smfCfgIfSmfUpTime  
object also be monitored."

::= { smfPerfIpv4InterfacePerfEntry 5 }

smfPerfIpv6InterfacePerfTable OBJECT-TYPE

SYNTAX SEQUENCE OF SmfPerfIpv6InterfacePerfEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The SMF Interface Performance Table  
describes the SMF counters per  
interface."

::= { smfPerfInterfaceGroup 2 }

smfPerfIpv6InterfacePerfEntry OBJECT-TYPE

SYNTAX SmfPerfIpv6InterfacePerfEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION



"The SMF Interface Performance entry describes the counters for a particular node interface."

```

INDEX { smfCfgIfIndex }
 ::= { smfPerfIpv6InterfacePerfTable 1 }

SmfPerfIpv6InterfacePerfEntry ::=
  SEQUENCE {
    smfPerfIpv6MultiPktsRecvPerIf          Counter32,
    smfPerfIpv6MultiPktsForwardedPerIf     Counter32,
    smfPerfIpv6DuplMultiPktsDetectedPerIf  Counter32,
    smfPerfIpv6DroppedMultiPktsTTLExceededPerIf Counter32,
    smfPerfIpv6TTLLargerThanPreviousPerIf Counter32,
    smfPerfIpv6HAVAssistsReqdPerIf        Counter32,
    smfPerfIpv6DpdHeaderInsertionsPerIf    Counter32
  }

smfPerfIpv6MultiPktsRecvPerIf  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS       "Packets"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "A counter of the number of
        multicast IP packets received by the
        SMF process on this device
        on this interface.

        There is the potential for a counter discontinuity
        in this object if the system SMF process had been
        disabled and later enabled on this interface.
        In order to check for the occurrence of such a
        discontinuity when monitoring this counter object,
        it is recommended that the smfCfgIfSmfUpTime
        object also be monitored."
    ::= { smfPerfIpv6InterfacePerfEntry 1 }

smfPerfIpv6MultiPktsForwardedPerIf  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS       "Packets"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "A counter of the number of
        multicast IP packets forwarded by the
        SMF process on this device
        on this interface."

```

There is the potential for a counter discontinuity in this object if the system SMF process had been disabled and later enabled on this interface.

In order to check for the occurrence of such a discontinuity when monitoring this counter object, it is recommended that the smfCfgIfSmfUpTime object also be monitored."

```
::= { smfPerfIpv6InterfacePerfEntry 2 }
```

smfPerfIpv6DuplMultiPktsDetectedPerIf OBJECT-TYPE

SYNTAX Counter32

UNITS "Packets"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A counter of the number of duplicate multicast IP packets detected by the SMF process on this device on this interface.

There is the potential for a counter discontinuity in this object if the system SMF process had been disabled and later enabled on this interface.

In order to check for the occurrence of such a discontinuity when monitoring this counter object, it is recommended that the smfCfgIfSmfUpTime object also be monitored."

```
::= { smfPerfIpv6InterfacePerfEntry 3 }
```

smfPerfIpv6DroppedMultiPktsTTLExceededPerIf OBJECT-TYPE

SYNTAX Counter32

UNITS "Packets"

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"A counter of the number of dropped multicast IP packets by the SMF process on this device on this interface due to TTL exceeded.

There is the potential for a counter discontinuity in this object if the system SMF process had been disabled and later enabled on this interface.

In order to check for the occurrence of such a discontinuity when monitoring this counter object, it is recommended that the smfCfgIfSmfUpTime object also be monitored."

```
::= { smfPerfIpv6InterfacePerfEntry 4 }

smfPerfIpv6TTLLargerThanPreviousPerIf  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "Packets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "A counter of the total number of IPv6 packets
        received which have a TTL larger than that
        of a previously received identical packet
        by the SMF process on this device on this
        interface.

        There is the potential for a counter discontinuity
        in this object if the system SMF process had been
        disabled and later enabled on this interface.
        In order to check for the occurrence of such a
        discontinuity when monitoring this counter object,
        it is recommended that the smfCfgIfSmfUpTime
        object also be monitored."
::= { smfPerfIpv6InterfacePerfEntry 5 }

smfPerfIpv6HAVAssistsReqdPerIf  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "Packets"
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "A counter of the total number of IPv6 packets
        received by the SMF process on this device
        on this interface which required the
        HAV assist for DPD.

        There is the potential for a counter discontinuity
        in this object if the system SMF process had been
        disabled and later enabled on this interface.
        In order to check for the occurrence of such a
        discontinuity when monitoring this counter object,
        it is recommended that the smfCfgIfSmfUpTime
        object also be monitored."
::= { smfPerfIpv6InterfacePerfEntry 6 }

smfPerfIpv6DpdHeaderInsertionsPerIf  OBJECT-TYPE
    SYNTAX      Counter32
    UNITS        "Packets"
    MAX-ACCESS   read-only
    STATUS       current
```

## DESCRIPTION

"A counter of the total number of IPv6 packets received by the SMF process on this device on this interface which the device inserted the DPD header option.

