

Network Working Group
Internet-Draft
Intended status: Informational
Expires: August 15, 2014

T. Clausen
LIX, Ecole Polytechnique
U. Herberg
Fujitsu Laboratories of America
February 11, 2014

Snapshot of OLSRv2-Routed MANET Management
draft-clausen-manet-olsrv2-management-snapshot-01

Abstract

This document describes how Mobile Ad Hoc Networks (MANETs) are typically managed, in terms of pre-deployment management, as well as rationale and means of monitoring and management of MANET routers running the routing protocol OLSRv2 and its constituent protocol NHDP. Apart from pre-deployment management for setting up IP addresses and security related credentials, OLSRv2 only needs routers to agree one single parameter (called "C"). Other parameters for tweaking network performance may be determined during operation of the network, and need not be the same in all routers. This, using MIB modules and related management protocols such as SNMP (or possibly other, less "chatty", protocols). In addition, for debugging purposes, monitoring data and performance related counters can be sent to the Network Management Station (NMS) via standardized management protocols.

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 15, 2014.

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
1.1. Statement of Purpose	3
2. Terminology	3
3. Pre-Deployment Management	3
3.1. Lower Layer Alignment	4
3.2. Interface Addresses	4
3.3. Security Material	5
3.4. The Value of C	5
4. How do we Manage MANETs?	5
4.1. Internal Management	6
4.2. External Management	6
5. What and Why do we Manage and Monitor?	7
6. This Document does not Constrain how to Manage and Monitor MANETs	8
7. Acknowledgments	9
8. Informative References	9
Authors' Addresses	10

1. Introduction

The MANET routing protocol OLSRv2 [OLSRv2], as well as its constituent parts NHDP [RFC6130], [RFC5497], [RFC5148], [RFC5444], [RFC6622bis], [OLSRv2-integrity], is designed to autonomously maintain routes across a dynamic network topology. OLSRv2 is designed so as to minimize operator intervention throughout the duration of a network deployment, and to allow for heterogeneous configuration of routers within the same network deployment: most configuration values are either of local significance only (e.g., message jitter parameters) or, when they are not, are carried in control signals exchanged between routers (e.g., information validity time).

All the same, a small set of configuration options must be established in each router prior to deployment, with some requiring agreement among all the routers within the same deployment. Furthermore, throughout the duration of a network deployment, external management and monitoring of a network may be desirable, e.g., for performance optimization or debugging purposes.

1.1. Statement of Purpose

Deployments of OLSRv2 are diverse, and may include community networks, constrained environments, tactical networks, etc. Each such environment may present distinctly different requirements as to management and monitoring.

This document does therefore explicitly not pretend to provide an exhaustive description of how all OLSRv2 network deployments should be managed and monitored - and does, specifically, not prescribe any management model.

What this document does, however, is to present how some OLSRv2 network deployments are managed and monitored, using well-established management patterns and well-known protocols.

2. Terminology

This document uses terminology from [OLSRv2], [RFC6130], and [RFC5497].

3. Pre-Deployment Management

Prior to operation of an OLSRv2 network, or more precisely, prior to proper operation of OLSRv2 and its constituent parts, certain

parameters need to be configured on each router. The following sections describe the required pre-deployment management.

3.1. Lower Layer Alignment

Interoperability between routers requires alignment of lower protocol layers below OLSRv2. In particular, all routers in the same MANET topology must be pre-configured to use the same IP address family (IPv4 or IPv6). In a single OLSRv2 topology, it is not possible to mix IPv4 and IPv6 addresses, notably because [RFC5444] messages can contain either IPv4 *or* IPv6 addresses, but not both at the same time. It is, however, possible to run two instances of OLSRv2, one instance for IPv4 and another one for IPv6, within the same network.

In addition to the IP address family, other lower layer parameters may also need to be aligned, e.g., radio channel selections. A single OLSRv2 topology may, of course, span different link layers (or the same link layer with different configuration settings such as cryptographic keys) when routers in the topology have OLSRv2 interfaces towards these different link layers.

3.2. Interface Addresses

According to [RFC6130], and as used by [OLSRv2], each interface of a router must be configured with at least one IP address. [RFC6130] provides guidance as to the characteristics of such IP addresses, including the (limited) conditions under which an IP address may be configured on multiple interfaces.

While automatic configuration of IP addresses on router interfaces is not excluded, currently no address autoconfiguration protocols have been standardized (in the IETF) to accomplish this. As a consequence, static configuration, or proprietary (as in: non-standardized) protocols ensure this.

Note that this required pre-deployment of interface addresses does not include "external" IP addresses, i.e., IP addresses that are configured on local non-MANET interfaces or IP addresses from remote destinations reachable through this router (i.e., addresses for which this router serves as gateway). These can be added or removed dynamically during runtime of OLSRv2. Such local non-MANET interface addresses are managed by way of the Local Interface Set (as defined in [RFC6130]) and remote addresses by way of the Attached Network Set (as defined in [OLSRv2]).

3.3. Security Material

Security material (keys, algorithms, etc.) must be available for generating Integrity Check Values (ICVs) for outgoing control messages, and to allow validating ICVs in incoming control messages [RFC6622bis] [OLSRv2-integrity].

The appropriate way of making such security material available is dependent on the deployment type. For example, community networks (such as "Funkfeuer", <http://funkfeuer.at>), do currently not use any security at all. Other deployment types may use a simple manual shared key distribution mechanism, or may use a proprietary key distribution protocol. Tactical networks have much more stringent requirements for distributing key material, e.g., using manual distribution of the keys on encrypted USB keys, and with defensive mechanisms (up to and including mechanisms involving depleted uranium) if the keys are compromised.

In general, Automatic Key Management (AKM) as well as static/manual or other out-of-band mechanisms, can be viable options for distributing keys. Currently, no standardized AKM mechanism for MANETs exist. If the IETF standardizes such mechanisms in the future, for deployment types where such is appropriate, these can be used for distributing keys. Until such time, manual key distribution as well as proprietary mechanisms, prevail.

The important point to make here, however, is that by whichever method (automatic/manual, dynamic/static, ...) a key and other security material is made available, the security mechanisms of OLSRv2, as defined by [OLSRv2-integrity], will be able to properly use it for generating and validating ICVs.

3.4. The Value of C

The only pre-deployment configuration parameter that directly impacts protocol operation is the value of C. This value is used by each router for calculating the representation of interval and validity time, as included in control messages. All routers in a deployment must agree on the value of C, as described in [RFC5497].

4. How do we Manage MANETs?

A deployed OLSRv2 network is, as previously discussed, operating autonomously, but occasionally with internal or external management operations being desirable, described in the following two sections.

4.1. Internal Management

Internal management describes a local process running on an router that automatically (i.e., without external messaging or human interaction) modifies the configuration of OLSRv2 based on different environmental factors. For example, the HELLO interval may be updated according to the rate of topology changes measured (or, inferred: after all, the 'M' in MANET is for "Mobility") locally: if the rate is high, then a more frequent HELLO update assures that routes are more accurate. At a lower rate of topology changes, network capacity and energy capacity of the router may be conserved by increasing the HELLO interval.

Depending on the use case, many different automatic configuration agents can be envisioned. As parameters in NHDP and OLSRv2 are either only used locally or, in the case of HELLO_INTERVAL and REFRESH_INTERVAL, are selected locally and then included in the messages exchanged between adjacent routers in their HELLO messages, none of these automatic local configuration methods needs necessarily to be standardized: different routers doing different things will interoperate.

4.2. External Management

For the deployments described by this document (but see Section 6), external management operations are undertaken by a central Network Management Station (NMS).

The MIB modules developed for OLSRv2 [OLSRv2-MIB] and for its constituent protocol NHDP [RFC6779] are verbose, in as much as that they expose for interrogation the complete protocol and router state, as well as enable setting all parameters (timer intervals, time-outs, jitter values etc.). They do explicitly not enable setting the value of C (as that is required to be constant and uniform across the network, see Section 3.4), nor distributing security material (see Section 3.3).

In some deployments, the NMS communicates with individual routers by way of SNMP - and, more commonly, by way of "proprietary" simpler, less verbose and (often) less secure protocols, and over UDP. Note that this does not constitute a recommendation, but rather an observation that (apparently) SNMP has found less application in MANETs.

The predecessor of OLSRv2, OLSR [RFC3626] did not have an associated MIB module. Many deployments of OLSR did not support network management operations per se (i.e., configuration-on-launch was the way in which routers in these deployments were managed). Those

implementations and deployments of OLSR that did support network management operations used a similar architecture to the one described in this document, but with "proprietary" protocols and APIs for parameters and router states, "proprietary" data-models, etc. It can be speculated that the "proprietary" protocols used for communication between the NMS and the MIB modules on each router also for OLSRv2, in part, exist as inherited from the protocols used for OLSR.

Finally, it is uncommon to see an NMS permanently active in a deployed OLSRv2 network; rather, on an "ad hoc" basis an NMS is introduced into the network, parameters configured or state interrogated, following which the NMS disappears.

5. What and Why do we Manage and Monitor?

As indicated earlier, OLSRv2 and its constituent protocol NHDP, are reasonably robust with respect to parameter values: a deployment can operate with different parameters used in different routers in the same network. That being said, adapting these parameters according to circumstances is (often) desired. For example, in a stable network (such as a wired network), TC messages may be sent infrequently and with long validity times, whereas in a highly dynamic network (such as in a vehicular network) TC messages may need to be sent more frequently and HELLO messages for discovering the local topology (almost) continuously. In a similar vein, the message emission intervals and the information validity times should also be commensurate with the available network capacity: millisecond intervals between TC messages, for example, will consume all available network capacity whereas hourly intervals will be inappropriate even for a static and stable, wired, network (by way of simply new routers arriving in the network, which will not "learn" the network topology before undue long delays).

This adaptation may happen autonomously by a central NMS monitoring and adopting the parameters globally, autonomously by an NMS in each router, monitoring its local topology (and its stability) and adapting parameters locally, or by manual operator intervention.

Given the dynamic and evolutive topology of OLSRv2 networks, a highly desirable property of an NMS is the ability to display and offer visibility of the current network status - for example, to display a graphical map of which routers are currently part of the network. As a proactive protocol, OLSRv2 maintains, in each router, a topology map including all destinations and a subset of the links present in the network (particularly true in a very dense network). A typical feature of an NMS is to inquire as to the topology map of a single

router. A slightly less typical feature is to inquire all (or, at least, many) routers in a network, with the purpose of presenting a complete topology map.

In addition to actively monitoring an OLSRv2 network, erroneous or unusual conditions on an router can be flagged to management, e.g., detection of an unusually high number of 1-hop or 2-hop neighborhood changes in a short amount of time, indicating potential problems in that area of the network. [RFC6779] and [OLSRv2-MIB] facilitate proactively sending "notifications" (also known as traps) from the router towards an NMS. The MIB modules defined in [RFC6779] and [OLSRv2-MIB] allow for defining both the threshold and the time window of how many times this erroneous condition may occur in the time window before the notification is sent to the NMS. Once the NMS receives a notification, a network operator may investigate if there is a problem that needs to be resolved, e.g., by changing parameters via the above-described external management.

6. This Document does not Constrain how to Manage and Monitor MANETs

As explained in Section 1, this document describes how, what and why some (typical) OLSRv2 networks are managed and monitored as of early 2014. As such, the document is reflexive, not prescriptive: it does not stipulate requirements for how to manage OLSRv2 networks, nor does it claim to be a complete list of all management patterns or protocols. Other ways of managing an OLSRv2 network are very well imaginable - now, or in future deployments of OLSRv2.

As an example of such a "future management scenario", rather than managing and monitoring routers from a central NMS, a distributed, autonomous management system between multiple routers can be envisioned. In particular, monitoring data that is used to debug network problems and to tweak inefficiencies could be distributed amongst a group of routers in the same network. This would both address problems of single point of failure when using only a single NMS, as well provide additional information about groups of multiple routers, rather than a single router. An example use for such a distributed information flow would be to identify areas of a network wherein, e.g., due to different router densities, message sending interval parameters could be exchanged and optimal values negotiated between routers, so as to obtain locally optimized performance.

While such a management model is highly interesting, it is also at present entirely fictional - at least outside the realm of research. It is included to, both, indicate directions being explored (but not exploited), and to insist that the intent of this document is not to prescribe how MANETs are to be managed, in the presence or in the

future, but to describe the (known) state of how MANETs are managed, presently.

7. Acknowledgments

The authors would like to gratefully acknowledge the following people for intense technical discussions, early reviews, and comments on the documents: Alan Cullen (BAE Systems), Christopher Dearlove (BAE Systems), and Adrian Farrel (Juniper).

8. Informative 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.
- [OLSRv2-MIB] Herberg, U., Cole, R., and T. Clausen, "Definition of Managed Objects for the Optimized Link State Routing Protocol version 2", work in progress draft-ietf-manet-olsrv2-mib-12, June 2013.
- [OLSRv2-integrity] Herberg, U., Dearlove, C., and T. Clausen, "Integrity Protection for Control Messages in NHDP and OLSRV2", work in progress draft-ietf-manet-nhdp-olsrv2-sec-05, September 2013.
- [RFC3626] Clausen, T. and P. Jacquet, "The Optimized Link State Routing Protocol", RFC 3626, October 2003.
- [RFC5148] Clausen, T., Dearlove, C., and B. Adamson, "Jitter Considerations in Mobile Ad Hoc Networks (MANETs)", RFC 5148, February 2008.
- [RFC5444] Clausen, T., Dearlove, C., Dean, J., and C. Adjih, "Generalized 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.

[RFC6622bis]

Herberg, U., Clausen, T., and C. Dearlove, "Integrity Check Value and Timestamp TLV Definitions for Mobile Ad Hoc Networks (MANETs)", work in progress draft-ietf-manet-rfc6622-bis-06, November 2013.

[RFC6779] Herberg, U., Cole, R., and I. Chakeres, "Definition of Managed Objects for the Neighborhood Discovery Protocol", RFC 6779, May 2012.

Authors' Addresses

Thomas Clausen
LIX, Ecole Polytechnique
91128 Palaiseau Cedex,
France

Phone: +33-6-6058-9349
Email: T.Clausen@computer.org
URI: <http://www.thomasclausen.org>

Ulrich Herberg
Fujitsu Laboratories of America
1240 E Arques Ave
Sunnyvale CA 94086,
US

Phone:
Email: ulrich@herberg.name
URI: <http://www.herberg.name>

Mobile Ad hoc Networking (MANET)
Internet-Draft
Updates: RFC 6130, RFC 7181
(if approved)
Intended status: Standards Track
Expires: December 22, 2014

C. Dearlove
BAE Systems ATC
T. Clausen
LIX, Ecole Polytechnique
June 20, 2014

An Optimization for the MANET Neighborhood Discovery Protocol (NHDP)
draft-dearlove-manet-nhdp-optimization-01

Abstract

The link quality mechanism of the MANET Neighborhood Discovery Protocol (NHDP) enables "ignoring" some 1-hop neighbors if the measured link quality from that 1-hop neighbor is below an acceptable threshold, while still retaining the corresponding link information as acquired from HELLO message exchange. This allows immediate reinstatement of the 1-hop neighbor if the link quality later improves sufficiently.

NHDP also collects information about symmetric 2-hop neighbors. However it specifies that if a link from a symmetric 1-hop neighbor ceases being symmetric, including while "ignored" as described above, then corresponding symmetric 2-hop neighbors are removed. This may lead to symmetric 2-hop neighborhood information being permanently removed (until further HELLO messages are received) if the link quality of a symmetric 1-hop neighbor drops below the acceptable threshold, even if only for a moment.

This specification updates NHDP, and the Optimized Link State Routing Protocol version 2 (OLSRv2) to permit retaining, but ignoring, symmetric 2-hop information when the link quality from the corresponding 1-hop neighbor drops below the acceptable threshold. This allows immediate reinstatement of the symmetric 2-hop neighbor if the link quality later improves sufficiently, thus making the symmetric 2-hop neighborhood more "robust".

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 22, 2014.

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 4
- 2. Terminology 5
- 3. Applicability Statement 5
- 4. Changes to NHDP 5
 - 4.1. Interface Information Bases 5
 - 4.2. HELLO Message Processing 6
 - 4.3. Information Base Changes 6
 - 4.4. Constraints 7
- 5. Changes to OLSRv2 7
- 6. MIB Considerations 8
 - 6.1. Updates to the State Group 9
 - 6.2. Updates to the Notification Group 10
- 7. IANA Considerations 11
- 8. Security Considerations 11
- 9. Acknowledgments 11
- 10. Normative References 11
- Authors' Addresses 12

1. Introduction

The MANET Neighborhood Discovery Protocol (NHDP) [RFC6130], Section 14, contains a link admission mechanism known as "link quality" that allows a router using that protocol to "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". Specifically, [RFC6130] permits a router to disallow consideration of some of its 1-hop neighbors, for as long as the quality of the link from that 1-hop neighbor is below an acceptable link quality threshold.

A feature of this mechanism is that while the link quality remains too low, the link information, established by the exchange of HELLO messages, is retained. Thus if the link quality later goes above the required threshold (note that a hysteresis mechanism means that two thresholds are used) then the link is immediately established and will be immediately available for use.

[RFC6130] collects not just 1-hop neighbor information, but also information about symmetric 2-hop neighbors. However [RFC6130] specifies that if a 1-hop neighbor was, but no longer is, considered symmetric, then the corresponding 2-Hop Tuples that may have been recorded for that 2-hop neighbor, are to be removed, without a retention mechanism for a (possibly temporary) loss due to link quality.

This means that if there is a short period in which link quality is too low, then when the link quality is reestablished, all 1-hop neighbor information is immediately available for use again. However, the corresponding symmetric 2-hop neighbor information has been removed, and is not available for use until restored by receipt of the next corresponding HELLO message.

This specification describes how [RFC6130] can be modified to avoid this situation, by retaining (but not using) 2-hop information, similar to what is done with 1-hop information. This modification is strictly optional, and routers that do and do not implement it can interwork entirely successfully (as they also can with different link quality specifications). In addition, by a suitable interpretation (that ignored 2-Hop Tuples are not externally advertised), this change can be invisible to any other protocols using [RFC6130], in particular [RFC7181]. However the impact on [RFC7181] when 2-Hop Tuples are not so handled is also described, in particular owing to the existence of implementations of that protocol that are not modularly separated from [RFC6130].

This specification therefore updates [RFC6130] and [RFC7181].

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].

Additionally, this document uses the terminology of [RFC6130] and [RFC7181].

3. Applicability Statement

This specification updates [RFC6130]. The optimization presented in this specification is simply permissive, as it allows retaining information which otherwise would have been removed, but does not use that information except when it could have been used by [RFC6130].

This can, in some cases, ensure that the symmetric 2-hop neighborhood is more robust against temporary link quality changes, and consequently yield a more stable network. The only other consequence of this optimization is that state for some otherwise expired 2-Hop Tuples may be maintained for longer.

This specification also updates [RFC7181]. This could be avoided by simply noting that this specification describes how the updates to [RFC6130] may be handled so as to be invisible to any other protocol using it. However as it is known that some implementations of [RFC7181] are not independent of the implementation of [RFC6130] that they use, it is useful to indicate the direct impact on [RFC7181].

A router that implements the optimization described in this specification will interoperate successfully with routers that implement [RFC6130], but do not implement this optimization.

4. Changes to NHDP

The following changes are made to [RFC6130] if using this specification. Note that while this specification is OPTIONAL, if any of these changes are made then all of these changes MUST be made.

4.1. Interface Information Bases

The 2-Hop Set is modified by adding this additional element to each 2-Hop Tuple:

N2_lost is a boolean flag, which indicates the state of the corresponding Link Tuple. If L_status = SYMMETRIC (and thus L_lost = false), then N2_lost = false. If L_SYM_time has not expired, and L_lost = false (and hence L_status = LOST), then N2_lost = true.

In all other cases, including other cases with L_status = LOST, there will be no such 2-Hop Tuples.

4.2. HELLO Message Processing

In Section 12.6 of [RFC6130] make the following changes:

- o In point 2, change "L_status = SYMMETRIC" to "L_SYM_time not expired".
- o When creating a 2-Hop Tuple, set N2_lost := L_lost.

4.3. Information Base Changes

In Section 13, replace the second bullet point by:

- o A Link Tuple's L_status changes from SYMMETRIC, L_SYM_time expires, or the Link Tuple is removed. In this case, the actions specified in Section 13.2 are performed.

and replace the paragraph after the bullet points by:

If a Link Tuple is removed, or if L_HEARD_time expires and either L_status changes from SYMMETRIC or L_SYM_time expires, then the actions specified in Section 13.2 MUST be performed before the actions specified in Section 13.3 are performed for that Link Tuple.

In Section 13.2 of [RFC6130], add the following, before all other text:

For each Link Tuple that has L_SYM_time not expired:

1. If L_SYM_time then expires, or if the Link Tuple is removed:
 1. Remove each 2-Hop Tuple for the same MANET interface with:
 - + N2_neighbor_iface_addr_list contains one or more network addresses in L_neighbor_iface_addr_list.
2. If L_status then changes from SYMMETRIC to LOST because L_lost is set to true:

1. For each 2-Hop Tuple for the same MANET interface with:
 - + N2_neighbor_iface_addr_list contains one or more network addresses in L_neighbor_iface_addr_list;

set N2_lost := true.

Also in Section 13.2 of [RFC6130], remove point 2, renumbering point 2 as point 1.

4.4. Constraints

In Appendix B, under "In each 2-Hop Tuple:" change the first bullet point to:

- o There MUST be a Link Tuple associated with the same MANET interface with:
 - * L_neighbor_iface_addr_list = N2_neighbor_iface_addr_list; AND
 - * L_SYM_time not expired; AND
 - * L_lost = N2_lost.

5. Changes to OLSRv2

If the implementation of [RFC6130] conceals from any protocol using it the existence of all 2-Hop Tuples with N2_lost = true, then no changes are required to any protocol using [RFC6130], in particular no changes are required to [RFC7181].

However if instead the implementation of [RFC6130] makes all 2-Hop Tuples visible, including those with N2_lost = true, then protocols using [RFC6130] MUST ignore such 2-Hop Tuples.

For [RFC7181], given that this protocol uses 2-hop information for MPR Set and Routing Set calculation, but not includes that information in control traffic, this means that an implementation must be (i) behaving as if a 2-Hop Tuple only exists if N2_lost=false, and (ii) as if a change of N2_lost (from false to true, or true to false) corresponds to a 2-Hop Tuple appearing or being removed. Specifically, this means behaving as if all of the following changes were to be made to [RFC7181]:

- o In Section 17.6 of [RFC7181], point 1, replace the final two bullet points with:

- * A 2-Hop Tuple with `N2_out_metric != UNKNOWN_METRIC` and `N2_lost = false` is added or removed, OR;
 - * A 2-Hop Tuple with `N2_out_metric != UNKNOWN_METRIC` has `N2_lost` changed, OR;
 - * The `N2_out_metric` of any 2-Hop Tuple with `N2_lost = false` changes, and either the flooding MPR selection process uses metric values (see Section 18.4) or the change is to or from `UNKNOWN_METRIC`.
- o In Section 17.6 of [RFC7181], point 3, replace the final two bullet points with:
 - * A 2-Hop Tuple with `N2_in_metric != UNKNOWN_METRIC` and `N2_lost = false` is added or removed, OR;
 - * A 2-Hop Tuple with `N2_in_metric != UNKNOWN_METRIC` has `N2_lost` changed, OR;
 - * The `N2_in_metric` of any 2-Hop Tuple with `N2_lost = false` changes.
 - o In Section 17.7 of [RFC7181], in the fifth bullet point, add "`and N2_lost = false`" after "`N2_out_metric != UNKNOWN_METRIC`".
 - o In Section 18.4 of [RFC7181], in the third bullet point, add "`, N2_lost = false`" after "`N2_out_metric != UNKNOWN_METRIC`".
 - o In Section 18.5 of [RFC7181], in the third bullet point, add "`, N2_lost = false`" after "`N2_in_metric != UNKNOWN_METRIC`".
 - o In Section 19.1 of [RFC7181], in the final main bullet point (marked as "(OPTIONAL)"), add "`and N2_lost = false`" after "`N2_out_metric != UNKNOWN_METRIC`".
 - o In Appendix C.7 of [RFC7181], in point 1, add "`and N2_lost = false`" after "`N2_out_metric != UNKNOWN_METRIC`".

6. MIB Considerations

This update to [RFC6130] does not change the definition of a symmetric 2-hop neighbor. It adds new information and states for each symmetric 2-hop neighbor, recorded in the Neighbor Information Base of a router and to be reflected in the appropriate tables of the corresponding NHDP-MIB module [RFC6779].

6.1. Updates to the State Group

This update introduces, to the state of each 2-Hop Tuple, the boolean flag `N2_lost`. In order to reflect this, the updates in this section are to be made to the State Group (`nhdpStateObjGrp`) of the NHDP-MIB module [RFC6779].

The DESCRIPTION of `nhdpIib2HopSetEntry` Object Type is to be updated, so as to read as follows:

```

nhdpIib2HopSetEntry OBJECT-TYPE
    SYNTAX      Nhdpiib2HopSetEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "nhdpIib2HopSetTable 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.

         (N2_neighbor_iface_addr_list,
          N2_2hop_addr, N2_lost, N2_time).

         The entries include the 2-hop neighbor addresses, which
         act as the table index, and associated symmetric 1-hop
         neighbor address set, designated through nhdpDiscIfIndex,
         an expiration time, and a flag indicating if the 1-hop
         neighbor, through which this 2-hop neighbor is reachable,
         is considered lost due to link quality, or not.

         The nhdpIfIndex in the INDEX is the interface index of
         the local interface through which these 2-hop addresses
         are accessible. The nhdpDiscIfIndex in the INDEX
         represents the 1-hop neighbor interface through which
         these 2-hop neighbor addresses are reachable."
    REFERENCE
        "RFC 6130 - Mobile Ad Hoc Network (MANET) Neighborhood
         Discovery Protocol (NHDP), Clausen, T., Dearlove,
         C., and J. Dean, April 2011"
    INDEX { nhdpIfIndex,
            nhdpDiscIfIndex,
            nhdpIib2HopSetIpAddressType,
            nhdpIib2HopSetIpAddress
          }
    ::= { nhdpIib2HopSetTable 1 }

```

The SEQUENCE of `Nhdpiib2HopSetEntry` is to be updated, so as to read as follows:

```

NhdpIib2HopSetEntry ::=
  SEQUENCE {
    nhdpIib2HopSetIpAddressType
      InetAddressType,
    nhdpIib2HopSetIpAddress
      InetAddress,
    nhdpIib2HopSetIpAddrPrefixLen
      InetAddressPrefixLength,
    nhdpIib2HopSet1HopIfIndex
      NeighborIfIndex,
    nhdpIib2HopSetN2Time
      TimeStamp,
    nhdpIib2HopSetN2Lost
      TruthValue
  }

```

The nhdpIib2HopSetN2Lost OBJECT-TYPE is to be defined as follows:

```

nhdpIib2HopSetN2Lost OBJECT-TYPE
  SYNTAX      TruthValue
  MAX-ACCESS  read-only
  STATUS      current
  DESCRIPTION
    "nhdpIib2HopSetN2Lost corresponds to N2_lost of NHDP and
     is a boolean flag, describing if for a 2-Hop Tuple, the
     corresponding Link Tuple currently is considered lost
     due to link quality."
  REFERENCE
    "draft-dearlove-manet-nhdp-optimization-01"
  ::= { nhdpIib2HopSetEntry 5}

```

6.2. Updates to the Notification Group

This update introduces an additional state for each 2-Hop Tuple. Whereas [RFC6130] has two states for 2-Hop Tuples, 'up' (a 2-Hop Tuple exists) and 'down' (a 2-Hop Tuple expires), this update introduces a third state for a 2-Hop Tuple: it exists, but (due to the link quality of the link to the corresponding 1-Hop neighbor) is not currently considered.

To reflect this, the SYNTAX and DESCRIPTION of nhdp2HopNbrState OBJECT-TYPE are to be updated, so as to read as follows:

```
nhdp2HopNbrState OBJECT-TYPE
  SYNTAX  INTEGER {
            down(0),
            up(1),
            notconsidered(2)
          }
  MAX-ACCESS  read-only
  STATUS      current
  DESCRIPTION
    "NHDP 2-hop neighbor states. In NHDP, it is not necessary
    to remove Protocol Tuples from Protocol Sets at the
    exact time indicated, only to behave as if the Protocol
    Tuples were removed at that time. This case is indicated
    here as 'down(0)'; otherwise, it is either 'up(1)', if
    N2_lost for the 2-Hop Tuple is equal to false, or
    'notconsidered(2)' otherwise."
 ::= { nhdpNotificationsStates 2 }
```

7. IANA Considerations

This document has no actions for IANA.

[This section may be removed by the RFC Editor.]

8. Security Considerations

The update to [RFC6130] enables the retention and reuse of some information collected by that protocol, for only the duration that it could have been used in any case. As such, this protocol introduces no new security considerations to an implementation of [RFC6130] or of any other protocol that uses it, such as [RFC7181].

9. Acknowledgments

The authors would like to thank Liz Cullen (BAE Systems) for first illustrating the issue addressed in this specification.

10. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6130] Clausen, T., Dean, J., and C. Dearlove, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)",

RFC 6130, April 2011.

[RFC6779] Herberg, U., Cole, R., and I. Chakeres, "Definition of Managed Objects for the Neighborhood Discovery Protocol", RFC 6779, October 2012.

[RFC7181] Clausen, T., Dearlove, C., Jacquet, P., and U. Herberg, "The Optimized Link State Routing Protocol version 2", RFC 7181, April 2014.

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
URI: <http://www.baesystems.com/>

Thomas Heide Clausen
LIX, Ecole Polytechnique

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

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: August 29, 2015

Y. Yi
S. Lee
University of California, Los Angeles
W. Su
The Boeing Company
M. Gerla
A. Colin de Verdiere
University of California, Los Angeles
February 25, 2015

On-Demand Multicast Routing Protocol (ODMRP) for Ad Hoc Networks
draft-gerla-manet-odmrp-05

Abstract

The On-Demand Multicast Routing Protocol (ODMRP) is a multicast routing protocol designed for ad hoc networks with mobile hosts. ODMRP is a mesh-based, rather than a conventional tree-based, multicast scheme and uses a forwarding group concept (only a subset of nodes forwards the multicast packets via scoped flooding). It applies on-demand procedures to dynamically build routes and maintain multicast group membership, without relying on pre-existing unicast routing protocols. ODMRP is well suited for ad hoc wireless networks with mobile hosts where bandwidth is limited, topology changes frequently and rapidly, and power is constrained.

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, 2015.

Copyright Notice

Copyright (c) 2015 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
1.1.	Motivation and Experiments	4
2.	Terminology and Notation	5
2.1.	Notation	5
2.2.	Terminology	5
3.	Applicability Statement	7
4.	Protocol Overview and Functioning	7
4.1.	Routers and Interfaces	8
4.2.	Information Base Overview	8
4.3.	Signaling Overview	9
4.4.	Overview	9
5.	Parameters and Constants	11
5.1.	Router Parameters	11
5.2.	Interface Parameters	11
6.	Sequence Numbers	12
7.	Packets and Messages	12
7.1.	Join Query Format	12
7.2.	Join Reply Format	13
8.	RFC5444 Encoding	13
8.1.	Join Query Encoding	14
8.2.	Join Reply Encoding	14
9.	Information Bases	15
9.1.	Local Interface Set	15
9.2.	Neighbor Interface Set	15
9.3.	Multicast Routing Set	16
9.4.	Forwarding Table	16
9.5.	Pending Acknowledgements	17
9.6.	Pre-acknowledgements	18
9.7.	Blacklist	18
9.8.	Sent JQ set	19
10.	Protocol Details	19
10.1.	Join Query	20

10.1.1.	Invalid Join Queries	20
10.1.2.	Join Query Generation	20
10.1.3.	Join Query Processing	21
10.1.4.	Join Query Forwarding	22
10.2.	Join Reply	22
10.2.1.	Invalid Join Replies	23
10.2.2.	Join Reply Generation	23
10.2.3.	Join Reply Processing	23
10.2.4.	Join Reply Forwarding	25
10.2.5.	Join Reply Transmission	26
10.3.	Forwarding Group Maintenance	27
10.4.	Message Transmission	27
11.	Unidirectional Links Handling	27
12.	SMF considerations	29
13.	IGMP and MLD considerations	29
14.	Multicast Packet Forwarding	29
15.	Security Considerations	30
15.1.	Confidentiality	30
15.2.	Integrity	30
15.3.	Channel Overload	31
16.	IANA Considerations	31
16.1.	Join Query Registries	31
16.2.	Join Reply Registries	32
17.	Acknowledgements	33
18.	References	33
18.1.	Normative References	33
18.2.	Informative References	34
Appendix A.	Illustrations	35
A.1.	Join Query Message	35
A.2.	Join Reply Message	36
Authors' Addresses		37

1. Introduction

This document describes the On-Demand Multicast Routing Protocol (ODMRP) [ODMRP-Journal]. ODMRP applies "on-demand" routing techniques to avoid channel overhead and improve scalability. It uses the concept of "forwarding group" [FGMP], a set of nodes responsible for forwarding multicast data, to build a forwarding mesh for each multicast group. By maintaining and using a mesh instead of a tree, the drawbacks of multicast trees in mobile wireless networks (e.g., intermittent connectivity, traffic concentration, frequent tree reconfiguration, non-shortest path in a shared tree, etc.) are avoided. A soft-state approach is taken to maintain multicast group members, meaning that no explicit control message is required to leave the group. ODMRP does not rely on any unicast routing protocol: in particular, it can operate in conjunction with both reactive and proactive unicast routing protocols.

1.1. Motivation and Experiments

The main rationale for ODMRP is its potential to reduce control and traffic overhead in certain MANET deployments, typically where multicast traffic is relatively sparse. While this protocol has been extensively studied in simulations, it does not yet benefit from sufficient operational experience in order to be considered for Standards Track. In addition to general operational experience such as interoperability testing, this specification is intended to collect data on the following points:

- o As a multicast routing protocol for MANET, ODMRP can be compared with [RFC6621], but can also be used in conjunction, taking advantage of its Duplicate Packet Detection and optimized flooding mechanisms. The rationale behind ODMRP is that, with sparser traffic, and in particular less sources, ODMRP should reduce the control overhead and number of data packets transmitted by making use of Forwarding Groups. This hypothesis should be validated, and experiments and operational deployments demonstrating the scenarios in which ODMRP performs better, or worse, than [RFC6621] should be performed.
- o The potential scope of deployment of ODMRP should be assessed, particularly in comparison to other MANET protocols.
- o Default values and guidelines for the parameters described in Section 5 should be provided, based on operational experience gathered from implementing and deploying this specification.
- o The feasibility of implementing ODMRP in common MANET situations should be examined. In particular, it should be determined if a

linux user space implementation is possible.

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].

This document also makes use of the terminology defined in [RFC5444]. Additionally, it uses the notation defined in Section 2.1, and the terminology defined in Section 2.2.

2.1. Notation

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

MsgType.field

where:

MsgType - is the type of message (e.g., JQ or JR);

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

Furthermore, the following notational conventions are used:

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 and only if the value of the left side (c) is equal to the value of the right side (d)

[x] a list containing x as its only element

The different messages, their fields and their meaning are described in Section 7.

2.2. Terminology

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

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

Neighbor (ODMRP) Router - An ODMRP Router A is a neighbor of another ODMRP Router B if B can receive control messages from A according to this specification. This relationship is not necessarily symmetrical.

Neighbor (ODMRP) Interface - An interface X of an ODMRP Router A is a neighbor relative to interface Y of ODMRP Router B if B can receive control messages sent by A over the link X - Y. The link, and this relationship, are not necessarily symmetrical.

Multicast session - The entity defined by a (multicast group, source) pair, representing the group to which a source sends multicast packets.

Forwarding group - A group of ODMRP Routers participating in multicast packet forwarding for a given Multicast Session. In particular, the Forwarding Group is constituted of the Multicast Source, the Multicast Receivers and the Intermediate Routers.

Multicast Receiver - An ODMRP Router is a Multicast Receiver, relative to a given Multicast Session, if it subscribes to the Multicast Session in order to receive data packets sent by the Multicast Source.

Intermediate Router - An Intermediate Router is an ODMRP Router that is a member of a Forwarding Group without being a Multicast Receiver. In other words, it joined the Forwarding Group to transmit control and data traffic between the Multicast Source and the Multicast Receivers.

Join Query - The control message sent by Multicast Sources to establish and update group memberships and routes.

Join Reply - The control message sent by Multicast Receivers and forwarded by Intermediate Routers to build the Forwarding Group according to group membership information.

Upstream - An ODMRP Router (A) is said to be "upstream" compared to another ODMRP Router (B), relatively to a given Multicast Session, if A is on the path, which is discovered by a Join Query-Join Reply exchange and used by data packets, between B and the Multicast Source. In other words, any data packet sent within the Multicast Session has to transit through A before reaching B.

Downstream - An ODMRP Router (A) is said to be "downstream" compared to another ODMRP Router (B), relatively to a given Multicast Session, if B is on the path which is discovered by a Join Query-Join Reply exchange and used by data packets, between A and the Multicast Source. In other words, A is "downstream" from B if B is "upstream" from A.

3. Applicability Statement

This protocol is a multicast routing protocol, intended for use in Mobile Ad Hoc Networks (MANETs). MANETs generally have constrained resources (processing power, battery, etc.) and very dynamic topologies. With ODMRP, routing state is installed and maintained in an on-demand fashion, which avoids the issue of frequent tree reconfiguration seen with more classic multicast routing protocols.

ODMRP does not rely on the use of any unicast routing protocol, whether reactive or proactive, but MAY be used coinjointly with such protocols, such as [RFC7181] or [RFC3561]. Additionally, ODMRP can run in conjunction with [RFC6621], and take advantage of any optimized flooding mechanism used in the network, such as those offered by SMF, to disseminate Join Query messages as described in Section 12.

4. Protocol Overview and Functioning

The objective of this protocol is to allow each ODMRP Router to:

- o Build a Forwarding Group only when it has data traffic to send to a Multicast group.
- o Maintain the Forwarding Group for as long as necessary, until there is no more data to be sent to the Multicast group.
- o Join any Forwarding Group, in order to receive multicast data packets from the corresponding multicast source. The decision to join a given Forwarding Group is triggered by Multicast membership information relative to the corresponding Multicast session. Such information can be received from other protocols, such as IGMP

[RFC3376] and MLD [RFC3810].

4.1. Routers and Interfaces

Each ODMRP Router MUST be provisioned with at least one ODMRP Interface, and keep a list of all these interfaces, as described in Section 9. The management of these interfaces (addition, deletion, re-addressing of any interface) is out of scope for this document.

4.2. Information Base Overview

Protocol state is recorded in eight distinct information sets: the Local Interface Set, the Neighbor Interface Set, the Multicast Routing Set, the Forwarding Table, the Pending Acknowledgement Set, the Pre-acknowledgement Set, the Blacklist and the Sent JQ Set. With the exception of the Local Interface Set, all these information sets are both used and updated by this protocol.

The Local Interface Set records a list matching each ODMRP interface of this router to the addresses in use for this interface. This set is used, but not updated by this protocol.

The Neighbor Interface Set records all the known addresses of neighbor ODMRP interfaces, by way of recording data from received JQ messages. This set can also be updated by other protocols with knowledge of neighbor interfaces, such as [RFC6130].

The Multicast Routing Set contains tuples, each representing the address of a multicast group, the address of a source sending data to this multicast group, and the next hop towards the multicast source.

The Forwarding Table contains tuples, each representing a given Multicast session for which the ODMRP Router forwards packets.

The Pending Acknowledgement Set contains tuples, each corresponding to a Join Reply message which has been sent by this Router and is waiting for an acknowledgement from a chosen upstream Router.