There is the potential for a counter discontinuity in this object if the system SMF process had been disabled and later enabled on this interface. In order to check for the occurrence of such a discontinuity when monitoring this counter object, it is recommended that the smfCfgIfSmfUpTime object also be monitored."

```
::= { smfPerfIpv6InterfacePerfEntry 7 }
```

```
--
```

```
-- Notifications
```

```
--
```

```
smfMIBNotifObjects OBJECT IDENTIFIER ::= { smfMIBNotifications 0 }
smfMIBNotifControl OBJECT IDENTIFIER ::= { smfMIBNotifications 1 }
```

```
-- smfMIBNotifObjects
```

```
smfNotifAdminStatusChange NOTIFICATION-TYPE
```

```
  OBJECTS { smfCfgRouterIDAddrType, -- The originator of
                                     -- the notification.
             smfCfgRouterID,         -- The originator of
                                     -- the notification.
             smfCfgAdminStatus       -- The new status of the
                                     -- SMF process.
          }
```

```
  STATUS current
```

```
  DESCRIPTION
```

"smfCfgAdminStatusChange is a notification sent when a the 'smfCfgAdminStatus' object changes."

```
::= { smfMIBNotifObjects 1 }
```

```
smfNotifConfiguredOpModeChange NOTIFICATION-TYPE
```

```
  OBJECTS { smfCfgRouterIDAddrType, -- The originator of
                                     -- the notification.
             smfCfgRouterID,         -- The originator of
                                     -- the notification.
             smfCfgOperationalMode   -- The new Operations
                                     -- Mode of the SMF
```

```

-- process.
    }
    STATUS          current
    DESCRIPTION
        "smfNotifConfiguredOpModeChange is a notification
        sent when the 'smfCfgOperationalMode' object
        changes."
    ::= { smfMIBNotifObjects 2 }

smfNotifIfAdminStatusChange NOTIFICATION-TYPE
    OBJECTS { smfCfgRouterIDAddrType, -- The originator of
        -- the notification.
        smfCfgRouterID, -- The originator of
        -- the notification.
        ifName, -- The interface whose
        -- status has changed.
        smfCfgIfAdminStatus -- The new status of the
        -- SMF interface.
    }
    STATUS          current
    DESCRIPTION
        "smfCfgIfAdminStatusChange is a notification sent when a
        the 'smfCfgIfAdminStatus' object changes."
    ::= { smfMIBNotifObjects 3 }

smfNotifDpdMemoryOverflowEvent NOTIFICATION-TYPE
    OBJECTS { smfCfgRouterIDAddrType, -- The originator of
        -- the notification.
        smfCfgRouterID, -- The originator of
        -- the notification.
        smfStateDpdMemoryOverflow -- The counter of
        -- the overflows.
    }
    STATUS          current
    DESCRIPTION
        "smfNotifDpdMemoryOverflowEvents is sent when the
        number of memory overflow events exceeds the
        the 'smfNotifDpdMemoryOverflowThreshold' within the
        previous number of seconds defined by the
        'smfNotifDpdMemoryOverflowWindow'."
    ::= { smfMIBNotifObjects 4 }

-- smfMIBNotifControl
smfNotifDpdMemoryOverflowThreshold OBJECT-TYPE
    SYNTAX          Integer32 (0..255)
    UNITS            "Events"

```

```

MAX-ACCESS      read-write
STATUS          current
DESCRIPTION
    "A threshold value for the
     'smfNotifDpdmemoryOverflowEvents' object.
     If the number of occurrences exceeds
     this threshold within the previous
     number of seconds
     'smfNotifDpdMemoryOverflowWindow',
     then the 'smfNotifDpdMemoryOverflowEvent'
     notification is sent.

     The default value for this object is
     '1'."
DEFVAL { 1 }
 ::= { smfMIBNotifControl 1 }

smfNotifDpdMemoryOverflowWindow OBJECT-TYPE
    SYNTAX      TimeTicks
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "A time window value for the
         'smfNotifDpdmemoryOverflowEvents' object.
         If the number of occurrences exceeds
         the 'smfNotifDpdMemoryOverflowThreshold'
         within the previous number of seconds
         'smfNotifDpdMemoryOverflowWindow',
         then the 'smfNotifDpdMemoryOverflowEvent'
         notification is sent.

         The default value for this object is
         '1'."
    DEFVAL { 1 }
    ::= { smfMIBNotifControl 2 }

--
-- Compliance Statements
--

smfCompliances OBJECT IDENTIFIER ::= { smfMIBConformance 1 }
smfMIBGroups   OBJECT IDENTIFIER ::= { smfMIBConformance 2 }

smfBasicCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION "The basic implementation requirements for
                managed network entities that implement