The Pre-acknowledgement Set contains tuples, representing overheard Join Reply messages, that are not destined to this Router but may pre-acknowledge a future Join Reply from this Router.

The Blacklist contains tuples, corresponding to neighbor ODMRP Routers, with which connectivity has been detected to be unidirectional.

The Sent JQ Set matches interfaces of this Router with the address carried by the last JQ message to have transited through that

interface.

4.3. Signaling Overview

This protocol generates and processes the following routing messages:

Join Query - Generated by an ODMRP Router while it has data packets to send to a multicast group, and flooded periodically to maintain the Forwarding Group necessary to deliver these data packets. A Join Query message hence advertises a Multicast Session, and contains:

- * The multicast group address
- * The source address
- * A sequence number
- * The last address used by this interface to send a JQ message

Join Reply - Generated by an ODMRP Router belonging to a multicast session, in reply to a Join Query message advertising this multicast session (corresponding Join Query), then forwarded by ODMRP Routers belonging to the same multicast session along the reverse path to the multicast source. A Join Reply message contains:

- * The multicast group address and source address, identifying the multicast session
- * The sequence number carried by the corresponding Join Query
- * The address of the next hop on the path towards the multicast session source

4.4. Overview

The objectives of this protocol are achieved, for each ODMRP Router, by the following operations:

- o When having data to send to a multicast group, for which no Forwarding Group is already established, an ODMRP Router generates a Join Query and transmits it over all of its ODMRP Interfaces. It then periodically repeat this process, until it has no more data to send to the multicast group.
- o Upon receiving a Join Query, an ODMRP Router installs or refreshes a tuple in the Multicast Routing Set indicating the reverse path

towards the source of the Join Query, then considers it for forwarding, according to the forwarding mechanism specified for the network.

- o If this Router belongs to, i.e., has attached hosts which have subscribed to, the multicast session that the Join Query advertises, it originates a Join Reply and transmits it over all of its ODMRP Interfaces.
- o Upon receiving a Join Reply, an ODMRP Router inspects the next hop address carried by this packet:
 - * If it corresponds to the address of an interface of this Router, and if this Router has a tuple in its Multicast Routing Set, corresponding to the advertised source, the ODMRP Router belongs to the Forwarding Group for the Multicast Session. Consequently, it installs or refreshes the corresponding entry in its Forwarding Table. It then considers the Join Reply for forwarding, according to the forwarding mechanism specified for the network.
 - * Otherwise, it verifies if any Pending Acknowledgement tuple corresponds to this Join Reply and marks each such tuple as acknowledged. It silently discards the Join Reply.
- o After sending a Join Reply, addressed to an upstream router A, an ODMRP Router looks in its Pre-acknowledgement Set for a corresponding Overheard tuple.
 - * If such a tuple exists, the Overheard tuple is discarded and no further action is taken.
 - * Otherwise, i.e., if the Pre-acknowledgement Set does not contain any corresponding Overheard tuple, it creates a Pending Acknowledgement tuple in the Pending Acknowledgement Set. If this tuple expires without being acknowledged, the link with router A is considered unidirectional: it is blacklisted, and the current router MAY try other means of joining the Forwarding Group.
- o While it has data to send to the multicast group, an ODMRP Router periodically originates a Join Query and transmits it to all of its neighbors, in order to maintain the Forwarding Group.

5. Parameters and Constants

This specification uses both Router parameters, described in Section 5.1, and per-interface parameters, described in Section 5.2.

5.1. Router Parameters

This specification uses the following Router parameters:

`ROUTE_REFRESH_INTERVAL` - is the interval between two periodic Join Queries sent by a Multicast Source

`FG_TIMEOUT` - is the minimum time a Forwarding Tuple SHOULD be kept in the Forwarding Table after it was last refreshed

5.2. Interface Parameters

This specification uses the following interface parameters:

`ROUTE_TIMEOUT` - is the minimum time a Routing Tuple SHOULD be kept in the Routing Set after it was last refreshed

`JR_RETRIES` - is the number of times an ODMRP Router SHOULD attempt to retransmit a given Join Reply before declaring the link with the upstream neighbor interface unidirectional

`ACK_TIMEOUT` - is the time after which a Pending Tuple expires and MUST be considered invalid, as well as trigger the appropriate action according to Section 11

`PRE_ACK_TIMEOUT` - is the time after which an Overheard Tuple expires and MUST be considered invalid

`LOCAL_ADDRESS_TIMEOUT` - is the time after which a sent JQ tuple expires and MUST be considered invalid. This parameter SHOULD be less than the time an interface address is expected to be in use for the corresponding communication medium

`NEIGHBOR_ADDRESS_TIMEOUT` - is the time after which an address tuple of a neighbor interface tuple expires and MUST be considered invalid. This parameter SHOULD be less than the time an interface address is expected to be in use for the corresponding communication medium

6. Sequence Numbers

Each ODMRP Router maintains a single sequence number, which must be included in each Join Query message it generates. Each ODMRP Router MUST make sure that no two Join Query messages are generated with the same sequence number, and MUST generate sequence numbers such that these are monotonically increasing. This sequence number is used as freshness information for when comparing routes to the ODMRP Router having generated the message.

However, with a limited number of bits for representing sequence numbers, wrap-around (that the sequence number is incremented from the maximum possible value to zero) will occur. To prevent this from interfering with the operation of the protocol, the following MUST be observed. The term `MAX_SEQ_NUM` 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:

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

7. Packets and Messages

This section describes the protocol messages generated and processed by ODMRP, according to the notations defined in Section 2. The objective of this section is to specify the content and meaning of each message. The specifics of the encoding of these messages, including the exact type and length of each field, in accordance with [RFC5444], are described in Section 8.

7.1. Join Query Format

A Join Query (JQ) message has the following fields:

`JQ.AddressLength` encodes the length of the addresses carried by this message as follows:

$$\text{JQ.AddressLength} := \text{the length of an address in octets} - 1$$

`JQ.MulticastGroupAddress` encodes the address of the Multicast Group, to which this Join Query is addressed

JQ.SourceAddress encodes an address of the source of this Join Query

JQ.SequenceNumber encodes the sequence number (see Section 6) of the ODMRP Router, generating the Join Query message

JQ.LastAddress encodes the address set as the source address of the last IP datagram sent through the same interface and containing a JQ message, or of the IP datagram carrying this JQ message if no such datagram is known

7.2. Join Reply Format

A Join Reply (JR) message has the following fields:

JR.AddressLength encodes the length of the addresses carried by this message as follows:

JR.AddressLength := the length of an address in octets - 1

JR.MulticastGroupAddress encodes the address of the Multicast Group, to which this Join Reply is addressed

JR.AckRequired is a boolean flag. When set ('1'), it specifies that the recipient of the Join Reply MUST acknowledge its reception by a sending Join Reply message. If cleared ('0'), the recipient of this message MAY suppress it's Join Reply transmission, according to Section 10

JR.SourceAddress encodes the address of the Source of the Multicast Session

JR.SequenceNumber encodes the sequence number (see Section 6) of the corresponding Join Query message

JR.NextHopAddress encodes the the address of the next hop on the path towards the source of the multicast session

8. RFC5444 Encoding

This section describes the encoding of ODMRP messages using [RFC5444].

8.1. Join Query Encoding

This protocol defines the Join Query message type. Hence, according to [RFC5444], all Join Query messages are generated, processed and transmitted following this specification. Table 1 shows the mapping between the Join Query elements described in Section 7.1 and their encoding. All elements described in Table 1 MUST be included in every Join Query message, with the exception of the JQ.LastAddress element. JQ.LastAddress MAY be omitted in a given JQ message if it corresponds to the source address of the IP datagram containing this message.

JQ Element	RFC5444 Element
JQ.AddressLength	<msg-addr-length>
JQ.SourceAddress	<msg-orig-addr>
JQ.MulticastGroupAddress	Address in address block + TLV
JQ.SequenceNumber	<msg-seq-num>
JQ.LastAddress	Address in address block + TLV

Table 1: Join Query Message Elements

8.2. Join Reply Encoding

This protocol defines the Join Reply message type. Hence, according to [RFC5444], all Join Reply messages are generated, processed and transmitted following this specification. Table 2 shows the mapping between the Join Reply elements described in Section 7.2 and their encoding. With the exception of the ACKREQUIRED TLV, all elements described in Table 2 MUST be included in every Join Reply message.

JR Element	RFC5444 Element
JR.AddressLength	<msg-addr-length>
JR.SourceAddress	<msg-orig-addr>
JR.MulticastGroupAddress	Address in address block + TLV
JR.SequenceNumber	<msg-seq-num>
JR.NextHopAddress	Address in address block + TLV
JR.AckRequired	ACKREQUIRED TLV

Table 2: Join Reply Message Elements

9. Information Bases

Each router maintains an Information Base, containing a Local Interface Set, a Neighbor Interface Set, a Multicast Routing Set, a Forwarding Table, a Pending Acknowledgement Set, a Pre-Acknowledgement Set, a Blacklist and a Sent JQ Set, as described in the following sections. These information sets are given so as to facilitate description of message generation, forwarding and processing rules. In particular, an implementation may chose any representation or structure for when maintaining this information.

9.1. Local Interface Set

The Local Interface Set records a list of all the interfaces of the ODMRP Router, which participate in the operations of this protocol; that is, over which ODMRP control messages are exchanged, according to this specification. Each tuple of the Interface Set, or Interface Tuple, is as follows:

(I_interface, I_interface_address_list)

Where:

I_interface - The local ODMRP Interface

I_interface_address_list - The list of addresses used by the local interface in the operations of this protocol.

9.2. Neighbor Interface Set

The Neighbor Interface Set records the known addresses of Neighbor ODMRP Interfaces. It is used by this protocol, and can be updated by this protocol or by any other suitable protocol in operation that provides the necessary information, such as NHDP [RFC6130]. Each neighbor interface tuple is as follows:

(N_interface_address_list)

Where:

N_interface_address_list - Is an unordered list of at least one address tuple (i_addr, i_addr_exp_time), where:

i_addr - is a known address of the Neighbor Interface, i.e., an address that was set as the sender of an IP datagram sent through this interface

`i_addr_exp_time` - is the time at which the address tuple MUST be considered expired and thus MUST NOT be taken in considerations for the operations of this protocol

A neighbor interface tuple that contains no valid (i.e., non-expired) address tuple MUST be considered expired and MUST NOT be taken in considerations for the operations of this protocol

9.3. Multicast Routing Set

The Multicast Routing Set contains Routing Tuples, indicating the path towards Multicast Sources, and containing the following fields:

```
(R_source, R_next_hop, R_local_interface,  
R_seq_num, R_exp_time)
```

Where:

`R_source` - is the address of the Multicast Source.

`R_next_hop` - is an address of the next hop along the path to the Multicast Source, i.e., an address of one of the interfaces of the neighbor ODMRP Router, from which the last valid Join Query message from this source was received, as recorded by the packet containing this Join Query.

`R_local_interface` - is the local interface, through which the next hop can be reached.

`R_seq_num` - corresponds to the `JQ.SequenceNumber` of the last valid Join Query originated by the Multicast Source and received by this ODMRP Router.

`R_exp_time` - is the time at which the tuple MUST be considered expired and thus MUST NOT be taken into consideration by the operations of this protocol.

9.4. Forwarding Table

The Forwarding Table contains Forwarding Tuples, representing Multicast Sessions for which the ODMRP Router forwards messages, i.e., the ODMRP Router is part of these Multicast Sessions' Forwarding Groups. These tuples are as follows:

```
(F_multicast_group, F_multicast_source,  
F_seq_num, F_exp_time)
```

Where:

F_multicast_group - is the address of the Multicast Group of the Multicast Session, for which the ODMRP Router forwards messages.

F_source - is the address of the Multicast Source of the Multicast Session, for which the ODMRP Router forwards messages.

F_seq_num - is the sequence number, corresponding to the last Join Query sent by the multicast source for the multicast session.

F_exp_time - is the time at which the tuple MUST be considered expired and thus MUST NOT be taken into consideration by the operations of this protocol.

9.5. Pending Acknowledgements

The Pending Acknowledgements Set contains Pending Acknowledgement tuples, representing Join Reply messages that are waiting to be acknowledged by the selected upstream Forwarding Group member. These tuples are as follows:

```
(P_multicast_group, P_multicast_source, P_seq_num,  
P_local_interface, P_next_hop, P_nth_time, P_exp_time)
```

Where:

P_multicast_group - is the JR.MulticastGroupAddress carried in the Join Reply awaiting acknowledgement (henceforth corresponding Join Reply).

P_multicast_source - is the JR.SourceAddress field carried in the corresponding Join Reply.

P_seq_num - is the JR.SequenceNumber field of the corresponding Join Reply.

P_next_hop - is the JR.NextHopAddress field of the corresponding Join Reply.

P_local_interface - is the local interface, through which the Join Reply was sent.

P_nth_time - corresponds to the number of times this Join Reply has been previously sent without being acknowledged.

P_exp_time - is the time at which this tuple MUST be considered expired.

P_acknowledged - is a boolean indicating whether the corresponding Join Reply has been acknowledged.

9.6. Pre-acknowledgements

The Pre-acknowledgements Set contains Overheard Tuples, corresponding to Join Reply messages, which have been sent by neighbors of this ODMRP Router but do not contain an address of this Router and do not acknowledge any tuple in the Pending Acknowledgement Set. The Overheard Tuples are as follows:

(O_multicast_group, O_multicast_source, O_seq_num,
O_originator, O_exp_time)

Where:

O_multicast_group - is the JR.MulticastGroupAddress carried in the overheard Join Reply.

O_multicast_source - is the JR.SourceAddress field carried in the corresponding Join Reply.

O_seq_num - is the JR.SequenceNumber field of the corresponding Join Reply.

O_originator - is the address of the ODMRP Router's interface which has sent the Join Reply.

O_exp_time - is the time at which this tuple expires MUST be considered invalid.

9.7. Blacklist

The Blacklist contains Blacklisted Tuples, corresponding to neighbor ODMRP Router interfaces, with which connectivity has been detected to be unidirectional, e.g., which have not acknowledged Join Replies from this Router, as specified in Section 10. In other words, a Blacklisted Tuple corresponds to a link between one local interface and one neighbor interface which has been detected to be unidirectional or broken. The Blacklist Tuples are as follows:

(B_neighbor_interface, B_local_interface, B_exp_time)

Where:

B_neighbor_interface_address_list - is a list of addresses belonging to the blacklisted interface.

B_local_interface - is the interface of this ODMRP router over which packets from the blacklisted interface were received.

B_exp_time - is the time at which this tuple expires and MUST be considered invalid.

9.8. Sent JQ set

The Sent JQ Set contains tuples matching transmitted (generated or relayed) Join Queries with interfaces addresses. Each of its tuples contains the source address, multicast group address and sequence number uniquely identifying a JQ message, as well as an interface and the address of that interface that was advertised when transmitting the packet containing the Join Query. More precisely, given a transmitted Join Query and an interface over which it was transmitted, a tuple of this set, or Sent JQ Tuple, is as follows:

Where:

S_interface - is the local interface, through which the JQ message was sent

S_interface_address - is the address of the ODMRP interface that was set as the packet source

S_exp_time - is the time at which this tuple expires and MUST be considered invalid.

10. Protocol Details

This protocol generates and processes Join Query and Join Reply messages, according to the operations described in the following sections. This section uses the additional notation and variables:

previous-hop-address - refers to the address of the neighbor ODMRP interface recorded by the source address field of the IP datagram carrying the message currently being processed (Join Query or Join Reply)

this Router - refers to the ODMRP Router generating, processing or forwarding the message (Join Query or Join Reply)

Receiving Interface (receiving-interface) - refers to the local ODMRP Interface, over which the message currently being processed was received

10.1. Join Query

A Join Query is generated by an ODMRP Router, which has data to send to a multicast group, for which no multicast session has been initialized. Join Queries are then periodically originated by the ODMRP Router while it has data to send to the multicast group.

10.1.1. Invalid Join Queries

A Join Query, received by an ODMRP Router, is invalid and MUST be discarded without processing (and in particular, MUST NOT be considered for forwarding) if any of these conditions applies:

- o The address length carried by the Join Query (see Section 7) differs from the length of the addresses of this Router
- o The Multicast Routing Set of this Router contains a Multicast Routing tuple, for which:
 - * `R_multicast_source = JQ.SourceAddress`, and
 - * `R_seqnum > JQ.SequenceNumber` or `R_seqnum = JQ.SequenceNumber`
- o `JQ.SourceAddress` is an address of an interface of this Router
- o The Blacklist contains a Blacklisted Tuple, for which
 - * `previous-hop-address` is contained in `B_neighbor_interface_address_list`
 - * `B_local_interface = receiving-interface`

10.1.2. Join Query Generation

A Join Query is generated according to Section 7 with the following content:

- o `JQ.AddressLength` set to the length of the addresses of this Router minus 1, as specified in Section 7

- o JQ.MulticastGroupAddress set to the address of the multicast group, to which this Router is sending data
- o JQ.SourceAddress set to an address of this ODMRP Router
- o JQ.SequenceNumber set to the current sequence number of this Router, as specified in Section 6

10.1.3. Join Query Processing

Upon receiving a valid Join Query message, an ODMRP Router proceeds as follows:

1. Find the neighbor interface tuple such as N_interface_address_list contains an address tuple with i_addr = previous-hop-address, and update the address tuple such as i_addr_exp_time := current-time + NEIGHBOR_ADDRESS_TIMEOUT
2. If no such tuple exists, create one with:
 - * N_interface_address_list := [(previous-hop-address, current-time + NEIGHBOR_ADDRESS_TIMEOUT)]
3. Find the Routing Tuple which satisfies: R_source = JQ.SourceAddress
4. If no such tuple exists, create a Routing Tuple with the following fields:
 - * R_source := JQ.SourceAddress
 - * R_next_hop := previous-hop
 - * R_local_interface := receiving-interface
 - * R_seq_num := JQ.SequenceNumber
 - * R_exp_time := current-time + ROUTE_TIMEOUTand insert this tuple in the Routing Set
5. Else, i.e., if such a tuple exist, update it as follows:
 - * R_next_hop := previous-hop
 - * R_seq_num := JQ.SequenceNumber

- * R_local_interface := receiving-interface
- * R_exp_time := current-time + ROUTE_TIMEOUT

6. Consider the Join Query for forwarding, according to Section 10.1.4
7. If this Router is a member of the Multicast Group, addressed by JQ.MulticastGroupAddress, create a new Join Reply according to Section 10.2 and transmit it to all of this Router's neighbors

10.1.4. Join Query Forwarding

This section defines the following additional variables:

this-interface - is the ODMRP interface being considered

packet-source-address - is the source address of the outbound IP datagram carrying the JQ message being transmitted

For each ODMRP interface over which a JQ message is to be transmitted, a router MUST proceed as follows:

- o Find the corresponding sent JQ tuple in the sent JQ set, such as S_interface = this-interface
- o If no such tuple exists, create one with:
 - * S_interface := this-interface
 - * S_interface_address := packet-source-address
 - * S_exp_time := current-time + LOCAL_ADDRESS_TIMEOUT
- o Set JQ.LastAddress to S_interface_address
- o Update the corresponding sent JQ tuple such as:
 - * S_interface_address = packet-source-address
 - * S_exp_time = current-time + LOCAL_ADDRESS_TIMEOUT

10.2. Join Reply

A Join Reply is generated by an ODMRP Router when it receives a Join Query such that at least one host attached to the ODMRP Router is a member of the Multicast Session advertised by the Join Query. This section makes use of the variable "new-jr", which is a boolean flag

set to TRUE if the Join Reply being processed contains more recent data than in the current information base. It has an initial value of FALSE.

10.2.1. Invalid Join Replies

A Join Reply, received by an ODMRP Router, is invalid and MUST be discarded without processing (and in particular, MUST NOT be considered for forwarding) if:

- o The address length carried by the Join Reply (see Section 7) differs from the length of the address of the ODMRP Router
- o There exists a Forwarding Tuple in this Router's Forwarding Group table, such as:
 - * F_source = JR.MulticastSourceAddress
 - * F_seq_num > JR.SequenceNumber

10.2.2. Join Reply Generation

An ODMRP Router MUST generate a Join Reply in response to a received Join Query (henceforth "corresponding Join Query"), if at least one host attached to this Router is a member of the Multicast Session, advertised by the Join Query. A Join Reply is generated according to Section 7 with the following content:

- o JR.AddressLength is set to the length of the address of this router minus 1, as specified in Section 7
- o JR.MulticastGroupAddress is set to JQ.MulticastGroupAddress for the corresponding Join Query
- o JR.SourceAddress is set to JQ.SourceAddress for the corresponding Join Query
- o JR.SequenceNumber is set to JQ.SequenceNumber for the corresponding Join Query
- o JR.NextHopAddress is set to the source address of the IP datagram containing the Join Query message

10.2.3. Join Reply Processing

Upon receiving a valid Join Reply, an ODMRP Router proceeds as follows:

1. If JR.NextHopAddress corresponds to an address recorded in the Local Interface Set of this ODMRP Router:
 1. Find the Forwarding Tuple (henceforth Matching Forwarding Tuple) such that:
 - + F_multicast_group = JR.MulticastGroupAddress
 - + F_multicast_source = JR.MulticastSourceAddress
 2. If no such tuple exists, insert in the Forwarding Table a new Forwarding Tuple such that:
 - + F_multicast_group = JR.MulticastGroupAddress
 - + F_multicast_source = JR.MulticastSourceAddress
 - + F_seq_num = JR.SequenceNumber
 - + F_exp_time = current-time + FG_TIMEOUTAnd set new-jr to TRUE
 3. Otherwise, the variable "new-jr" is set to TRUE if JR.SequenceNumber > F_seq_num, and to FALSE otherwise. Then, the pre-existing Matching Forwarding Tuple is updated as follows:
 - + F_seq_num := JR.SequenceNumber
 - + F_exp_time := current-time + FG_TIMEOUT
 4. If new-jr = TRUE or if JR.AckRequired is set the Join Reply is considered for forwarding. Otherwise, it is not processed further; in particular, it MUST NOT be considered for forwarding.
2. Otherwise, find the Multicast Routing Tuple in the Routing Set (henceforth "Matching Multicast Routing Tuple"), such as:
 - * R_source = JR.SourceAddress
 - * R_seq_num <= JR.SequenceNumberIf previous-hop-address = R_next_hop, then:
 3. If the Pending Acknowledgement Set contains a Pending Tuple (henceforth "Matching Pending Tuple") such as:

- + P_multicast_group = JR.MulticastAddress
- + P_multicast_source = JR.SourceAddress
- + P_seq_num = JR.SequenceNumber
- + P_next_hop = previous-hop-address

The Matching Pending Tuple MUST be updated as follows:

- + P_acknowledged = TRUE
- + P_exp_time = EXPIRED

The Join Reply is not processed further, and in particular MUST NOT be considered for forwarding

4. Otherwise, if the Pre-Acknowledgement Set does not contain any Overheard Tuple such as:

- + O_multicast_group = JR.MulticastGroupAddress
- + O_multicast_source = JR.SourceAddress
- + O_seq_num = JR.SequenceNumber
- + O_originator = previous-hop-address

Insert a tuple with these fields, and O_exp_time = current-time + PRE_ACK_TIMEOUT in the Pre-Acknowledgement Set. The Join Reply is not processed further, and in particular MUST NOT be considered for forwarding

3. Otherwise, the Join Reply is silently discarded without further processing

10.2.4. Join Reply Forwarding

A Join Reply, considered for forwarding, MUST be updated as follows:

- o Find the Matching Routing Tuple, such that:
 - * R_source = JR.MulticastSourceAddress
 - * R_seq_num <= JR.SequenceNumber

- o If no such tuple exists, then the Join Reply is not processed further, and in particular MUST NOT be forwarded
- o Otherwise, set JR.NextHop to R_next_hop

The Join Reply is then transmitted according to Section 10.2.5

10.2.5. Join Reply Transmission

A Join Reply is transmitted to all of an ODMRP Router's neighbors, in order to achieve two objectives:

- o Set up or refresh the corresponding Forwarding Tuple for the upstream ODMRP neighbor
- o If the Join Reply was not originated by this router, acknowledge its reception to the previous hop

Before transmitting the Join Reply, the Information Base is updated as follows:

1. If the Pre-acknowledgement Set contains a tuple, such that:

- * O_multicast_group = JR.MulticastGroupAddress
- * O_multicast_source = JR.SourceAddress
- * O_seq_num = JR.SequenceNumber
- * O_originator = JR.NextHopAddress

Then clear the JR.AckRequired flag, and set O_exp_time to EXPIRED

2. Otherwise, if the Pending Acknowledgement Set contains a Pending Tuple such as:

- * P_multicast_group = JR.MulticastGroupAddress
- * P_multicast_source = JR.SourceAddress
- * P_seq_num = JR.SequenceNumber
- * P_next_hop = JR.NextHopAddress

Then set JR.AckRequired, and increase P_nth_time by 1

3. Finally, if neither the Pre-acknowledgement Set nor the Pending Acknowledgement Set contain a corresponding tuple:

1. Insert a Pending Tuple in the Pending Acknowledgement Set, such as:

- + P_multicast_group = JR.MulticastGroupAddress
- + P_multicast_source = JR.SourceAddress
- + P_seq_num = JR.SequenceNumber
- + P_next_hop = JR.NextHopAddress
- + P_nth_time = 1
- + P_acknowledged = FALSE
- + P_expiration_time = current-time + ACK_TIMEOUT

2. Clear the JR.AckRequired flag

10.3. Forwarding Group Maintenance

While an ODMRP Router has data to send to a Multicast Group (on behalf of the Multicast Source), it MUST maintain the Forwarding Group generated by the initial Join Query. To this end, it MUST periodically generate JQ messages, according to Section 10.1.2. The interval between two Join Queries SHOULD be no less than ROUTE_REFRESH_INTERVAL. Note should be taken that, if the Multicast Session has no member other than the source, the Forwarding Group may contain only the designated ODMRP Router for the Multicast Source. That Router still needs to periodically flood Join Queries in order to rebuild a Forwarding Group if necessary.

10.4. Message Transmission

When using physical media subject to collisions and packet loss, both Join Query and Join Reply messages SHOULD be jittered to minimize the effect of collisions, as described in [RFC5148]

11. Unidirectional Links Handling

After sending a Join Reply, an ODMRP Router MUST verify that the upstream neighbor has joined the Forwarding Group. To this end, the following three mechanisms are used after transmitting a given Join Reply:

- o If the ODMRP Router overhears a corresponding Join Reply from the upstream neighbor (see Section 10.2.3), this verifies that the

link is bidirectional and that the upstream neighbor has joined the Forwarding Group (passive acknowledgement)

- o If the ODMRP Router has already overheard a corresponding Join Reply from the upstream neighbor prior to transmitting its own Join Reply, this means that the upstream neighbor has already joined the Forwarding Group (see Section 10.2.3) (pre-acknowledgement)
- o Otherwise, i.e., if neither the pre-acknowledgement nor the passive acknowledgement have verified that the upstream neighbor joined the Forwarding Group (i.e., if the corresponding Pending Tuple expires with P_acknowledged set to False), then the ODMRP Router MUST proceed as follows:
 1. If the corresponding Pending tuple verifies $P_nth_time < JR_RETRIES$, then the ODMRP Router MUST retransmit the Join Reply with the JR.AckRequired flag set
 2. Otherwise, the link between the local interface and the interface of the upstream ODMRP Router identified by JR.NextHopAddress is considered unidirectional. In that case, the ODMRP Router SHOULD proceed as follows:
 - + Find the neighbor interface tuple such that N_address_list contains an address tuple with i_addr = JR.NextHopAddress, and set the variable blacklisted-addresses to the list of addresses contained in N_address_list
 - + Otherwise, if no such tuple exists, set the variable blacklisted-addresses to [JR.NextHopAddress]
 - + Add a tuple in the Blacklist such as:
 - B_neighbor_interface_address_list := blacklisted-addresses
 - B_local_interface := P_local_interface
 - B_exp_time = current-time + BLACKLIST_TIMEOUT

An ODMRP Router MAY attempt to use other mechanisms, such as [I-D.gerla-manet-odmrp-asym], to resume the Forwarding Group building process, instead of or in addition to using the Blacklist

12. SMF considerations

This protocol MAY be run in conjunction with SMF [RFC6621], and benefit from some of its features. In particular, if SMF is in use, it is RECOMMENDED that its duplicate packet detection feature described in Section 6 be used for multicast packet forwarding. Additionally, optimized flooding mechanisms, such as E-CDS or S-MPR, as specified in Appendices A through C of [RFC6621], MAY be used to flood Join Query messages throughout the network.

13. IGMP and MLD considerations

In order to determine whether or not it needs to reply to a Join Query message with a Join Reply message (as specified in Section 10.1.3), an ODMRP Router needs Multicast Group membership information. Such information can be provided by protocols such as IGMP [RFC3376] and/or MLD [RFC3810]. In particular, an ODMRP Router MUST reply with a Join Reply message to a valid Join Query messages advertising a Multicast Session if any of those conditions apply:

- o This Router is subscribed to the corresponding Multicast Group.
- o A host attached to this Router has signaled, for example using IGMP, that it has subscribed to the corresponding Multicast Group.

14. Multicast Packet Forwarding

ODMRP Routers originating and forwarding multicast packets MUST implement a duplicate packet detection (DPD) mechanism. If using IPv4 or IPV6 addresses, the use of SMF [RFC6621] is RECOMMENDED, as described in Section 12.

An ODMRP Router, receiving a non-duplicate multicast data packet, transmits it over all of its interfaces if it is a member of the forwarding group for this data packet, i.e., there exists a tuple in the Forwarding Group Table such as:

F_multicast_group correspond to the multicast address of this packet

F_multicast_source corresponds to the source of this packet

15. Security Considerations

As a multicast routing protocol, this protocol is potentially vulnerable to a number of attacks. This section attempts to describe the envisioned threats to the protocol, as well as some guidelines as to how to ensure confidentiality and integrity of the operations of ODMRP, and to mitigate threats of network overload.

This protocol relies on the use of a Duplicate Packet Detection (DPD) mechanism, such as one described in [RFC6621] (SMF), and suggests the use of optimized flooding to disseminate JQ messages. Some deployments of ODMRP are thus expected to function on top of [RFC6621] by taking advantage of the DPD and optimized flooding mechanisms provided by SMF. Such deployments are thus subject to the same security threats as SMF, such as those described in [I-D.ietf-manet-smf-threats].

15.1. Confidentiality

ODMRP routers which forward packets for multicast data source have to periodically transmit JQ messages throughout the network. In an unsecured network, an attacker could then eavesdrop on those messages and learn part or all of the network topology, depending on the traffic pattern.

15.2. Integrity

ODMRP relies on routers, in particular intermediate routers, to correctly transmit JQ and JR messages. An ODMRP router could, by malice or malfunction, originate JQ messages on behalf of a target multicast source with high enough sequence numbers to replace routing information in other routers. Such behavior would prevent the interested multicast receivers from receiving data packets sent by the target multicast source. An ODMRP router could also forward JQ messages with altered sequence numbers, thus preventing future routing updates. Both behaviors can be mitigated by end-to-end authentication of routing messages.

If NHDP [RFC6130] is not in use to update the Neighbor Interface Set, ODMRP relies on routers correctly informing their neighbors of the addresses they use via the JQ.LastAddress field. Upon transmission of a JQ message, an ODMRP router could, by malice or malfunction, set JQ.LastAddress to a network address that does not belong to this router (address spoofing). This could force neighbor ODMRP routers to blacklist this address in case the malicious router simulate unidirectional links by withholding JR messages. This behavior would break or slow down protocol convergence, potentially triggering data packet loss for multicast receivers. If NHDP is in use, the

deployment is subject to its own security vulnerabilities, such as those described in [RFC7186].

15.3. Channel Overload

ODMRP's main construct, the forwarding group, is built and maintained by having the source ODMRP router periodically flood JQ messages, which can be a costly operation in terms of bandwidth, processing and battery life, if applicable. A malicious router could flood JQ messages at a very high rate to overload the network. It is thus RECOMMENDED that ODMRP routers in a given deployment implement a rate-limit mechanism to prevent such behavior.

The efficiency (in terms of number of multicast data packets transmitted) of forwarding groups depends on routers actually sending JR messages only when necessary, in order to build a graph as sparse as possible and avoid redundant transmissions. Thus an ODMRP router which replies to JQ messages by transmitting one JR message for each of its known neighbors and with JR.NextHopAddress set to an address of this neighbor, would severely harm the efficiency of ODMRP by forcing the routers to build a forwarding group with unnecessary redundancy. Such behavior could also result in routing loops.

16. IANA Considerations

This specification defines two new Message Types, Join Query and Join Reply, which must be allocated from the "Message Type" repository of [RFC5444].

16.1. Join Query Registries

IANA is requested to create a registry for Message-Type-specific Message TLV Types for Join Query messages, with initial assignments according to Table 3.

Type	Description	Allocation Policy
128-223	Unassigned	Expert Review

Table 3: Join Query Message-Type-specific Message TLV Types

IANA is requested to create a registry for Message-Type-specific Address Block TLV Types for Join Query messages, with initial assignments according to Table 4.

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

Table 4: Join Query Message-Type-specific Address Block TLV Types

Allocation of the ADDR-TYPE TLV from the Join Query specific Address Block TLV Types will create a new Type Extension Registry with initial assignments as specified in Table 5.

Name	Type	Type Ext.	Description	Al. Policy
ADDR-TYPE	128	0	MULTICAST-GROUP-ADDRESS	
ADDR-TYPE	128	1	LAST-ADDRESS	
ADDR-TYPE	128	2-255	Unassigned	Expert Review

Table 5: Address Block TLV Type assignment for ADDR-TYPE

16.2. Join Reply Registries

IANA is requested to create a registry for Message-Type-Specific Message TLV Types for Join Reply messages, with initial assignments according to Table 6.

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

Table 6: Join Reply Message-Type-specific Message TLV Types

IANA is requested to create a registry for Message-Type-specific Address Block TLV Types for Join Reply messages, with initial assignments according to Table 7.

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

Table 7: Join Reply Message-Type-specific Address Block TLV Types

Allocation of the ADDR-TYPE TLV from the Join Reply specific Address Block TLV Types will create a new Type Extension Registry with initial assignments as specified in Table 8.

Name	Type	Type Ext.	Description	Al. Policy
ADDR-TYPE	128	0	MULTICAST-GROUP-ADDRESS	
ADDR-TYPE	128	1	NEXT-HOP-ADDRESS	Expert Review
ADDR-TYPE	128	2-255	Unassigned	Expert Review

Table 8: Address Block TLV Type assignment for ADDR-TYPE

17. Acknowledgements

The authors would like to thank Thomas Clausen and Justin Dean for their insightful reviews and comments.

18. References

18.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, BCP 14, March 1997.
- [RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, October 2002.
- [RFC3810] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, June 2004.
- [RFC5148] Clausen, T., Dearlove, C., and B. Adamson, "Jitter

Considerations in Mobile Ad Hoc Networks (MANETs)",
RFC 5148, February 2008.

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

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

18.2. Informative References

[FGMP] Chiang, C., Gerla, M., and L. Zhang, "Forwarding Group
Multicast Protocol (FGMP) for Multihop, Mobile Wireless
Networks", Avril 1998.

[I-D.gerla-manet-odmrp-async]
Gerla, M., Oh, S., and A. Colin de Verdiere, "ODMRP_ASYNC",
draft-gerla-manet-odmrp-async-00 (work in progress).

[I-D.ietf-manet-smf-threats]
Yi, J., Clausen, T., and U. Herberg, "Security Threats for
Simplified Multicast Forwarding (SMF)",
draft-ietf-manet-smf-threats-00 (work in progress),
August 2014.

[ODMRP-Journal]
Lee, S., Su, W., and M. Gerla, "On-Demand Multicast
Routing Protocol in Multihop Wireless Networks",
Journal of Mobile Networks and Applications, Volume 7
Issue 6, Pages 441 - 453, December 2002.

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

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

[RFC7181] Clausen, T., Dearlove, C., Jacquet, P., and U. Herberg,
"The Optimized Link State Routing Protocol Version 2",
RFC 7181, April 2014.

[RFC7186] Yi, J., Herberg, U., and T. Clausen, "Security Threats for
the Neighborhood Discovery Protocol (NHDP)", RFC 7186,
April 2014.

Appendix A. Illustrations

This section shows examples of ODMRP control messages encoded using [RFC5444]. [RFC5444] specifies that a packet is formed by a packet header, an optional TLV block and zero or more messages. This specification does not use or require any packet TLV. Additionally, the minimal packet header required by ODMRP is shown in Figure 1.

```

0
0 1 2 3 4 5 6 7
+-----+
| Ver=0 | PF=0 |
+-----+
```

Figure 1: Packet Header

A.1. Join Query Message

JQ messages are instances of [RFC5444] messages. This section illustrates an example of one such message.

The JQ message's header's flag octet has a value of 9, meaning that the sequence number and source address fields are present, encoding respectively the sequence number and the address of the multicast source that originated the message. Additionally, the address length field (MAL) is set to 3, corresponding to an address length of 4 octets (i.e., the length of an IPv4 address). The overall message size is 23 octets.

An additional Message-Type specific address block is present, with one address and a flag octet (ABF) having value 0, meaning that the address block has no Head or Tail element. The Mid element encodes the Multicast group address. The associated TLV is of type ADDR-TYPE and value 0, i.e. MULTICAST-GROUP-ADDRESS.

The LastAddress element is omitted, meaning that the last JQ message from this interface was transmitted using the same address as this one.

```

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
+-----+-----+-----+-----+-----+-----+-----+-----+
|  Join Query   | 1 0 0 1 | MAL=3 |           Message Length = 23   |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Multicast Source Address         |
+-----+-----+-----+-----+-----+-----+-----+-----+
|           Message Sequence Number           |           TLVs length = 0           |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Num Adrs = 1 |           ABF = 0           |           Multicast Group           ... |
+-----+-----+-----+-----+-----+-----+-----+-----+
| ... Address                                     | Address TLV Block Length = 3 |
+-----+-----+-----+-----+-----+-----+-----+-----+
| ADDR-TYPE   | 1 0 0 0 0 0 0 0 |           0           |
+-----+-----+-----+-----+-----+-----+-----+

```

A.2. Join Reply Message

JR messages are instances of [RFC5444] messages. This section illustrates an example of one such message.

The JR message’s header’s flag octet has a value of 9, meaning that the sequence number and source address fields are present, encoding respectively the sequence number and the address of the multicast source that originated the message. Additionally, the address length field (MAL) is set to 3, corresponding to an address length of 4 octets (i.e., the length of an IPv4 address). The overall message size is 34 octets.

Two additional Message-Type specific address blocks are present, both with one address and a flag octet (ABF) having value 0, meaning that the address block has no Head or Tail element. For the first address block, the Mid element encodes the Multicast group address; the associated Message-Type-specific TLV is of type ADDR-TYPE and value 0, i.e. MULTICAST-GROUP-ADDRESS. The second address block’s Mid element encodes the Next Hop address; its associated Message-Type-specific TLV is of type ADDR-TYPE and value 1, i.e., NEXT-HOP-ADDRESS.

```

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
+-----+-----+-----+-----+-----+-----+-----+-----+
|  Join Reply  | 1 0 0 1 | MAL=3 |           Message Length = 34           |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Multicast Source Address                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
|           Message Sequence Number           |           TLVs length = 0           |
+-----+-----+-----+-----+-----+-----+-----+-----+
| Num Addrs = 1 |           ABF = 0           |           Multicast Group           ... |
+-----+-----+-----+-----+-----+-----+-----+-----+
| ... Address                                     | Address TLV Block Length = 3 |
+-----+-----+-----+-----+-----+-----+-----+-----+
| ADDR-TYPE | 1 0 0 0 0 0 0 0 |           0           | Num Addrs = 1 |
+-----+-----+-----+-----+-----+-----+-----+-----+
|           ABF = 0           |           Next Hop           ... |
+-----+-----+-----+-----+-----+-----+-----+-----+
| ... Address | Address TLV Block Length = 3 | ADDR-TYPE |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 0 0 0 0 0 0 0 |           1           |
+-----+-----+-----+-----+-----+-----+-----+

```

Authors' Addresses

Yunjung Yi
University of California, Los Angeles

Sung-Ju Lee
University of California, Los Angeles

William Su
The Boeing Company

Email: william.su@boeing.com

Mario Gerla
University of California, Los Angeles
3732F Boelter Hall
Computer Science Department
University of California
Los Angeles, CA 90095-1596,
USA

Phone: +1 310 825-4367
Email: gerla@cs.ucla.edu

Axel Colin de Verdiere
University of California, Los Angeles

Email: axel@axelcdv.com

Mobile Ad Hoc Networking (MANET)
Internet-Draft
Intended status: Experimental
Expires: May 20, 2015

M. Gerla
University of California, Los Angeles
S. Oh
UtopiaCompression Corporation
A. Colin de Verdiere
University of California, Los Angeles
November 16, 2014

ODMRP-ASYM
draft-gerla-manet-odmrp-async-01

Abstract

This document describes ODMRP-ASYM, an extension for the On Demand Multicast Routing Protocol [ODMRP], which enables routers to use unidirectional links to route multicast messages.