```

```
        the SMF RSSA process."
MODULE -- this module
MANDATORY-GROUPS { smfCapabObjectsGroup,
                    smfConfigObjectsGroup }
 ::= { smfCompliances 1 }

smfFullCompliance MODULE-COMPLIANCE
STATUS current
DESCRIPTION "The full implementation requirements for
             managed network entities that implement
             the SMF RSSA process."
MODULE -- this module
MANDATORY-GROUPS { smfCapabObjectsGroup,
                    smfConfigObjectsGroup,
                    smfStateObjectsGroup,
                    smfPerfObjectsGroup,
                    smfNotifObjectsGroup,
                    smfNotificationsGroup
                  }
 ::= { smfCompliances 2 }

--
-- Units of Conformance
--

smfCapabObjectsGroup OBJECT-GROUP
OBJECTS {
    smfCapabilitiesOpModeID,
    smfCapabilitiesRssaID
}
STATUS current
DESCRIPTION
    "Set of SMF configuration objects implemented
    in this module."
 ::= { smfMIBGroups 1 }

smfConfigObjectsGroup OBJECT-GROUP
OBJECTS {
    smfCfgAdminStatus,
    smfCfgSmfSysUpTime,
    smfCfgRouterIDAddrType,
    smfCfgRouterID,
    smfCfgOperationalMode,
    smfCfgRssaMember,
    smfCfgIpv4Dpd,
    smfCfgIpv6Dpd,
    smfCfgMaxPktLifetime,
    smfCfgDpdEntryMaxLifetime,

```

```
        smfCfgNhdpRssaMesgTLVIncluded,
        smfCfgNhdpRssaAddrBlockTLVIncluded,

        smfCfgAddrForwardingGroupName,
        smfCfgAddrForwardingAddrType,
        smfCfgAddrForwardingAddress,
        smfCfgAddrForwardingAddrPrefixLength,
        smfCfgAddrForwardingStatus,

        smfCfgIfAdminStatus,
        smfCfgIfSmfUpTime,
        smfCfgIfRowStatus
    }
    STATUS current
    DESCRIPTION
        "Set of SMF configuration objects implemented
         in this module."
    ::= { smfMIBGroups 2 }

smfStateObjectsGroup OBJECT-GROUP
    OBJECTS {
        smfStateNodeRsStatusIncluded,
        smfStateDpdMemoryOverflow,

        smfStateNeighborRSSA,
        smfStateNeighborNextHopInterface
    }
    STATUS current
    DESCRIPTION
        "Set of SMF state objects implemented
         in this module."
    ::= { smfMIBGroups 3 }

smfPerfObjectsGroup OBJECT-GROUP
    OBJECTS {
        smfPerfIpv4MultiPktsRecvTotal,
        smfPerfIpv4MultiPktsForwardedTotal,
        smfPerfIpv4DuplMultiPktsDetectedTotal,
        smfPerfIpv4DroppedMultiPktsTTLExceededTotal,
        smfPerfIpv4TTLLargerThanPreviousTotal,

        smfPerfIpv6MultiPktsRecvTotal,
        smfPerfIpv6MultiPktsForwardedTotal,
        smfPerfIpv6DuplMultiPktsDetectedTotal,
        smfPerfIpv6DroppedMultiPktsTTLExceededTotal,
        smfPerfIpv6TTLLargerThanPreviousTotal,
        smfPerfIpv6HAVAssistsReqdTotal,
        smfPerfIpv6DpdHeaderInsertionsTotal,
```



```
        smfPerfIpv4MultiPktsRecvPerIf,
        smfPerfIpv4MultiPktsForwardedPerIf,
        smfPerfIpv4DuplMultiPktsDetectedPerIf,
        smfPerfIpv4DroppedMultiPktsTTLExceededPerIf,
        smfPerfIpv4TTLLargerThanPreviousPerIf,

        smfPerfIpv6MultiPktsRecvPerIf,
        smfPerfIpv6MultiPktsForwardedPerIf,
        smfPerfIpv6DuplMultiPktsDetectedPerIf,
        smfPerfIpv6DroppedMultiPktsTTLExceededPerIf,
        smfPerfIpv6TTLLargerThanPreviousPerIf,
        smfPerfIpv6HAVAssistsReqdPerIf,
        smfPerfIpv6DpdHeaderInsertionsPerIf
    }
    STATUS current
    DESCRIPTION
        "Set of SMF performance objects implemented
         in this module by total and per interface."
    ::= { smfMIBGroups 4 }

smfNotifObjectsGroup OBJECT-GROUP
    OBJECTS {
        smfNotifDpdMemoryOverflowThreshold,
        smfNotifDpdMemoryOverflowWindow
    }
    STATUS current
    DESCRIPTION
        "Set of SMF notification control
         objects implemented in this module."
    ::= { smfMIBGroups 5 }

smfNotificationsGroup NOTIFICATION-GROUP
    NOTIFICATIONS {
        smfNotifAdminStatusChange,
        smfNotifConfiguredOpModeChange,
        smfNotifIfAdminStatusChange,
        smfNotifDpdMemoryOverflowEvent
    }
    STATUS current
    DESCRIPTION
        "Set of SMF notifications implemented
         in this module."
    ::= { smfMIBGroups 6 }

END
```

## 8. IANA-SMF-MIB Definitions

This section contains the IANA-SMF-MIB module. This MIB module defines two textual conventions for which IANA SHOULD maintain and keep synchronized with the registry identified below within the IANAsmfOpModeIdTC and the IANAsmfRssaIdTC TEXTUAL-CONVENTIONS.