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 May 20, 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	4
2. Terminology and Notations	4
2.1. Terminology	4
2.2. Notational conventions	5
3. Applicability Statement	6
4. Protocol Overview and Functioning	6
5. Parameters and Constants	7
5.1. Router Parameters	7
5.2. Constants	8
6. Packets and Messages	8
6.1. Join Query	8
6.2. Loop Discovery	8
6.3. Loop Marking	9
7. RFC5444 Encoding	10
7.1. Loop Discovery Encoding	10
7.2. Loop Marking Encoding	11
8. Information Bases	12
8.1. Distance Set	12
8.2. Pending Loops Set	13
9. Protocol Details	13
9.1. Join Query	14
9.1.1. Join Query generation	14
9.1.2. Additional Join Query processing	14
9.1.3. Join Query transmission	14
9.2. Loop Discovery	15
9.2.1. Invalid Loop Discoveries	15
9.2.2. Loop Discovery Generation	15
9.2.3. Loop Discovery Processing	15
9.2.4. Loop Discovery Forwarding	16
9.2.5. Loop Discovery Transmission	17
9.3. Loop Marking	17
9.3.1. Invalid Loop Marking messages	17
9.3.2. Loop Marking Generation	17
9.3.3. Loop Marking Processing	18
9.3.4. Loop Marking Message Forwarding	19
10. Security Considerations	19
11. IANA considerations	20
11.1. Loop Discovery registries	20
11.2. Loop Marking registries	21
12. Acknowledgements	22
13. References	23

13.1. Normative References 23
13.2. Informative References 23
Appendix A. Illustrations 23
A.1. Loop Discovery message 23
A.2. Loop Marking message 24
Authors' Addresses 26

1. Introduction

Due to the nature of the wireless media used, MANETs typically exhibit a non-negligible proportion of asymmetric, or even unidirectional links, even more so when routers themselves make use of disparate transmission powers (such as when using satellite communications). Most routing protocols make sure to avoid these links, typically by detecting and blacklisting them, and use only the fully connected graph formed by the bidirectional links of the network, even though taking advantage of these links can provide significant performance improvement, and even in some cases allow data to flow between weakly connected parts of the network, i.e., parts of the network which are only connected by unidirectional links.

This document specifies ODMRP-ASYM, an extension for the ODMRP protocol [ODMRP] that allows routers to make use of unidirectional links instead of avoiding them. It does so by enabling ODMRP-ASYM routers to discover alternative paths to forward Join Reply messages to the multicast source, building the forwarding group along the way.

2. Terminology and Notations

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 uses the terminology and notation defined in [ODMRP]. Additionally, it uses the terminology defined in Section 2.1 and the notational conventions defined in Section 2.2.

2.1. Terminology

This document defines and uses the following terminology:

ODMRP-ASYM Router - A router that implements this specification, in addition to implementing the original ODMRP specification, as described in [ODMRP]. An ODMRP_ASYM Router MUST use all its ODMRP Interfaces for the operations of this protocol. In other words, an ODMRP_ASYM Router MUST NOT select only a subset of its ODMRP Interfaces over which to process and generate ODMRP_ASYM control messages.

HC2SRC - Abbreviation for Hop-Count to Source; for a given ODMRP-ASYM Router, this refers to the number of hops separating this router from a multicast source, as determined by the Hop Count field of the last valid JQ message received from this source.

Loop - The Loop is the basic construct used by this specification to discover an alternate reverse path to a Multicast Source. A Loop consists in a chain of adjacent (i.e., each Router in the chain can receive messages from the previous one) ODMRP-ASYM Routers. It is closed if the first ODMRP-ASYM Router is the same as the last one in the chain; otherwise, it is open. Furthermore, a Loop has an Originator (the first Router in the chain) and a Destination. Loop Discovery and Loop Marking messages (see Section 6) represent a Loop by an ordered list of one address per ODMRP-ASYM Router belonging to the Loop.

Loop Originator - Refers to the ODMRP-ASYM Router, which has started the Loop Discovery process. It is the first router in the chain, and the last one when the Loop is closed.

Loop Destination - A Loop is built in order to discover an alternate route through which Join Reply messages can be forwarded towards a Multicast Source. This Multicast Source is the Destination of the Loop.

Loop Summit - The objective of building a Loop is to find an ODMRP-ASYM router, strictly closer (in terms of hop count) to the Loop Destination than the Loop Originator. A Loop can contain zero or more of such routers. If the Loop contains at least one such router, the closest to the Loop Destination is the Loop Summit.

Original Join Reply - Refers to the JR message, which failed transmission triggered the Loop Discovery process, as described in Section 9.

2.2. Notational conventions

This document defines and uses the following notational conventions:

tail - Loop Discovery and Loop Marking messages both carry an AddressList field. "tail" is defined such that tail(AddressList) is a list created by removing the first element in AddressList.

head - Conversely, head(AddressList) refers to the first element in AddressList.

length - length(AddressList) is the number of addresses in AddressList.

l[n] - is the nth element of list l. Lists in this specification are 1-based, meaning that they are indexed from 1 to length(list).

3. Applicability Statement

This protocol:

- o Is an extension of the ODMRP [ODMRP] protocol.
- o Enables ODMRP-ASYM routers to make use of unidirectional links for forwarding multicast data packets.
- o Discovers alternative routes on-demand, meaning that it does not impose any extra overhead on ODMRP when there are no unidirectional link present on the path between the Multicast Sources and Receivers.

4. Protocol Overview and Functioning

The objective of this extension is to enable ODMRP Routers to make use of unidirectional links for forwarding multicast packets. In doing so, it may allow data to transit through shorter paths, and in some cases permit packets to be exchanged between weakly connected components of the network, i.e., parts of the network that are only connected by unidirectional links. This mechanism was originally described in [ODMRP_ASYM].

This objective is fulfilled by the following process:

1. Upon detection by an ODMRP-ASYM Router (the Loop Originator) that a Join Reply (towards one Multicast Source) has not been successfully delivered to an upstream router (through the acknowledgement mechanism described in [ODMRP]), the Loop Originator triggers a Loop Discovery procedure as follows:
 1. The Loop Originator generates and broadcasts a Loop Discovery message advertising the Multicast Source as its destination.
 2. The Loop Discovery message is retransmitted by intermediate routers. Each router updates the LD message so as to reflect the list of ODMRP-ASYM routers this message has transited through, as well as the Loop Summit, if there is one among the routers in the Loop.

3. Upon reception of a Loop Discovery message it generated, the Loop Originator verifies that it advertises a valid closed loop, i.e. that at least one router it transited through is closer to the Multicast Source than itself. If that is the case, it starts the Loop Marking procedure.
2. The Loop Marking procedure is as follows:
 1. The Loop Originator generates a Loop Marking message, advertising the list of routers that the LD message went through (the Loop), as well as the Loop Summit. The Loop Marking message is source-routed through the Loop.
 2. Each ODMRP-ASYM router, receiving the Loop Marking message, proceeds as follows:
 - + If that Router's position in the Loop, as recorded by the Address List (see Section 6), is before the Loop Summit, the router forwards the LM message along the Loop.
 - + If the router is the Loop Summit, it restarts the Join Reply procedure by generating a Join Reply message built with information carried within the LM message and forwarding it towards the Multicast Source, then forwards the LM message along the Loop.
 - + Otherwise, i.e., if the router's position in the Loop is after the Loop Summit in the Loop, it adds or updates a Forwarding Tuple to its Forwarding Table (see [ODMRP] section 10), as if it had received the corresponding Join Reply message.

5. Parameters and Constants

In addition to those described in [ODMRP], this specification uses the parameters and constants defined in this section.

5.1. Router Parameters

This specification defines the following router parameters:

`PENDING_LOOP_TIMEOUT` is the minimum time a Loop tuple SHOULD be kept in the Pending Loop set after it was last refreshed.

DEFAULT_LD_HOP_LIMIT is the default value for the LD.HOPLIMIT field used by ODMRP-ASYM Routers generating an LD message.

5.2. Constants

This specification defines the following constants:

NO_SUMMIT - is a value, carried by the LoopSummit field of Loop Discovery and Loop Marking messages, meaning that the corresponding Loop currently has no Loop Summit or that the Loop Summit is not encoded in this message. For example, a Loop Marking message that has already transited through the Loop Summit does not carry its address anymore. NO_SUMMIT has a value of 0.

6. Packets and Messages

This section describes the protocol messages generated and processed by ODMRP-ASYM, according to the terminology defined in Section 2. The objective of this section is to describe the meaning and content of the fields contained in each message. The specifics the encoding of these messages using [RFC5444] is described in Section 7.

6.1. Join Query

In addition to the fields specified in [ODMRP] section 7, Join Query messages generated by ODMRP-ASYM router have an optional hop count field (noted as JQ.HopCount), encoded by way of the <msg-hop-count> field of the JQ message header, as specified in [RFC5444]. This field MUST be included in every JQ message originated by an ODMRP-ASYM router, but MUST NOT be added to transmitted JQ messages if the received JQ message did not contain this field (see Section 9.1.3).

6.2. Loop Discovery

A Loop Discovery (LD) message is generated and processed in order to discover and build a loop. It has the following fields:

LD.AddressLength encodes the length of the addresses carried by this message as follows:

LD.AddressLength := the length of an address in octets - 1

LD.MulticastGroupAddress encodes the address of the Multicast Group, to which this Loop Discovery is addressed.

LD.Destination encodes the address of the Loop Destination.

LD.AddressList is an ordered list of addresses of the routers that this message has traversed, starting with the router that has generated the LD message. This means that head(LD.AddressList) is an address of the Loop Originator.

LD.LoopSummit represents the index in the LD.AddressList field of an address of the corresponding Loop Summit, if it exists. In other words, LD.LoopSummit is either NO_LOOPSUMMIT if the Loop has no Summit, or such that LD.AddressList[LD.LoopSummit] is an address of this Loop's Summit.

LD.MinHC represents the HC2SRC value of the Loop Summit. This field MUST be set to 0 and ignored on reception if LD.LoopSummit = NO_LOOPSUMMIT.

LD.HOPLIMIT represents the maximum number of hops this message can traverse.

LD.HOPCOUNT represents the number of hops this message has already traversed.

6.3. Loop Marking

LM.AddressLength encodes the length of the addresses carried by this message as follows:

LM.AddressLength := the length of an address in octets - 1

LM.MulticastGroupAddress encodes the address of the Multicast Group, to which this Loop Marking message is addressed.

LM.SourceAddress encodes the address of the Multicast Source, towards which the Loop Summit will transmit a Join Reply.

LM.AddressList is an ordered list of addresses of the routers this message has yet to transit through, i.e., head(LM.AddressList) is an address of the next router that should receive this LM. The Loop Marking message is effectively source-routed through these routers.

LM.LoopSummit represents the index in the LD.AddressList field of an address of the corresponding Loop Summit, if it is present in LM.AddressList. In other words, LD.LoopSummit is either NO_LOOPSUMMIT if the LM has already transited through the Loop Summit, or such that LD.AddressList[LD.LoopSummit] is an address of this Loop's Summit.

LM.SourceSequenceNumber corresponds to the sequence number carried by the Original Join Reply message.

7. RFC5444 Encoding

This section describes the encoding of ODMRP_ASYM messages using [RFC5444].

7.1. Loop Discovery Encoding

This protocol defines the Loop Discovery message type. Hence, according to [RFC5444], all Loop Discovery messages are generated, processed and transmitted following this specification. Table 1 shows the mapping between the Loop Discovery elements described in Section 6.2 and their encoding. Each LD message MUST contain exactly one of each of these elements.

LD Element	RFC5444 Element
LD.AddressLength	<msg-addr-length>
LD.MulticastGroupAddress	Address in address block + TLV
LD.Destination	Address in address block + TLV
LD.AddressList	Addresses in address block + TLV
LD.LoopSummit	LOOPSUMMIT Message-type-specific TLV
LD.MinHC	MINHC Message-type-specific TLV
LD.HOPLIMIT	<msg-hop-limit>
LD.HOPCOUNT	<msg-hop-count>

Table 1: Loop Discovery Message Elements

The following considerations apply for when encoding the elements described by Table 1:

LD.AddressLength - Encodes addresses that are between 1 and 16 bytes long.

- LD.MulticastGroupAddress - Is encoded by way of an address block with associated message-type-specific TLV of type ADDR-TYPE and value MULTICAST-GROUP-ADDRESS.
- LD.Destination - Is encoded by way of an address block with associated message-type-specific TLV of type ADDR-TYPE and value DESTINATION-ADDRESS.
- LD.AddressList - Is encoded by way of an address block with <num-addr> = length(LD.AddressList), containing the addresses in order. The address block has an associated message-type-specific TLV of type ADDR-TYPE and value ADDRESS-LIST.
- LD.LoopSummit - Is encoded by way of a message-type-specific TLV of type LOOPSUMMIT, with all the flags cleared except <thasvalue>, <length> = 1 and where <value> is the index of the Loop Summit in LD.AddressList. If LD.LoopSummit = NO_LOOPSUMMIT, <thasvalue> is cleared, and the <length> and <value> fields are omitted.
- LD.HOPLIMIT - Is encoded by way of the <msg-hop-limit> field of the LD message header.
- LD.HOPLIMIT - Is encoded by way of the <msg-hop-count> field of the LD message header.

7.2. Loop Marking Encoding

This protocol defines the Loop Marking message type. Hence, according to [RFC5444], all Loop Marking messages are generated, processed and transmitted following this specification. Table 2 shows the mapping between the Loop Marking elements described in Section 6.3 and their encoding. Each LD message MUST contain exactly one of each of these elements.

LM Element	RFC5444 Element
LM.AddressLength	<msg-addr-length>
LM.MulticastGroupAddress	Address in address block + TLV
LM.SourceAddress	Address in address block + TLV
LM.AddressList	Addresses in address block + TLV
LM.LoopSummit	LOOPSUMMIT Message-type-specific TLV
LM.SourceSequenceNumber	<msg-seq-num>

Table 2: Loop Marking Message Elements

The following considerations apply for when encoding the elements

described by Table 2:

LM.AddressLength - Encodes addresses that are between 1 and 16 bytes long.

LM.MulticastGroupAddress - Is encoded by way of an address block with associated message-type-specific TLV of type ADDR-TYPE and value MULTICAST-GROUP-ADDRESS.

LM.SourceAddress - Is encoded by way of an address block with associated message-type-specific TLV of type ADDR-TYPE and value SOURCE-ADDRESS.

LM.AddressList - Is encoded by way of an address block with <num-addr> = length(LM.AddressList), containing the addresses in order. The address block has an associated message-type-specific TLV of type ADDR-TYPE and value ADDRESS-LIST.

LM.LoopSummit - Is encoded by way of a message-type-specific TLV of type LOOPSUMMIT, with all the flags cleared except <thasvalue>, <length> = 1 and where <value> is the index of the Loop Summit in LM.AddressList. If LM.LoopSummit = NO_LOOPSUMMIT, <thasvalue> is cleared, and the <length> and <value> fields are omitted.

8. Information Bases

In additions to the information bases described in [ODMRP], each ODMRP-ASYM Router maintains a Distance Set and a Pending Loop Set, as described in the following sections. These information sets are given so as to facilitate description of message generation, forwarding and processing rules. An implementation may chose any representation or structure to maintain this information.

8.1. Distance Set

The Distance set contains distance tuples, recording the distance in hop count to (active) Multicast sources, as recorded by received Join Query messages, and containing the following fields:

(D_source, D_hop_count, D_seq_num, D_exp_time)

Where:

D_source - is the address of the Multicast Source, carried by the corresponding Join Query message.

D_hop_count - is the distance, in hops, to the Multicast Source, as recorded by the most recent Join Query received that was originated by this source.

D_seq_num - is the sequence number of the Join Query message that updated this tuple.

D_exp_time - is the time at which the tuple MUST be considered expired and thus MUST NOT be taken into consideration by the operations of this protocol extension.

8.2. Pending Loops Set

The Pending Loops Set contains pending loop tuples, each recording information about an open Loop that this Router is part of, either because it has initiated the corresponding Loop Discovery process (i.e., this router is the Loop Originator) or because it has received and forwarded a corresponding Loop Discovery message. It contains the following fields:

(L_originator, L_destination, L_exp_time)

Where:

L_originator - is an address of the Loop Originator, as recorded by the corresponding Loop Discovery message

L_destination - is an address of the Loop Destination, as recorded by the corresponding Loop Discovery message

L_exp_time - is the time at which the tuple MUST be considered expired and thus MUST NOT be taken into consideration by the operations of this protocol extension.

9. Protocol Details

This protocol generates, processes and forwards Loop Discovery and Loop Marking messages, according to the following sections. This section makes use of the terminology defined in Section 10 of [ODMRP], as well as the following additional notation:

OJR Refers to the corresponding Original Join Reply, as defined in Section 2.

9.1. Join Query

ODMRP-ASYM requires that JQ messages carry a HopCount field, in order to maintain the Distance Set. This section specifies the additional processing required.

9.1.1. Join Query generation

In addition to the fields described in [ODMRP] section 10.1.2, Join Query messages, originated by ODMRP-ASYM Routers, MUST be generated with the JQ.HopCount field present and set to 0.

9.1.2. Additional Join Query processing

Upon receiving a valid Join Query message, an ODMRP-ASYM router MUST proceed as follows, after executing the steps specified in [ODMRP], section 10.1.3:

1. Find the Distance tuple in the Distance Set, such as:

- * D_source = JQ.SourceAddress

And update it in the following way:

- * D_hop_count := JQ.HopCount

- * D_seq_num := JQ.SequenceNumber

- * D_exp_time := current-time + ROUTE_TIMEOUT

2. If no such tuple exists, create one with the following fields:

- * D_source = JQ.SourceAddress

- * D_hop_count := JQ.HopCount

- * D_seq_num := JQ.SequenceNumber

- * D_exp_time := current-time + ROUTE_TIMEOUT

9.1.3. Join Query transmission

A JQ message, considered for forwarding, MUST be updated as follows, after following the process specified in [ODMRP], section 10.1.4:

- o If the HopCount field is present, update it so that JQ.HopCount := JQ.HopCount + 1.

9.2. Loop Discovery

9.2.1. Invalid Loop Discoveries

A Loop Discovery Message, received by an ODMRP-ASYM Router, is invalid and MUST be discarded without further processing, and in particular MUST NOT be considered for forwarding, if:

- o The address length carried by the received Loop Discovery Message (see Section 6) differs from the length of the addresses of this Router.
- o LD.HOPCOUNT > LD.HOPLIMIT, or LD.HOPCOUNT = LD.HOPLIMIT and LD.Originator is not an address of this Router.
- o LD.Originator is an address of this Router, and there isn't any Pending Loop tuple in the Pending Loops set, such as:
 - * L_originator is an address of this Router.
 - * L_destination = LD.Destination.

9.2.2. Loop Discovery Generation

A Loop Discovery message SHOULD be generated upon detection that a Join Reply (the Unsuccessful Join Reply) was unable to be delivered to the upstream router. A Loop Discovery message is generated according to Section 6, with the following fields:

- o LD.AddressLength := UJR.AddressLength.
- o LD.MulticastGroupAddress := UJR.MulticastGroupAddress.
- o LD.AddressList set to a list, containing as its only element an address of this Router.
- o LD.LoopSummit := NO_SUMMIT.
- o LD.MinHC := D_hop_count.

9.2.3. Loop Discovery Processing

Upon receiving a valid Loop Discovery message, an ODMRP-ASYM Router proceeds as follows:

1. If head(LD.AddressList) is an address of this Router (i.e., the Loop Discovery message advertises a closed Loop originated by this router), then:

1. If there exists a Distance tuple (henceforth "corresponding Distance tuple") in the Distance set, such as:

- + D_source = LD.Destination
- + D_hop_count < LD.MinHC

Then this Loop Discovery message advertises a valid closed Loop. Otherwise, the advertised loop is invalid. The Loop Discovery message MUST be discarded and MUST NOT be processed further.

2. Generate a new Loop Marking message according to Section 9.3.2
 3. Set the corresponding Pending Loop tuple as expired, by setting P_exp_time to current_time - 1. The Loop Discovery message is not processed further, and in particular MUST NOT be considered for forwarding.
2. Else, find the Pending Loop tuple (denoted "matching Pending Loop tuple") such that:

- * L_originator = head(LD.AddressList).
- * L_destination = LD.Destination.

If it exists, update it such that L_exp_time := PENDING_LOOP_TIMEOUT. Else, create one with the required fields set as above, and L_exp_time := PENDING_LOOP_TIMEOUT.

3. The Loop Discovery message is then considered for forwarding, according to Section 9.2.4.

9.2.4. Loop Discovery Forwarding

A Loop Discovery message, considered for forwarding, MUST be updated as follows, prior to being transmitted:

1. Append an address of this Router to LD.AddressList. The appended address MUST be an address of an ODMRP interface of this Router, but MAY be chosen freely among these addresses if there are more than one.
2. Find the Distance Tuple defined by:
 - * D_source = LD.Destination.

3. If such a tuple exists, and if `D_hop_count < LD.MinHC`, then update the Loop Discovery message as follows:
 1. `LD.MinHC := D_hop_count`.
 2. `LD.LoopSummit := length(LD.AddressList)`.
4. `LD.HOPCOUNT := LD.HOPCOUNT + 1`.

9.2.5. Loop Discovery Transmission

A Loop Discovery message **MUST** be broadcasted on all participating ODMRP interfaces.

9.3. Loop Marking

9.3.1. Invalid Loop Marking messages

A Loop Marking message (LM), received by an ODMRP-ASYM Router, is invalid and **MUST** be discarded without further processing, and in particular **MUST NOT** be considered for forwarding, if:

- o The address length carried by the Loop Marking Message (see Section 6) differs from the length of the addresses of the ODMRP-ASYM Router.
- o `head(LM.AddressList)` is not an address of this Router.

9.3.2. Loop Marking Generation

A Loop Marking Message **MUST** be generated by an ODMRP-ASYM Router upon receiving a Loop Discovery (referenced as LD in the following) message advertising a valid closed Loop originated by this router, as described in Section 9.2.3. A Loop Marking message **MUST** be generated according to Section 6, with the following contents, computed from the corresponding Distance tuple and received LD message:

- o `LM.AddressLength := LD.AddressLength`
- o `LM.MulticastGroupAddress := LD.MulticastGroupAddress`
- o `LM.SourceAddress := LD.Destination`
- o `LM.AddressList := LD.AddressList`
- o `LM.LoopSummit := LD.LoopSummit`

- o LM.SequenceNumber := D_seq_num (from the corresponding Distance tuple)

9.3.3. Loop Marking Processing

Upon receiving a valid Loop Marking message, an ODMRP-ASYM Router proceeds as follows:

1. If LM.LoopSummit = 1, this Router is the Loop Summit for this Loop, and MUST do the following:
 1. Find the Routing Tuple (corresponding Routing Tuple) in the Routing Set such as:
 - + R_source = LM.SourceAddress.
 2. If no such tuple exists, the Loop Marking message is invalid and MUST be discarded without any further processing.
 3. Create a new Join Reply with the following fields:
 - + JR.AddressLength := LM.AddressLength.
 - + JR.MulticastGroupAddress := LM.MulticastGroupAddress.
 - + JR.AckRequired := 0.
 - + JR.SourceAddress := LM.SourceAddress.
 - + JR.SequenceNumber := R_seq_num.
 - + JR.NextHopAddress := R_next_hop.

The new Join Reply is then considered for forwarding, according to [ODMRP].

2. If LM.LoopSummit = 1 or LM.LoopSummit = NO_SUMMIT, then this Router is either the Loop Summit, or after the Loop Summit in the corresponding Loop. Hence, it MUST join the corresponding forwarding group by adding or updating an entry in its forwarding table, by proceeding as follows:
 1. Find the Forwarding Tuple (matching Forwarding Tuple) in the Forwarding Table such as:
 - + F_multicast_group = LM.MulticastGroupAddress.

- + F_source = LM.SourceAddress.
- 2. If no matching Forwarding Tuple is found, create a Forwarding Tuple with:
 - + F_multicast_group := LM.MulticastGroupAddress.
 - + F_source := LM.SourceAddress.
 - + F_seq_num := -1.
 - + F_exp_time := FG_TIMEOUT.
- 3. The matching Forwarding Tuple, existing or new, is compared with the matching Routing Tuple: if F_seq_num <= LM.SourceSequenceNumber, then update the tuple as follows:
 - 1. F_seq_num := LM.SourceSequenceNumber.
 - 2. Set F_exp_time := current-time + FG_TIMEOUT.
- 3. The Loop Marking message is then considered for forwarding, according to Section 9.3.4.

9.3.4. Loop Marking Message Forwarding

A Loop Marking message, considered for forwarding, MUST be updated as follows:

- o Remove the first address from LM.AddressList, i.e., set LM.AddressList to tail(LM.AddressList). If LM.AddressList becomes empty, the Loop Marking message MUST be discarded and MUST NOT be processed further.
- o If LM.LoopSummit != NO_SUMMIT, then update LM.LoopSummit such that LM.LoopSummit := LM.LoopSummit - 1, or LM.LoopSummit := NO_LOOPSUMMIT if LM.LoopSummit = 1.

The Loop Marking message is then transmitted in unicast to the ODMRP-ASYM Router, identified by head(LM.AddressList).

10. Security Considerations

As an extension of ODMRP, this protocol inherits from all the security considerations described in [ODMRP]. This document does not currently describe additional security concerns or specify any other security measure.

11. IANA considerations

The IANA is requested to assign two new message types for Loop Discovery and Loop Marking messages, as well as one Message-Type-Specific TLV Type and one Message-Type-Specific Address block TLV registry for each of those message types, as specified below.

11.1. Loop Discovery registries

IANA is requested to create a registry for Message-Type-specific Message TLVs for LD messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 3.

Type	Description	Allocation policy
128	LOOPSUMMIT	
129	MINHC	
130-223	Unassigned	Expert review

Table 3: Loop Discovery Message-Type-Specific TLV types

Allocation of the LOOPSUMMIT and MINHC TLVs from the LD Message-Type-specific Message TLV types in Table 3 will create two new Type Extension registries, with assignments specified in Table 4 and Table 5.

Type Extension	Description	Allocation policy
0-255	Unassigned	Expert review

Table 4: LD Message TLV Type assignment: LOOPSUMMIT

Type Extension	Description	Allocation policy
0-255	Unassigned	Expert review

Table 5: LD Message TLV Type assignment: MINHC

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

specified in Table 6.

Type	Description	Allocation policy
128	ADDR-TYPE	
129-223	Unassigned	Expert review

Table 6: Loop Discovery Message-Type-specific address block TLV types

Allocation of the ADDR-TYPE TLV from the LD Message-Type-specific TLV Address block TLV Types will create a new Type extension registry, with assignments specified in Table 7.

Type Extension	Description	Allocation policy
0	MULTICAST-GROUP-ADDRESS	
1	DESTINATION-ADDRESS	
2	ADDRESS-LIST	
3-255	Unassigned	Expert review

Table 7: LD Message Address block TLV Type assignments: ADDR-TYPE

11.2. Loop Marking registries

IANA is requested to create a registry for Message-Type-specific Message TLVs for LM messages, in accordance with Section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 8.

Type	Description	Allocation policy
128	LOOPSUMMIT	
129-223	Unassigned	Expert review

Table 8: Loop Discovery Message-Type-Specific TLV types

Allocation of the LOOPSUMMIT TLV from the LM Message-Type-specific TLV Types will create a new Type extension registr, with assignments specified in Table 9.

Type extension	Description	Allocation policy
0-255	Unassigned	Expert review

Table 9: LM Message Types assignments: LOOPSUMMIT

IANA is requested to create a registry for Message-Type-specific address block TLVs for LM messages, in accordance with section 6.2.1 of [RFC5444], and with initial assignments and allocation policies as specified in Table 10.

Type	Description	Allocation policy
128	ADDR-TYPE	
129-223	Unassigned	Expert review

Table 10: Loop Marking Message-Type-specific address block TLV types

Allocation of the ADDR-TYPE TLV from the LM Message-Type-specific TLV Address block TLV Types will create a new Type extension registry, with assignments specified in Table 11.

Type Extension	Description	Allocation policy
0	MULTICAST-GROUP-ADDRESS	
1	MULTICAST-SOURCE-ADDRESS	
2	ADDRESS-LIST	
3-255	Unassigned	Expert review

Table 11: LD Message Address block TLV Type assignments: ADDR-TYPE

12. Acknowledgements

The authors would like to thank Yeng-Zhong Lee, Joon-Sang Park, and Yunjung Yi for their work on the original protocol, as published in [ODMRP_ASYM].

13. References

13.1. Normative References

- [ODMRP] Yi, Y., Lee, S., Su, W., Gerla, M., and A. Colin de Verdiere, "The On Demand Multicast Routing Protocol", draft-gerla-manet-odmrp (work in progress).
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, BCP 14, March 1997.
- [RFC5444] Clausen, T., Dearlove, C., Dean, J., and C. Adjih, "Generalized MANET Packet/Message Format", RFC 5444, February 2009.

13.2. Informative References

- [ODMRP_ASYM] Gerla, M., Lee, Y., Park, J., and Y. Yi, "On Demand Multicast Routing With Unidirectional Links".

Appendix A. Illustrations

This section shows examples of ODMRP-ASYM control messages encoded using [RFC5444]. [RFC5444] specifies that a packet is formed by a packet header, an optional TLV block and zero or more messages. ODMRP-ASYM does not use any packet TLV, and the minimal packet header required by ODMRP-ASYM does not differ from the one required by ODMRP (see [ODMRP], Appendix B).

A.1. Loop Discovery message

LD messages are instances of [RFC5444] messages. This section illustrates an example of LD message.

The LD message's header has the bits 1 (mhashoplimit) and 2 (mhashopcount) set, indicating that the hop count and hop limit fields are present, but not the originator address and sequence number. Its address length field is set to 3, indicating that the addresses carried by this message are $3 + 1 = 4$ octets long. The overall message length is 56 octets.

Both the LD.LOOPSUMMIT and LD.MINHC field are required, and encoded by the two TLVs with corresponding types that the message carries, with respective values LS and MHC.

The message has 3 address blocks. The first two encode LD.MulticastGroupAddress and LD.Destination respectively, and have a flag octet (ABF) value of 0, hence with no Tail or Head section, and

a Mid section of length 4 octets. The third address block encodes LD.AddressList, and contains 4 addresses, sharing a 3 octets prefix (Head), as specified by the Field octet value of 128 (bit 0 set). These address blocks have each one associated Message-Type-specific Address block TLV of type ADDR-TYPE and type extension 0 (MULTICAST-GROUP-ADDRESS), 1 (DESTINATION-ADDRESS) and 2 (ADDRESS-LIST) respectively.

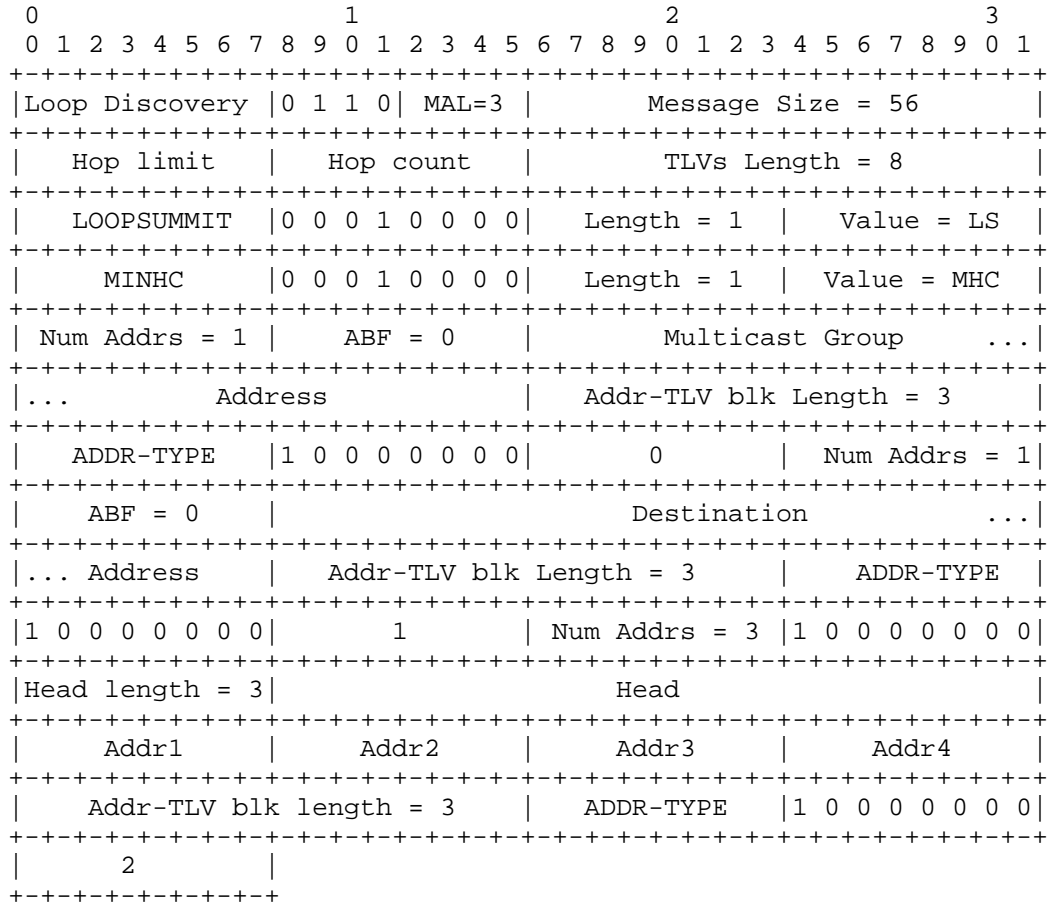


Figure 1

A.2. Loop Marking message

LM messages are instances of [RFC5444] messages. This section illustrates an example of LM message.

The LM message's header has only the bit 3 (mhasseqnum) set,

indicating that the sequence number field is present, but that the originator address, hop count and hop limit fields are omitted. Its address length field is set to 3, indicating that the addresses carried by this message are $3 + 1 = 4$ octets long. The overall message length is 52 octets.

The message contains one Message-Type-specific TLV, of Type LOOPSUMMIT and value LS = LM.LoopSummit.

The message has 3 address blocks. The first two encode LM.MulticastGroupAddress and LM.Destination respectively, and have a flag octet (ABF) value of 0, hence with no Tail or Head section, and a Mid section of length 4 octets. The third address block encodes LM.AddressList, and contains 4 addresses, sharing a 3 octets prefix (Head), as specified by the Field octet value of 128 (bit 0 set). These address blocks have each one associated Message-Type-specific Address block TLV of type ADDR-TYPE and type extension 0 (MULTICAST-GROUP-ADDRESS), 1 (MULTICAST-SOURCE-ADDRESS) and 2 (ADDRESS-LIST) respectively.

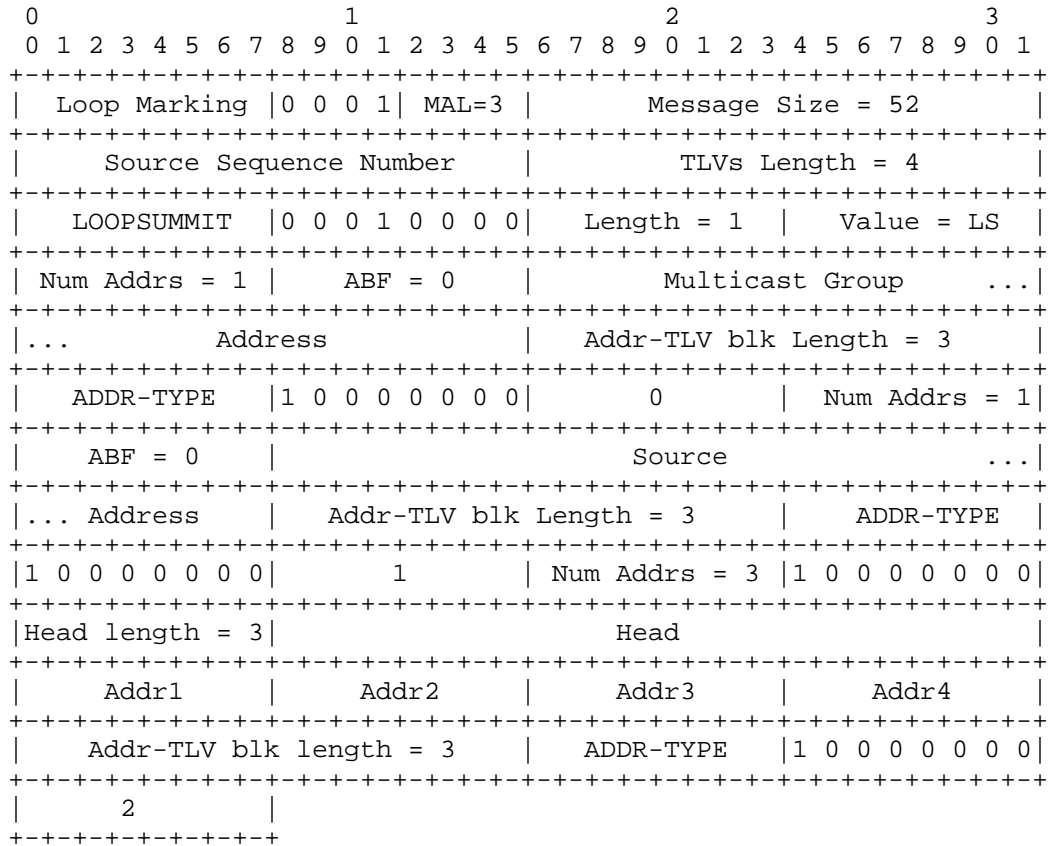


Figure 2

Authors' Addresses

Mario Gerla
 University of California, Los Angeles
 3732F Boelter Hall
 Computer Science Department
 University of California
 Los Angeles, CA 90095-1596,
 USA

Phone: +1 310 825-4367
 Email: gerla@cs.ucla.edu

Soon Young Oh
UtopiaCompression Corporation

Email: soon@utopiacompression.com

Axel Colin de Verdiere
University of California, Los Angeles

Email: axel@axelcdv.com

Mobile Ad hoc Networks Working Group
Internet-Draft
Intended status: Experimental
Expires: November 4, 2016

C. Perkins
Futurewei
S. Ratliff
Idirect
J. Dowdell
Airbus Defence and Space
L. Steenbrink
HAW Hamburg, Dept. Informatik
V. Mercieca
Airbus Defence and Space
May 3, 2016

Ad Hoc On-demand Distance Vector Version 2 (AODVv2) Routing
draft-ietf-manet-aodvv2-16

Abstract

The Ad Hoc On-demand Distance Vector Version 2 (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.

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 4, 2016.