The IANAsmfOpModeIdTC defines an index that identifies through reference to a specific SMF operations mode. The index is an integer valued named-number enumeration consisting of an integer and label. IANA is to create and maintain this textual convention. Future assignments are made to anyone on a first come, first served basis. There is no substantive review of the request, other than to ensure that it is well-formed and does not duplicate an existing assignment. However, requests must include a minimal amount of clerical information, such as a point of contact (including an email address) and a brief description of the method being identified as a new SMF operations mode.

The IANAsmfRssaIdTC defines an index that identifies through reference to a specific Reduced Set Selection Algorithm (RSSA). The index is an integer valued named-number enumeration consisting of an integer and label. IANA is to create and maintain this textual convention.

Future assignments to the IANAsmfRssaIdTC for the index range 5-127 require an RFC publication (either as an IETF submission or as an RFC Editor Independent submission [RFC5742]). The type of RFC MUST be Standards Track. The specific RSSA algorithms MUST be documented in sufficient detail so that interoperability between independent implementations is possible.

Future assignments to the IANAsmfRssaIdTC for the index range 128-239 are private or local use only, with the type and purpose defined by the local site. No attempt is made to prevent multiple sites from using the same value in different (and incompatible) ways. There is no need for IANA to review such assignments (since IANA will not record these) and assignments are not generally useful for broad interoperability. It is the responsibility of the sites making use of the Private Use range to ensure that no conflicts occur (within the intended scope of use).

Future assignments to the IANAsmfRssaIdTC for the index range 240-255 are to facilitate experimentation. These require an RFC publication (either as an IETF submission or as an RFC Editor Independent submission [RFC5742]). The type of RFC MUST be Experimental. The RSSA algorithms MUST be documented in sufficient detail so that interoperability between independent implementations is possible.

```
IANA-SMF-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```
    MODULE-IDENTITY, mib-2
        FROM SNMPv2-SMI
    TEXTUAL-CONVENTION
        FROM SNMPv2-TC;
```

```
ianaSmfMIB MODULE-IDENTITY
```

```
    LAST-UPDATED "201408121300Z" -- August 12, 2014
    ORGANIZATION "IANA"
    CONTACT-INFO "Internet Assigned Numbers Authority
```

```
        Postal: ICANN
                4676 Admiralty Way, Suite 330
                Marina del Rey, CA 90292
```

```
        Tel:    +1 310 823 9358
        E-Mail: iana@iana.org"
```

```
DESCRIPTION "This MIB module defines the
              IANAsmfOpModeIdTC and IANAsmfRssaIdTC
              Textual Conventions, and thus the
              enumerated values of the
              smfCapabilitiesOpModeID and
              smfCapabilitiesRssaID objects defined
              in the SMF-MIB."
REVISION    "201408121300Z" -- August 12, 2014
DESCRIPTION "Initial version of this MIB as published in
              RFC KKKK."
 ::= { mib-2 kkkk }
```

```
IANAsmfOpModeIdTC ::= TEXTUAL-CONVENTION
```

```
    STATUS      current
```

```
DESCRIPTION
```

```
    "An index that identifies through reference to a specific
     SMF operations mode. There are basically three styles
     of SMF operation with reduced relay sets currently
     identified:
```

```
        Independent operation 'independent(1)' -
            SMF performs its own relay
            set selection using information from an associated
            MANET NHDP process.
```

```
        CDS-aware unicast routing operation 'routing(2)' -
            a coexistent unicast routing
            protocol provides dynamic relay
```

set state based upon its own control plane  
CDS or neighborhood discovery information.

Cross-layer operation 'crossLayer(3)' -  
SMF operates using neighborhood  
status and triggers from a  
cross-layer information base for dynamic relay  
set selection and maintenance.

IANA MUST update this textual convention accordingly.

The definition of this textual convention with the  
addition of newly assigned values is published  
periodically by the IANA, in either the Assigned  
Numbers RFC, or some derivative of it specific to  
Internet Network Management number assignments. (The  
latest arrangements can be obtained by contacting the  
IANA.)

Requests for new values SHOULD be made to IANA via  
email (iana@iana.org)."

#### REFERENCE

"See Section 7.2. 'Reduced Relay Set Forwarding',  
and the Appendices A, B and C in  
RFC 6621 - Simplified Multicast Forwarding  
(SMF), Macker, J., May 2012."

SYNTAX INTEGER {  
    independent (1),  
    routing (2),  
    crossLayer (3)  
    -- future (4-255)  
}

IANAsmfRssaIdTC ::= TEXTUAL-CONVENTION

STATUS current

#### DESCRIPTION

"An index that identifies through reference to a specific  
RSSA algorithms. Several are currently defined  
in the Appendix A, B and C of RFC 6621.

Examples of RSSA algorithms already identified within  
this TC are:

Classical Flooding (cF(1)) - is the standard  
flooding algorithm where each node in the next  
retransmits the information on each of its interfaces.

Source-Based Multipoint Relay (sMPR(2)) -  
this algorithm is used by Optimized Link State Routing (OLSR) and OLSR version 2 (OLSRv2) protocols for the relay of link state updates and other control information [RFC3626]. Since each router picks its neighboring relays independently, sMPR forwarders depend upon previous hop information (e.g., source MAC address) to operate correctly.

Extended Connected Dominating Set (eCDS(3)) -  
defined in [RFC5614] this algorithm forms a single CDS mesh for the SMF operating region. Its packet-forwarding rules are not dependent upon previous hop knowledge in contrast to sMPR.

Multipoint Relay Connected Dominating Set (mprCDS(4)) -  
This algorithm is an extension to the basic sMPR election algorithm that results in a shared (non-source-specific) SMF CDS. Thus, its forwarding rules are not dependent upon previous hop information, similar to eCDS.

IANA MUST update this textual convention accordingly.

The definition of this textual convention with the addition of newly assigned values is published periodically by the IANA, in either the Assigned Numbers RFC, or some derivative of it specific to Internet Network Management number assignments. (The latest arrangements can be obtained by contacting the IANA.)

Requests for new values SHOULD be made to IANA via email (iana@iana.org)."

#### REFERENCE

"See, e.g.,

Section 8.1.1. 'SMF Message TLV Type',  
Appendix A. 'Essential Connecting Dominating Set (E-CDS) Algorithm',  
Appendix B. 'Source-Based Multipoint Relay (S-MPR) Algorithm', and  
Appendix C. 'Multipoint Relay Connected Dominating Set (MPR-CDS) Algorithm'  
in RFC 6621 - Macker, J., 'Simplified Multicast Forwarding (SMF)', May 2012.

RFC 3626 - Clausen, T., and P. Jacquet, 'Optimized Link

State Routing Protocol (OLSR)', October 2003.

RFC 5614 - Ogier, R. and P. Spagnolo, 'Mobile Ad Hoc Network (MANET) Extension of OSPF Using Connected Dominating Set (CDS) Flooding', August 2009.

```

"
SYNTAX      INTEGER {
                cF(1),
                sMPR(2),
                eCDS(3),
                mprCDS(4)
                -- future(5-127)
                -- noStdAction(128-239)
                -- experimental(240-255)
            }

```

END

## 9. Security Considerations

This section discusses security implications of the choices made in this SMF-MIB module.

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations. These are the tables and objects and their sensitivity/vulnerability:

- o 'smfCfgAdminStatus' - this writable configuration object controls the operational status of the SMF process. If this setting is configured inconsistently across the MANET multicast domain, then delivery of multicast data may be inconsistent across the domain; some nodes may not receive multicast data intended for them.
- o 'smfCfgRouterIDAddrType' and 'smfCfgRouterID' - these writable configuration objects define the ID of the SMF process. These objects should be configured with a routable address defined on the local SMF device. The smfCfgRouterID is a logical identification that MUST be configured as unique across inter-operating SMF neighborhoods and it is RECOMMENDED to be chosen as the numerically largest address contained in a node's 'Neighbor Address List' as defined in NHDP. A smfCfgRouterID MUST be unique within the scope of the operating MANET network regardless of the method used for selecting it. If these objects are mis-configured or configured in-consistently across the MANET, then the ability

of various RSSA algorithms, e.g., ECDS, may be compromised. This would potentially result in some routers within the MANET not receiving multicast packets destined to them. Hence, intentionally mis-configuring these objects could pose a form of Denial-of-Service (DOS) attack against the MANET.