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. Overview	4
2. Terminology	5
3. Applicability Statement	9
4. Purpose of the Experiment	11
5. Data Structures	12
5.1. InterfaceSet	12
5.2. Router Client Set	12
5.3. Neighbor Set	13
5.4. Sequence Numbers	14
5.5. Local Route Set	15
5.6. Multicast Route Message Set	17
5.7. Route Error (RERR) Set	19
6. Metrics	19
7. AODVv2 Protocol Operations	21
7.1. Initialization	21
7.2. Next Hop Monitoring	21
7.3. Neighbor Set Update	23
7.4. Interaction with the Forwarding Plane	24
7.5. Message Transmission	26
7.6. Route Discovery, Retries and Buffering	27
7.7. Processing Received Route Information	28
7.7.1. Evaluating Route Information	29
7.7.2. Applying Route Updates	30
7.8. Suppressing Redundant Messages Using the Multicast Route Message Set	33
7.9. Suppressing Redundant Route Error Messages using the Route Error Set	35
7.10. Local Route Set Maintenance	35
7.10.1. LocalRoute State Changes	35
7.10.2. Reporting Invalid Routes	38
8. AODVv2 Protocol Messages	38
8.1. Route Request (RREQ) Message	38
8.1.1. RREQ Generation	40
8.1.2. RREQ Reception	41
8.1.3. RREQ Forwarding	42
8.2. Route Reply (RREP) Message	42
8.2.1. RREP Generation	43
8.2.2. RREP Reception	45
8.2.3. RREP Forwarding	46

8.3.	Route Reply Acknowledgement (RREP_Ack) Message	47
8.3.1.	RREP_Ack Request Generation	47
8.3.2.	RREP_Ack Reception	48
8.3.3.	RREP_Ack Response Generation	49
8.4.	Route Error (RERR) Message	49
8.4.1.	RERR Generation	50
8.4.2.	RERR Reception	51
8.4.3.	RERR Regeneration	53
9.	RFC 5444 Representation	53
9.1.	Route Request Message Representation	54
9.1.1.	Message Header	55
9.1.2.	Message TLV Block	55
9.1.3.	Address Block	55
9.1.4.	Address Block TLV Block	55
9.2.	Route Reply Message Representation	56
9.2.1.	Message Header	56
9.2.2.	Message TLV Block	56
9.2.3.	Address Block	57
9.2.4.	Address Block TLV Block	57
9.3.	Route Reply Acknowledgement Message Representation	58
9.3.1.	Message Header	58
9.3.2.	Message TLV Block	58
9.3.3.	Address Block	58
9.3.4.	Address Block TLV Block	58
9.4.	Route Error Message Representation	58
9.4.1.	Message Header	58
9.4.2.	Message TLV Block	59
9.4.3.	Address Block	59
9.4.4.	Address Block TLV Block	59
10.	Simple External Network Attachment	60
11.	Configuration	62
11.1.	Timers	62
11.2.	Protocol Constants	64
11.3.	Local Settings	65
11.4.	Network-Wide Settings	65
11.5.	MetricType Allocation	66
11.6.	RFC 5444 Message Type Allocation	66
11.7.	RFC 5444 Message TLV Types	66
11.8.	RFC 5444 Address Block TLV Type Allocation	67
11.9.	ADDRESS_TYPE TLV Values	67
12.	IANA Considerations	68
13.	Security Considerations	68
13.1.	Availability	68
13.1.1.	Denial of Service	68
13.1.2.	Malicious RERR messages	69
13.1.3.	False Confirmation of Link Bidirectionality	70
13.1.4.	Message Deletion	71
13.2.	Confidentiality	71

13.3. Integrity	71
13.3.1. Message Insertion	71
13.3.2. Message Modification - Man in the Middle	72
13.3.3. Replay Attacks	73
13.4. Protection Mechanisms	73
13.4.1. Confidentiality and Authentication	73
13.4.2. Integrity and Trust using ICVs	73
13.4.3. Replay Protection using Timestamps	73
13.4.4. Application to AODVv2	74
13.5. Key Management	79
14. Acknowledgments	81
15. References	81
15.1. Normative References	81
15.2. Informative References	82
Appendix A. AODVv2 Draft Updates	83
Authors' Addresses	83

1. Overview

The Ad hoc On-Demand Distance Vector Version 2 (AODVv2) protocol enables dynamic, multihop routing between participating mobile nodes wishing to establish and maintain an ad hoc network. The basic operations of the AODVv2 protocol are route discovery and route maintenance. AODVv2 does not require nodes to maintain routes to destinations that are not in active communication. AODVv2 allows mobile nodes to respond to link breakages and changes in network topology in a timely manner. The operation of AODVv2 is loop-free, and by avoiding the Bellman-Ford "counting to infinity" problem offers quick convergence when the ad hoc network topology changes (typically, when a node moves in the network). When links break, AODVv2 causes the affected set of nodes to be notified so that they are able to invalidate the routes using the lost link.

One distinguishing feature of AODVv2 is its use of a destination sequence number for each route entry. The destination sequence number is created by the destination to be included along with any route information it sends to requesting nodes. Using destination sequence numbers ensures loop freedom and is simple to program. Given the choice between two routes to a destination, a requesting node is required to select the one with the greatest sequence number.

Compared to AODV [RFC3561], AODVv2 has moved some features out of the scope of the document, notably intermediate route replies, expanding ring search, route lifetimes and precursor lists. However, the document has been designed to allow their specification in a separate document. Hello messages and local repair have been removed. AODVv2 provides a mechanism for the use of multiple metric types. Message formats have been updated and made compliant with [RFC5444]. AODVv2

control messages are defined as sets of data, which are mapped to message elements using the Generalized MANET Packet/Message Format defined in [RFC5444] and sent using the parameters in [RFC5498]. Verification of link bidirectionality has been substantially improved, and additional refinements made for route timeouts and state management.

The basic protocol mechanisms are as follows. Since AODVv2 is a reactive protocol, route discovery is initiated only when a route to the target is needed (i.e. when a router' client wants to send data). AODVv2 does this with the help of a Route Request (RREQ) and Route Reply (RREP) cycle: an RREQ is distributed across the network until it arrives at the target. When forwarding an RREQ, all routers across the network store the neighbor they've received the RREQ from, memorizing a possible route back to the originator of the RREQ. When the target receives the RREQ, it answers with an RREP, which then travels back to the originator along the path memorized by the intermediate routers. A metric value is included within the messages to record the cost of the route. AODVv2 uses sequence numbers to identify stale routing information, and compares route metric values to determine if advertised routes could form loops.

Route maintenance includes confirming bidirectionality of links to next hop AODVv2 routers and issuing Route Error (RERR) messages informing other routers of broken links. It also includes reacting to received Route Error messages, and extending and enforcing route timeouts.

The on-demand nature of AODVv2 requires signals to be exchanged between AODVv2 and the forwarding plane. These signals indicate when: * a packet is to be forwarded, in order to initiate route discovery * packet forwarding fails, in order to initiate route error reporting * a packet is successfully forwarded, for route maintenance.

Security for authentication of AODVv2 routers and encryption of control messages is accomplished using the TIMESTAMP and ICV TLVs defined in [RFC7182].

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]. In addition, this document uses terminology from [RFC5444], and defines the following terms:

AddressList

A list of IP addresses as used in AODVv2 messages.

AckReq

Used in a Route Reply Acknowledgement message to indicate that a Route Reply Acknowledgement is expected in return.

AdvRte

A route advertised in an incoming route message.

AODVv2 Router

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

CurrentTime

The current time as maintained by the AODVv2 router.

ENAR (External Network Access Router)

An AODVv2 router with an interface to an external, non-AODVv2 network.

InterfaceSet

The set of all network interfaces supporting AODVv2.

Invalid route

A route that cannot be used for forwarding but still contains useful sequence number information.

LocalRoute

An entry in the Local Route Set as defined in Section 5.5.

MANET

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

MetricType

The metric type for a metric value included in a message.

MetricTypeList

A list of metric types associated with the addresses in the AddressList of a Route Error message.

Neighbor

An AODVv2 router from which an RREQ or RREP message has been received. Neighbors exchange routing information and verify bidirectionality of the link to a neighbor before installing a route via that neighbor into the Local Route Set.

OrigAddr

The source IP address of the IP packet triggering route discovery.

OrigMetric

The metric value associated with the route to OrigPrefix.

OrigPrefix

The prefix configured in the Router Client entry which includes OrigAddr.

OrigPrefixLen

The prefix length, in bits, configured in the Router Client entry which includes OrigAddr.

OrigSeqNum

The sequence number of the AODVv2 router which originated the Route Request on behalf of OrigAddr.

PktSource

The source address of the IP packet that triggered a Route Error message.

PrefixLengthList

A list of routing prefix lengths associated with the addresses in the AddressList of a message.

Reactive

Performed only in reaction to specific events. In AODVv2, routes are requested only when data packets need to be forwarded. In this document, "reactive" is synonymous with "on-demand".

RERR (Route Error)

The AODVv2 message type used to indicate that an AODVv2 router does not have a valid LocalRoute toward one or more particular destinations.

RERR_Gen (RERR Generating Router)

The AODVv2 router generating a Route Error message.

RerrMsg (RERR Message)

A Route Error (RERR) message.

Routable Unicast IP Address

A routable unicast IP address is a unicast IP address that is scoped sufficiently to be forwarded by a router. Globally-scoped unicast IP addresses and Unique Local Addresses (ULAs) [RFC4193] are examples of routable unicast IP addresses.

Router Client

An address or address range configured on an AODVv2 router, on behalf of which that router will initiate and respond to route

discoveries. These addresses may be used by the AODVv2 router itself or by its Router Clients that are reachable without traversing another AODVv2 router.

RREP (Route Reply)

The AODVv2 message type used to reply to a Route Request message.

RREP_Gen (RREP Generating Router)

The AODVv2 router that generates the Route Reply message, i.e., the router configured with TargAddr as a Router Client.

RREQ (Route Request)

The AODVv2 message type used to discover a route to TargAddr and distribute information about a route to OrigPrefix.

RREQ_Gen (RREQ Generating Router)

The AODVv2 router that generates the Route Request message, i.e., the router configured with OrigAddr as a Router Client.

RteMsg (Route Message)

A Route Request (RREQ) or Route Reply (RREP) message.

SeqNum

The sequence number maintained by an AODVv2 router to indicate freshness of route information.

SeqNumList

A list of sequence numbers associated with the addresses in the AddressList of a message.

TargAddr

The target address of a route request, i.e., the destination address of the IP packet triggering route discovery.

TargMetric

The metric value associated with the route to TargPrefix.

TargPrefix

The prefix configured in the Router Client entry which includes TargAddr.

TargPrefixLen

The prefix length, in bits, configured in the Router Client entry which includes TargAddr.

TargSeqNum

The sequence number of the AODVv2 router which originated the Route Reply on behalf of TargAddr.

Unreachable Address

An address reported in a Route Error message, as described in Section 8.4.1.

Upstream

In the direction from destination to source (from TargAddr to OrigAddr).

Valid route

A route that can be used for forwarding, as described in Section 8.4.1.

This document uses the notational conventions in Table 1 to simplify the text.

Notation	Meaning
Route[Address]	A route toward Address
Route[Address].Field	A field in a route toward Address
RteMsg.Field	A field in either RREQ or RREP

Table 1: Notational Conventions

3. Applicability Statement

The AODVv2 routing protocol is a reactive routing protocol intended for use in mobile ad hoc wireless networks. A reactive protocol only sends messages to discover a route when there is data to send on that route. Therefore, a reactive routing protocol requires certain interactions with the forwarding plane (for example, to indicate when a packet is to be forwarded, in order to initiate route discovery). The set of signals exchanged between AODVv2 and the forwarding plane are discussed in Section 7.4.

AODVv2 is designed for stub or disconnected mobile ad hoc networks, i.e., non-transit networks or those not connected to the internet. AODVv2 can, however, be configured to perform gateway functions when attached to external networks, as discussed in Section 10.

AODVv2 handles a wide variety of mobility and traffic patterns by determining routes on-demand. In networks with a large number of routers, AODVv2 is best suited for relatively sparse traffic scenarios where each router forwards IP packets to a small percentage of other AODVv2 routers in the network. In this case fewer routes are needed, and therefore less control traffic is produced. In large networks with very frequent or bursty traffic, AODVv2 control

messages may cause a broadcast storm, overwhelming the network with control messages and preventing routes from being established. This especially applies to networks with point-to-point or point-to-multipoint traffic. In this case, the transmission priorities described in Section 7.5 prioritize route maintenance traffic over route discovery traffic.

Data packets may be buffered until a route to their destination is available, as described in Section 7.6.

AODVv2 provides for message integrity and security against replay attacks by using integrity check values, timestamps and sequence numbers, as described in Section 13. If security associations can be established, encryption can be used for AODVv2 messages to ensure that only trusted routers participate in routing operations.

Since the route discovery process aims for a route to be established in both directions along the same path, uni-directional links are not suitable. AODVv2 will detect and exclude those links from route discovery. The route discovered is optimised for the requesting router, and the return path may not be the optimal route.

AODVv2 is applicable to memory constrained devices, since only a little routing state is maintained in each AODVv2 router. AODVv2 routes that are not needed for forwarding data do not need to be maintained. On routers unable to store persistent AODVv2 state, recovery can impose a performance penalty (e.g., in case of AODVv2 router reboot), since if a router loses its sequence number, there is a delay before the router can resume full operations. This is described in Section 7.1.

AODVv2 supports routers with multiple interfaces and multiple IP addresses per interface. A router may also use the same IP address on multiple interfaces. AODVv2 requires only that each interface configured for AODVv2 has at least one unicast IP address. Address assignment procedures are out of scope for AODVv2.

AODVv2 supports Router Clients with multiple interfaces, as long as each interface is configured with its own unicast IP address. Multi-homing of a Router Client IP address is not supported by AODVv2, and therefore an IP address SHOULD NOT be configured as a Router Client on more than one AODVv2 router at any one time.

The routing algorithm in AODVv2 MAY be operated at layers other than the network layer, using layer-appropriate addresses.

4. Purpose of the Experiment

AODVv2 is an Experimental protocol. While it is based on AODV [RFC3561], important protocol mechanisms have changed: *

- * Bidirectionality is ensured using a new mechanism
- * Alternate metrics may be used to determine route quality
- * Support for multiple interfaces has been improved
- * Support for multi-interface IP addresses has been added
- * A new security model allowing end to end integrity checks has been added
- * A new message format ([RFC5444]) is used.

Many of these changes have been made quite recently, after a protocol development hiatus of several years.

Thus, the purpose of the experiment is to gain information on the behavior of these significant changes in real-world deployments, not only to learn about AODVv2 in particular, but also to further the knowledge base of reactive protocols in general.

Suitable future experiments could be:

- o Evaluation of the new features mentioned above with regard to performance and functionality
- o determining default values for configuration parameters such as timeouts, numbers of retries, buffer sizes, control message limits (ensuring the level of multicast traffic does not interfere with data traffic throughput)
- o specification of optimisations / verification of minimum requirements for low-power or low-memory routers
- o developing security strategies for different environments
- o Quantification of effectiveness and performance of precursors
- o Evaluation of different metric types and their suitability for reactive distance vector protocols
- o Evaluation of use of an AODVv2 router as an External Network Attached Router or gateway router, including network topologies including multiple gateways.
- o Achieving implementations
- o multiple and interoperable
- o deployments in different network types

- o Analysis of the effects of buffering traffic while route discovery is in progress
- o Specification of extensions to deal with timed routes, expanding ring multicast, unicast RERR to specific route precursors, accurate bidirectional metric discovery, dealing with and allowing uni-directional links and routes

The final goal of the experiment is to determine if sufficient demand exists for the AODVv2 protocol to prompt an effort to bring the protocol to Standards Track.

5. Data Structures

5.1. InterfaceSet

The InterfaceSet is a conceptual data structure which contains information about all interfaces configured for use by AODVv2. Any interface with an IP address can be used. Multiple interfaces on a single router can be used. Multiple interfaces on the same router may be configured with the same IP address.

Each element in the InterfaceSet MUST contain the following:

Interface.Id

An identifier that is unique in node-local scope and that allows the AODVv2 implementation to identify exactly one local network interface.

If multiple interfaces of the AODVv2 router are configured for use by AODVv2, they MUST be configured in the InterfaceSet.

Implementations for constrained devices using only one interface MAY choose not to use the InterfaceSet.

5.2. Router Client Set

An AODVv2 router provides route discovery services for its own local applications and for its Router Clients that are reachable without traversing another AODVv2 router. The addresses used by these devices, and the AODVv2 router itself, are configured in the Router Client Set. An AODVv2 router will only originate Route Request and Route Reply messages on behalf of configured Router Client addresses.

Router Client Set entries MUST contain:

RouterClient.IPAddress

An IP address or the start of an address range that requires route discovery services from the AODVv2 router.

RouterClient.PrefixLength

The length, in bits, of the routing prefix associated with the RouterClient.IPAddress. If the prefix length is not equal to the address length of RouterClient.IPAddress, the AODVv2 router MUST participate in route discovery on behalf of all addresses within that prefix.

RouterClient.Cost

The cost associated with reaching this address or address range.

A Router Client address MUST NOT be served by more than one AODVv2 router at any one time. To shift responsibility for a Router Client to a different AODVv2 router, correct AODVv2 routing behavior MUST be observed; The AODVv2 router adding the Router Client MUST wait for any existing routing information about this Router Client to be purged from the network, i.e., at least MAX_SEQNUM_LIFETIME since the last SeqNum update on the router that is removing this Router Client.

5.3. Neighbor Set

A Neighbor Set MUST be maintained with information about neighboring AODVv2 routers. Neighbor Set entries are stored when AODVv2 messages are received. If the Neighbor is chosen as a next hop on an installed route, the link to the Neighbor MUST be tested for bidirectionality and the result stored in this set. A route will only be considered valid when the link is confirmed to be bidirectional.

Neighbor Set entries MUST contain:

Neighbor.IPAddress

An IP address of the neighboring router, learned from the source IP address of a received route message.

Neighbor.State

Indicates whether the link to the neighbor is bidirectional. There are three possible states: Confirmed, Heard, and Blacklisted. Heard is the initial state. Confirmed indicates that the link to the neighbor has been confirmed as bidirectional. Blacklisted indicates that the link to the neighbor is unidirectional. Section 7.2 discusses how to monitor link bidirectionality.

Neighbor.Timeout

Indicates at which time the Neighbor.State should be updated:

- o If the value of Neighbor.State is Blacklisted, this indicates the time at which Neighbor.State will revert to Heard. By default this value is calculated at the time the router is blacklisted and is equal to CurrentTime + MAX_BLACKLIST_TIME.
- o If Neighbor.State is Heard, and an RREP_Ack has been requested from the neighbor, it indicates the time at which Neighbor.State will be set to Blacklisted, if an RREP_Ack has not been received.
- o If the value of Neighbor.State is Heard and no RREP_Ack has been requested, or if Neighbor.State is Confirmed, this time is set to INFINITY_TIME.

Neighbor.Interface

The interface on which the link to the neighbor was established.

Neighbor.AckSeqNum

The next sequence number to use for the TIMESTAMP value in an RREP_Ack request, in order to detect replay of an RREP_Ack response. Initially set to a random value.

Neighbor.HeardRERRSeqNum

The last heard sequence number used as the TIMESTAMP value in a RERR received from this neighbor, saved in order to detect replay of a RERR message. Initially set to zero.

See Section 13.4.4.3 and Section 13.4.4.4 for more information on how Neighbor.AckSeqNum and Neighbor.HeardRERRSeqNum are used.

5.4. Sequence Numbers

Sequence Numbers enable AODVv2 routers to determine the temporal order of route discovery messages, identifying stale routing information so that it can be discarded. The sequence number fulfills the same roles as the "Destination Sequence Number" of DSDV [Perkins94], and the AODV Sequence Number in [RFC3561].