- o 'smfCfgOpMode' - this writable configuration object defines the operational mode of the SMF process. The operational mode defines how the SMF process receives its data to form its local estimate of the CDS. It is recommended that the value for this object be set consistently across the MANET to ensure proper operation of the multicast packet forwarding. If the value for this object is set inconsistently across the MANET, the result may be that multicast packet delivery will be compromised within the MANET. Hence, intentionally mis-configuring this object could pose a form DOS attack against the MANET.
- o 'smfCfgRssa' - this writable configuration object sets the specific Reduced Set Selection Algorithm (RSSA) for the SMF process. If this object is set inconsistently across the MANET domain, multicast delivery of data will likely fail. Hence, intentionally mis-configuring this object could pose a form DOS attack against the MANET.
- o 'smfCfgRssaMember' - this writable configuration object sets the 'interest' of the local SMF node in participating in the CDS. Setting this object to 'never(3)' on a highly highly connected device could lead to frequent island formation. Setting this object to 'always(2)' could support data ex-filtration from the MANET domain.
- o 'smfCfgIpv4Dpd' - this writable configuration object sets the duplicate packet detection method, i.e., H-DPD or I-DPD, for forwarding of IPv4 multicast packets. Forwarders may operate with mixed H-DPD and I-DPD modes as long as they consistently perform the appropriate DPD routines outlined [RFC6621]. However, it is RECOMMENDED that a deployment be configured with a common mode for operational consistency.
- o 'smfCfgIpv6Dpd' - this writable configuration object sets the duplicate packet detection method for forwarding of IPv6 multicast packets. Since IPv6 SMF does specify an option header, the interoperability constraints are not as loose as in the IPv4 version, and forwarders SHOULD NOT operate with mixed H-DPD and I-DPD modes. Hence the value for this object SHOULD be consistently set within the forwarders comprising the MANET, else inconsistent forwarding may result unnecessary multicast packet dropping.

- o 'smfCfgMaxPktLifetime' - this writable configuration object sets the estimate of the network packet traversal time. If set too small, this could lead to poor multicast data delivery ratios throughout the MANET domain. This could serve as a form of DOS attack if this object value is set too small.
- o 'smfCfgDpdEntryMaxLifetime' - this writable configuration object sets the maximum lifetime (in seconds) for the cached DPD records for the combined IPv4 and IPv6 methods. If the memory is running low prior to the MaxLifetime being exceeded, the local SMF devices should purge the oldest records first. If this object value is set too small, then the effectiveness of the SMF DPD algorithms may become greatly diminished causing a higher than necessary packet load on the MANET.
- o 'smfCfgNhdpRssaMesgTLVIncluded' - this writable configuration object indicates whether the associated NHDP messages include the RSSA Message TLV, or not. It is highly RECOMMENDED that this object be set to 'true(1)' when the SMF operation mode is set to independent as this information will inform the local forwarder of the RSSA algorithm implemented in neighboring forwarders and is used to ensure consistent forwarding across the MANET. While it is possible that SMF neighbors MAY be configured differently with respect to the RSSA algorithm and still operate cooperatively, but these cases will vary dependent upon the algorithm types designated and this situation SHOULD be avoided.
- o 'smfCfgNhdpRssaAddrBlockTLVIncluded' - this writable configuration object indicates whether the associated NHDP messages include the the RSSA Address Block TLV, or not. The smfNhdpRssaAddrBlockTLVIncluded is optional in all cases as it depends on the existence of an address block which may not be present. If this SMF device is configured with NHDP, then this object should be set to 'true(1)' as this TLV enables CDS relay algorithm operation and configuration to be shared among 2-hop neighborhoods. Some relay algorithms require 2-hop neighbor configuration in order to correctly select relay sets.
- o 'smfCfgAddrForwardingTable' - the writable configuration objects in this table indicate which multicast IP addresses are to be forwarded by this SMF node. Misconfiguration of rows within this table can limit the ability of this SMF device to properly forward multicast data.
- o 'smfCfgInterfaceTable' - the writable configuration objects in this table indicate which SMF node interfaces are participating in the SMF packet forwarding process. Misconfiguration of rows within this table can limit the ability of this SMF device to



properly forward multicast data.

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to even encrypt the values of these objects when sending them over the network via SNMP. These are the tables and objects and their sensitivity/vulnerability:

- o 'smfNodeRsStatusIncluded' - this readable state object indicates that this SMF node is part of the CDS, or not. Being part of the CDS makes this node a distinguished device. It could be exploited for data ex-filtration, or denial of service attacks.
- o 'smfStateNeighborTable' - the readable state objects in this table indicate current neighbor nodes to this SMF node. Exposing this information to an attacker could allow the attacker easier access to the larger MANET domain.

The remainder of the objects in the SMF-MIB module are performance counter objects. While these give an indication of the activity of the SMF process on this node, it is not expected that exposing these values pose a security risk to the MANET network.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPSec), even then, there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.

Implementations MUST provide the security features described by the SNMPv3 framework (see [RFC3410] ), including full support for authentication and privacy via the User-based Security Model (USM) [RFC3414] with the AES cipher algorithm [RFC3826]. Implementations MAY also provide support for the Transport Security Model (TSM) [RFC5591] in combination with a secure transport such as SSH [RFC5592] or TLS/DTLS [RFC6353].

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

## 10. Applicability Statement

This document describes objects for configuring parameters of the Simplified Multicast Forwarding [RFC6621] process on a Mobile Ad-Hoc Network (MANET) router. This MIB module, denoted SMF-MIB, also reports state and performance information and notifications. This section provides some examples of how this MIB module can be used in MANET network deployments. A fuller discussion of MANET network management use cases and challenges will be provided elsewhere.

SMF is designed to allow MANET routers to forward IPv4 and IPv6 packets over the MANET and cover the MANET nodes through the automatic discovery of efficient estimates of the Minimum Connected Dominating Set (MCDS) of nodes within the MANET. The MCDS are estimated using the Relay Set Selection Algorithms (RSSAs) discussed within this document. In the following, three scenarios are listed where this MIB module is useful, i.e.,

- o For a Parking Lot Initial Configuration Situation - it is common for the vehicles comprising the MANET being forward deployed at a remote location, e.g., the site of a natural disaster, to be off-loaded in a parking lot where an initial configuration of the networking devices is performed. The configuration is loaded into the devices from a fixed location Network Operation Center (NOC) at the parking lot and the vehicles are stationary at the parking lot while the configuration changes are made. Standards-based methods for configuration management from the co-located NOC are necessary for this deployment option. The set of interesting configuration objects for the SMF process are listed within this MIB module.
- o For Mobile vehicles with Low Bandwidth Satellite Link to a Fixed NOC - Here the vehicles carrying the MANET routers carry multiple wireless interfaces, one of which is a relatively low-bandwidth on-the-move satellite connection which interconnects a fix NOC to the nodes of the MANET. Standards-based methods for monitoring and fault management from the fixed NOC are necessary for this deployment option.
- o For Fixed NOC and Mobile Local Manager in Larger Vehicles - for larger vehicles, a hierarchical network management arrangement is useful. Centralized network management is performed from a fixed NOC while local management is performed locally from within the vehicles. Standards-based methods for configuration, monitoring and fault management are necessary for this deployment option.

Here we provide an example of the simplest of configurations to establish an operational multicast forwarding capability in a MANET.

This discussion only identifies the configuration necessary through the SMF-MIB module and assumes that other configuration has occurred. Assume that the MANET is to support only IPv4 addressing and that the MANET nodes are to be configured in the context of the Parking Lot Initialization case above. Then the SMF-MIB module defines ten configuration OIDs and two configuration tables, i.e., the smfCfgAddrForwardingTable and the smfCfgInterfaceTable. Of the ten OIDs defined, all but one, i.e., the smfCfgRouterID, have DEFVAL clauses which allow for a functional configuration of the SMF process within the MANET. The smfCfgRouterIDType defaults to 'ipv4' so the smfCfgRouterID can be set as, e.g. (assuming the use of the Net-SNMP toolkit),:

```
snmpset [options] <smfCfgRouterID_OID>.0 a 192.0.2.100
```

If the smfCfgAddrForwardingTable is left empty, then the SMF local forwarder will forward all multicast addresses. So this table does not require configuration if you want to have the MANET forward all multicast addresses.

All that remains is to configure at least one row in the smfCfgInterfaceTable. Assume that the node has a wireless interface with an <ifName>='wlan0' and an <ifIndex>='1'. All of the objects in the rows of the smfCfgInterfaceTable have a DEFVAL clause, hence only the RowStatus object needs to be set. So the SMF process will be activated on the 'wlan0' interface by the following network manager snmpset command:

```
snmpset [options] <smfCfgIfRowStatus>.1 i active(1)
```

At this point, the configured forwarder will begin a Classical Flooding algorithm to forward all multicast addresses IPv4 packets it receives.

To provide a more efficient multicast forwarding within the MANET, the network manager could walk the smfCapabilitiesTable to identify other SMF operational modes, e.g.,:

```
snmpwalk [options] <smfCapabilitiesTable>
```

```
SMF-MIB::smfCapabilitiesIndex.1 = INTEGER: 1
```

```
SMF-MIB::smfCapabilitiesIndex.2 = INTEGER: 2
```

```
SMF-MIB::smfCapabilitiesOpModeID.1 = INTEGER: cfOnly(1)
```

```
SMF-MIB::smfCapabilitiesOpModeID.2 = INTEGER: independent(2)
```

SMF-MIB::smfCapabilitiesRssaID.1 = INTEGER: cF(1)

SMF-MIB::smfCapabilitiesRssaID.2 = INTEGER: eCDS(3)

In this example, the forwarding device also supports the Extended Connected Dominating Set (eCDS) RSSA with the forwarder in the 'independent(2)' operational mode. If the network manager were to then issue an snmpset, e.g.,:

```
snmpset [options] <smfCfgOperationalMode>.0 i 2
```

then the local forwarder would switch its forwarding behavior from Classical Flooding to the more efficient eCDS flooding.

## 11. IANA Considerations

This document defines two MIB modules:

- o SMF-MIB is defined in Section 7 and is an experimental MIB module.
- o IANA-SMF-MIB is defined in Section 8 and is an IANA MIB module that IANA is requested to maintain.

Thus, there are three actions requested of IANA:

1. IANA is requested to allocate an OBJECT IDENTIFIER value and record it in the SMI Numbers registry in the sub-registry called "SMI Experimental Codes" under the experimental branch (1.3.6.1.3).

Decimal	Name	Description	Reference
xxxx	smfMib	SMF-MIB	[This.I-D]

[RFC Editor Note: Please replace the tag "xxxx" in this document with the value assigned by IANA and remove this note.]

2. IANA is requested to allocate an OBJECT IDENTIFIER value and record it in the SMI Numbers registry in the sub-registry called "SMI Network Management MGMT Codes Internet-standard MIB" under the mib-2 branch (1.3.6.1.2.1).

Decimal	Name	Description	Reference
kkkk	ianaSmfMIB	IANA-SMF-MIB	[This.I-D]

[RFC Editor Note: Please replace the tag "kkkk" in this document with the value assigned by IANA and

remove this note.]

3. IANA is requested to maintain a MIB module called ianaSmfMIB and populate it with the initial MIB module defined in Section 8 of this document by creating a new entry in the registry "IANA Maintained MIBs" called "IANA-SMF-MIB".

## 12. Contributors

This MIB document uses the template authored by D. Harrington which is based on contributions from the MIB Doctors, especially Juergen Schoenwaelder, Dave Perkins, C.M.Heard and Randy Presuhn.

## 13. Acknowledgements

The authors would like to acknowledge the valuable comments from Sean Harnedy in the early phases of the development of this MIB module. The authors would like to thank Adrian Farrel, Dan Romascanu, Juergen Shoenwaelder, Stephen Hanna, and Brian Haberman for their careful review of this document and their insightful comments. We also wish to thank the entire MANET WG for many reviews of this document. Further the authors would like to thank James Nguyen for his careful review and comments on this MIB module and his work on the definitions of the follow-on MIB modules to configure specific RSSA algorithms related to SMF. Further, the authors would like to acknowledge to work of James Nguyen, Brian Little, Ryan Morgan and Justin Dean on their software development of the SMF-MIB.

## 14. References

### 14.1. Normative References

- [RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.
- [RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.
- [RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.
- [RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.

- [RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.
- [RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.
- [RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.
- [RFC3626] Clausen, T. and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)", RFC 3626, October 2003.
- [RFC5742] Alvestrand, H. and R. Housley, "IESG Procedures for Handling of Independent and IRTF Stream Submissions", BCP 92, RFC 5742, December 2009.
- [RFC5614] Ogier, R. and P. Spagnolo, "Mobile Ad Hoc Network (MANET) Extension of OSPF Using Connected Dominating Set (CDS) Flooding", RFC 5614, August 2009.
- [RFC6621] Macker, J., "Simplified Multicast Forwarding", RFC 6621, May 2012.

#### 14.2. Informative References

- [RFC3414] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", STD 62, RFC 3414, December 2002.
- [RFC3826] Blumenthal, U., Maino, F., and K. McCloghrie, "The Advanced Encryption Standard (AES) Cipher Algorithm in the SNMP User-based Security Model", RFC 3826, June 2004.
- [RFC5591] Harrington, D. and W. Hardaker, "Transport Security Model for the Simple Network Management Protocol (SNMP)", STD 78, RFC 5591, June 2009.
- [RFC5592] Harrington, D., Salowey, J., and W. Hardaker, "Secure

Shell Transport Model for the Simple Network Management Protocol (SNMP)", RFC 5592, June 2009.

- [RFC6353] Hardaker, W., "Transport Layer Security (TLS) Transport Model for the Simple Network Management Protocol (SNMP)", STD 78, RFC 6353, July 2011.
- [RFC4293] Routhier, S., "Management Information Base for the Internet Protocol (IP)", RFC 4293, April 2006.
- [RFC5132] McWalter, D., Thaler, D., and A. Kessler, "IP Multicast MIB", RFC 5132, December 2007.

#### Appendix A.

```
*****
* Note to the RFC Editor (to be removed prior to publication) *
*
* 1) The reference to RFCXXXX throughout this document point *
* to the current draft-ietf-manet-smf-xx.txt. This needs *
* to be replaced with the XXXX RFC number for the SMF *
* publication. *
*
* 2) This document also contains the IANA-SMF-MIB module *
* which is defined by this specification above. IANA should *
* take over the IANA-SMF-MIB and keep it synchronized with *
* the registries identified within the contained *
* IANAsmfOpModeIdTC and IANAsmfRssaIdTC TEXTUAL-CONVENTIONS. *
*
*****
```

#### Authors' Addresses

Robert G. Cole  
US Army CERDEC  
6010 Frankford Road  
Aberdeen Proving Ground, Maryland 21005  
USA

Phone: +1 443 395 8744  
EMail: robert.g.cole@us.army.mil

Joseph Macker  
Naval Research Laboratory  
Washington, D.C. 20375  
USA

EMail: macker@itd.nrl.navy.mil

Brian Adamson  
Naval Research Laboratory  
Washington, D.C. 20375  
USA

EMail: adamson@itd.nrl.navy.mil