Each AODVv2 router in the network MUST maintain its own sequence number. All RREQ and RREP messages created by an AODVv2 router include the router's sequence number, reported as a 16-bit unsigned integer. Each AODVv2 router MUST ensure that its sequence number is strictly increasing, and that it is incremented by one (1) whenever an RREQ or RREP is created, except when the sequence number is 65,535 (the maximum value of a 16-bit unsigned integer), in which case it MUST be reset to one (1) to achieve wrap around. The value zero (0) is reserved to indicate that the sequence number is unknown.

An AODVv2 router MUST only attach its own sequence number to information about a route to one of its configured Router Clients, all route messages forwarded by other routers retain the originator's sequence number.

To determine if newly received information is stale and therefore redundant, the sequence number attached to the information is compared to the sequence number of existing information about the same route. The comparison is carried out by subtracting the existing sequence number from the newly received sequence number, using unsigned arithmetic. The result of the subtraction is to be interpreted as a signed 16-bit integer.

- o If the result is negative, the newly received information is considered older than the existing information and is considered stale and redundant and MUST therefore be discarded.
- o If the result is positive, the newly received information is considered newer than the existing information and is not considered stale or redundant and MUST therefore be processed.
- o If the result is zero, the newly received information is not considered stale, and therefore MUST be processed further to determine if it is redundant. For example, it is considered redundant if the metric attached to the newly received information is higher than the metric of existing information about the same route (see Section 7.7.1 and Section 7.8).

This, along with the processes in Section 7.7.1, ensures loop freedom.

An AODVv2 router SHOULD maintain its sequence number in persistent storage. If the sequence number is lost, the router MUST follow the procedure in Section 7.1 to safely resume routing operations with a new sequence number.

5.5. Local Route Set

All AODVv2 routers MUST maintain a Local Route Set, containing information about routes learned from AODVv2 route messages. The Local Route Set is stored separately from the forwarding plane's routing table (referred to as Routing Information Base (RIB)), which may be updated by other routing protocols operating on the AODVv2 router as well. The Routing Information Base is updated using information from the Local Route Set. Alternatively, implementations MAY choose to modify the Routing Information Base directly.

Routes learned from AODVv2 route messages are referred to in this document as `LocalRoutes`, and **MUST** contain the following information:

`LocalRoute.Address`

An address, which, when combined with `LocalRoute.PrefixLength`, describes the set of destination addresses this route includes.

`LocalRoute.PrefixLength`

The prefix length, in bits, associated with `LocalRoute.Address`.

`LocalRoute.SeqNum`

The sequence number associated with `LocalRoute.Address`, obtained from the last route message that successfully updated this entry.

`LocalRoute.NextHop`

The source IP address of the IP packet containing the AODVv2 message advertising the route to `LocalRoute.Address`, i.e. an IP address of the AODVv2 router used for the next hop on the path toward `LocalRoute.Address`.

`LocalRoute.NextHopInterface`

The interface used to send IP packets toward `LocalRoute.Address`.

`LocalRoute.LastUsed`

If this route is installed in the Routing Information Base, the time it was last used to forward an IP packet.

`LocalRoute.LastSeqNumUpdate`

The time `LocalRoute.SeqNum` was last updated.

`LocalRoute.MetricType`

The type of metric associated with this route.

`LocalRoute.Metric`

The cost of the route toward `LocalRoute.Address` expressed in units consistent with `LocalRoute.MetricType`.

`LocalRoute.State`

The last known state (Unconfirmed, Idle, Active, or Invalid) of the route.

There are four possible states for a `LocalRoute`:

`Unconfirmed`

A route learned from a Route Request message, which has not yet been confirmed as bidirectional. It **MUST NOT** be used for forwarding IP packets, and therefore it is not referred to as a

valid route. This state only applies to routes learned through RREQ messages.

Idle

A route which has been learned from a route message, and has also been confirmed, but has not been used in the last ACTIVE_INTERVAL. It is able to be used for forwarding IP packets, and therefore it is referred to as a valid route.

Active

A route which has been learned from a route message, and has also been confirmed, and has been used in the last ACTIVE_INTERVAL. It is able to be used for forwarding IP packets, and therefore it is referred to as a valid route.

Invalid

A route which has expired or been lost. It MUST NOT be used for forwarding IP packets, and therefore it is not referred to as a valid route. Invalid routes contain sequence number information which allows incoming information to be assessed for freshness.

When the Local Route Set is stored separately from the Routing Information Base, routes are added to the Routing Information Base when LocalRoute.State is valid (set to Active or Idle), and removed from the Routing Information Base when LocalRoute.State becomes Invalid.

Changes to LocalRoute state are detailed in Section 7.10.1.

Multiple valid routes for the same address and prefix length but for different metric types may exist in the Local Route Set, but the decision of which of these routes to install in the Routing Information Base to use for forwarding is outside the scope of AODVv2.

5.6. Multicast Route Message Set

Route Request (RREQ) messages are multicast by default and forwarded multiple times. This set stores recently received RREQs in order that received RREQs can be tested for redundancy to avoid unnecessary processing and forwarding.

The Multicast Route Message Set is a conceptual set which contains information about previously received multicast route messages, so that incoming route messages can be compared with previously received messages to determine if the incoming information is redundant or stale, and the router can avoid sending redundant control traffic.

Multicast Route Message Set entries MUST contain the following information:

RteMsg.OrigPrefix

The prefix associated with OrigAddr, the source address of the IP packet triggering the route request.

RteMsg.OrigPrefixLen

The prefix length associated with RteMsg.OrigPrefix, originally from the Router Client entry on RREQ_Gen which includes OrigAddr.

RteMsg.TargPrefix

The prefix associated with TargAddr, the destination address of the IP packet triggering the route request. In an RREQ this MUST be set to TargAddr.

RteMsg.OrigSeqNum

The sequence number associated with the route to OrigPrefix, if RteMsg is an RREQ.

RteMsg.TargSeqNum

The sequence number associated with the route to TargPrefix.

RteMsg.MetricType

The metric type of the route requested.

RteMsg.Metric

The metric value received in the RteMsg.

RteMsg.Timestamp

The last time this Multicast Route Message Set entry was updated.

RteMsg.RemoveTime

The time at which this entry MUST be removed from the Multicast Route Message Set. This is set to CurrentTime + MAX_SEQNUM_LIFETIME, whenever the RteMsg.OrigSeqNum of this entry is updated.

RteMsg.Interface

The interface on which the message was received.

The Multicast Route Message Set is maintained so that no two entries have the same OrigPrefix, OrigPrefixLen, TargPrefix, and MetricType. See Section 7.8 for details about updating this set.

5.7. Route Error (RERR) Set

Each RERR message sent because no route exists for packet forwarding SHOULD be recorded in a conceptual set called the Route Error (RERR) Set. Each entry contains the following information:

RerrMsg.Timeout

The time after which the entry SHOULD be deleted.

RerrMsg.UnreachableAddress

The UnreachableAddress reported in the AddressList of the RERR.

RerrMsg.PktSource:

The PktSource of the RERR.

See section Section 7.9 for instructions on how to update the set.

6. Metrics

Metrics measure a cost or quality associated with a route or a link, e.g., latency, delay, financial cost, energy, etc. Metric values are reported in Route Request and Route Reply messages.

In Route Request messages, the metric describes the cost of the route from OrigPrefix to the router sending the Route Request. For RREQ_Gen, this is the cost associated with the Router Client entry which includes OrigAddr. For routers which forward the RREQ, this is the cost from OrigPrefix to the forwarding router, combining the metric value from the received RREQ message with knowledge of the link cost from the sender to the receiver, i.e., the incoming link cost. This updated route cost is included when forwarding the Route Request message, and used to install a route to OrigPrefix.

Similarly, in Route Reply messages, the metric reflects the cost of the route from TargPrefix to the router sending the Route Reply. For RREP_Gen, this is the cost associated with the Router Client entry which includes TargAddr. For routers which forward the RREP, this is the cost from TargPrefix to the forwarding router, combining the metric value from the received RREP message with knowledge of the link cost from the sender to the receiver, i.e., the incoming link cost. This updated route cost is included when forwarding the Route Reply message, and used to install a route to TargPrefix.

Assuming link metrics are symmetric, the cost of the routes installed in the Local Route Set at each router will be correct. While this assumption is not always correct, calculating incoming/outgoing metric data is outside of scope of this document. The route

discovered is optimised for the requesting router, and the return path may not be the optimal route.

AODVv2 enables the use of multiple metric types. Each route discovery attempt indicates the metric type which is requested for the route. Only one metric type MUST be used in each route discovery attempt.

For each MetricType, AODVv2 requires:

- o A MetricType number, to indicate the metric type of a route. MetricType numbers allocated are detailed in Section 11.5.
- o A maximum value, denoted MAX_METRIC[MetricType]. This MUST always be the maximum expressible metric value of type MetricType. Field lengths associated with metric values are found in Section 11.5. If the cost of a route exceeds MAX_METRIC[MetricType], the route is ignored.
- o A function for incoming link cost, denoted Cost(L). Using incoming link costs means that the route learned has a path optimized for the direction from OrigAddr to TargAddr.
- o A function for route cost, denoted Cost(R).
- o A function to analyze routes for potential loops based on metric information, denoted LoopFree(R1, R2). LoopFree verifies that a route R2 is not a sub-section of another route R1. An AODVv2 router invokes LoopFree() as part of the process in Section 7.7.1, when an advertised route (R1) and an existing LocalRoute (R2) have the same destination address, metric type, and sequence number. LoopFree returns FALSE to indicate that an advertised route is not to be used to update a stored LocalRoute, as it may cause a routing loop. In the case where the existing LocalRoute is Invalid, it is possible that the advertised route includes the existing LocalRoute and came from a router which did not yet receive notification of the route becoming Invalid, so the advertised route should not be used to update the Local Route Set, in case it forms a loop to a broken route.

AODVv2 currently supports cost metrics where Cost(R) is strictly increasing, by defining:

- o $Cost(R) := \text{Sum of } Cost(L) \text{ of each link in the route}$
- o $LoopFree(R1, R2) := (Cost(R1) <= Cost(R2))$

Implementers MAY consider other metric types, but the definitions of Cost and LoopFree functions for such types are undefined, and interoperability issues need to be considered.

7. AODVv2 Protocol Operations

The AODVv2 protocol's operations include managing sequence numbers, monitoring next hop AODVv2 routers on discovered routes and updating the Neighbor Set, performing route discovery and dealing with requests from other routers, processing incoming route information and updating the Local Route Set, updating the Multicast Route Message Set and suppressing redundant messages, and reporting broken routes. These processes are discussed in detail in the following sections.

7.1. Initialization

During initialization where an AODVv2 router does not have information about its previous sequence number, or if its sequence number is lost at any point, the router resets its sequence number to one (1). However, other AODVv2 routers may still hold sequence number information that this router previously issued. Since sequence number information is removed if there has been no update to the sequence number in MAX_SEQNUM_LIFETIME, the initializing router MUST wait for MAX_SEQNUM_LIFETIME before it creates any messages containing its new sequence number. It can then be sure that the information it sends will not be considered stale.

During this wait period, the router is permitted to do the following:

- o Process information in a received RREQ or RREP message to learn a route to the originator or target of that route discovery
- o Forward a received RREQ or RREP
- o Send an RREP_Ack
- o Maintain valid routes in the Local Route Set
- o Create, process and forward RERR messages

7.2. Next Hop Monitoring

To ensure AODVv2 routers do not establish routes over unidirectional links, AODVv2 routers MUST verify that the link to the next hop router is bidirectional before marking a route as valid in the Local Route Set.

AODVv2 provides a mechanism for testing bidirectional connectivity during route discovery, and blacklisting routers where bidirectional connectivity is not available. If a route discovery is retried by RREQ_Gen, the blacklisted routers can be excluded from the process, and a different route can be discovered. Further, a route is not to be used for forwarding until the bidirectionality of the link to the next hop is confirmed. AODVv2 routers do not need to monitor bidirectionality for links to neighboring routers which are not used as next hops on routes in the Local Route Set.

- o Bidirectional connectivity to upstream routers is tested by requesting acknowledgement of RREP messages by also sending an RREP_Ack, including an AckReq element to indicate that an acknowledgement is requested. This MUST be answered by sending an RREP_Ack in response. Receipt of an RREP_Ack within RREP_Ack_SENT_TIMEOUT proves that bidirectional connectivity exists. Otherwise, a link is determined to be unidirectional. All AODVv2 routers MUST support this process, which is explained in Section 8.2 and Section 8.3.
- o For the downstream router, receipt of an RREP message containing the route to TargAddr is confirmation of bidirectionality, since an RREP message is a reply to a RREQ message which previously crossed the link in the opposite direction.

To assist with next hop monitoring, a Neighbor Set (Section 5.3) is maintained. When an RREQ or RREP is received, search for an entry in the Neighbor Set where all of the following conditions are met:

- o Neighbor.IPAddress == IP address from which the RREQ or RREP was received
- o Neighbor.Interface == Interface on which the RREQ or RREP was received.

If such an entry does not exist, a new entry is created as described in Section 7.3. While the value of Neighbor.State is Heard, acknowledgement of RREP messages sent to that neighbor MUST be requested. If an acknowledgement is not received within the timeout period, the neighbor MUST have Neighbor.State set to Blacklisted. If an acknowledgement is received within the timeout period, Neighbor.State is set to Confirmed. While the value of Neighbor.State is Confirmed, the request for an acknowledgement of any other RREP message is unnecessary.

When routers perform other operations such as those from the list below, these MAY be used as additional indications of connectivity:

- o NHDP HELLO Messages [RFC6130]
- o Route timeout
- o Lower layer triggers, e.g. message reception or link status notifications
- o TCP timeouts
- o Promiscuous listening
- o Other monitoring mechanisms or heuristics

If such an external process signals that the link to a neighbor is bidirectional, the AODVv2 router MAY update the matching Neighbor Set entry by changing the value of Neighbor.State to Confirmed, e.g. receipt of a Neighborhood Discovery Protocol HELLO message with the receiving router listed as a neighbor. If an external process signals that a link is not bidirectional, the the value of Neighbor.State MAY be changed to Blacklisted, e.g. notification of a TCP timeout.

7.3. Neighbor Set Update

On receipt of an RREQ or RREP message, the Neighbor Set MUST be checked for an entry with Neighbor.IPAddress which matches the source IP address of a packet containing the AODVv2 message. If no matching entry is found, a new entry is created.

A new Neighbor Set entry is created as follows:

- o Neighbor.IPAddress := Source IP address of the received route message
- o Neighbor.State := Heard
- o Neighbor.Timeout := INFINITY_TIME
- o Neighbor.Interface := Interface on which the RREQ or RREP was received. MUST equal Interface.Id of one of the entries in the InterfaceSet (see Section 5.1).

When an RREP_Ack is sent to a neighbor, the Neighbor Set entry is updated as follows:

- o Neighbor.Timeout := CurrentTime + RREP_Ack_SENT_TIMEOUT

When a received message is one of the following:

- o an RREP which answers an RREQ sent within RREQ_WAIT_TIME over the same interface as Neighbor.Interface
- o an RREP_Ack response received from a Neighbor with Neighbor.State set to Heard, where Neighbor.Timeout > CurrentTime

the link to the neighbor is bidirectional and the Neighbor Set entry is updated as follows:

- o Neighbor.State := Confirmed
- o Neighbor.Timeout := INFINITY_TIME

When the Neighbor.Timeout is reached and Neighbor.State is Heard, then an RREP_Ack response has not been received from the neighbor within RREP_Ack_SENT_TIMEOUT of sending the RREP_Ack request. The link is considered to be uni-directional and the Neighbor Set entry is updated as follows:

- o Neighbor.State := Blacklisted
- o Neighbor.Timeout := CurrentTime + MAX_BLACKLIST_TIME

When the Neighbor.Timeout is reached and Neighbor.State is Blacklisted, the Neighbor Set entry is updated as follows:

- o Neighbor.State := Heard

If an external mechanism reports a link as broken, the Neighbor Set entry SHOULD be removed.

Route requests from neighbors with Neighbor.State set to Blacklisted are ignored to avoid persistent IP packet loss or protocol failures. Neighbor.Timeout allows the neighbor to again be allowed to participate in route discoveries after MAX_BLACKLIST_TIME, in case the link between the routers has become bidirectional.

7.4. Interaction with the Forwarding Plane

The signals described in the following are conceptual signals, and can be implemented in various ways. Conformant implementations of AODVv2 are not mandated to implement the forwarding plane separately from the control plane or data plane; these signals and interactions are identified simply as assistance for implementers who may find them useful.

AODVv2 requires signals from the forwarding plane:

- o A packet cannot be forwarded because a route is unavailable: AODVv2 needs to know the source and destination IP addresses of the packet. If the source of the packet is configured as a Router Client, the router should initiate route discovery to the destination. If it is not a Router Client, the router should create a Route Error message.
- o A packet is to be forwarded: AODVv2 needs to check the state of the route to ensure it is still valid.
- o Packet forwarding succeeds: AODVv2 needs to update the record of when a route was last used to forward a packet.
- o Packet forwarding failure occurs: AODVv2 needs to create a Route Error message.

AODVv2 needs to send signals to the forwarding plane:

- o A route discovery is in progress: buffering might be configured for packets requiring a route, while route discovery is attempted.
- o A route discovery failed: any buffered packets requiring that route should be discarded, and the source of the packet should be notified that the destination is unreachable (using an ICMP Destination Unreachable message). Route discovery fails if an RREQ cannot be generated because the control message generation limit has been reached, or if an RREP is not received within RREQ_WAIT_TIME (see Section 7.6).
- o A route discovery is not permitted: any buffered packets requiring that route should be discarded. A route discovery will not be attempted if the source address of the packet needing a route is not configured as a Router Client.
- o A route discovery succeeded: install a corresponding route into the Routing Information Base and begin transmitting any buffered packets.
- o A route has been made invalid: remove the corresponding route from the Routing Information Base.
- o A route has been updated: update the corresponding route in the Routing Information Base.

7.5. Message Transmission

AODVv2 sends [RFC5444] formatted messages using the parameters for port number and IP protocol specified in [RFC5498]. Mapping of AODVv2 data to [RFC5444] messages is detailed in Section 9. AODVv2 multicast messages are sent to the link-local multicast address LL-MANET-Routers [RFC5498]. All AODVv2 routers MUST subscribe to LL-MANET-Routers on all AODVv2 interfaces [RFC5498] to receive AODVv2 messages. Note that multicast messages 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.

When multiple interfaces are available, an AODVv2 router transmitting a multicast message to LL-MANET-Routers MUST send the message on all interfaces that have been configured for AODVv2 operation, as given in the InterfaceSet (Section 5.1).

To avoid congestion, each AODVv2 router's rate of message generation SHOULD be limited (CONTROL_TRAFFIC_LIMIT) and administratively configurable. Messages SHOULD NOT be sent more frequently than one message per $(1 / \text{CONTROL_TRAFFIC_LIMIT})$ th of a second. If this threshold is reached, messages MUST be sent based on their priority:

- o Highest priority SHOULD be given to RREP_Ack messages. This allows links between routers to be confirmed as bidirectional and avoids undesired blacklisting of next hop routers.
- o Second priority SHOULD be given to RERR messages for undeliverable IP packets. This avoids repeated forwarding of packets over broken routes that are still in use by other routers.
- o Third priority SHOULD be given to RREP messages in order that RREQs do not time out.
- o Fourth priority SHOULD be given to RREQ messages.
- o Fifth priority SHOULD be given to RERR messages for newly invalidated routes.
- o Lowest priority SHOULD be given to RERR messages generated in response to RREP messages which cannot be forwarded. In this case the route request will be retried at a later point.

To implement the congestion control, a queue length is set. If the queue is full, in order to queue a new message, a message of lower priority must be removed from the queue. If this is not possible,

the new message MUST be discarded. The queue should be sorted in order of message priority

7.6. Route Discovery, Retries and Buffering

AODVv2's RREQ and RREP messages are used for route discovery. RREQ messages are multicast to solicit an RREP, whereas RREP are unicast. The constants used in this section are defined in Section 11.

When an AODVv2 router needs to forward an IP packet (with source address OrigAddr and destination address TargAddr) from one of its Router Clients, it needs a route to TargAddr in its Routing Information Base. If no route exists, the AODVv2 router generates (RREQ_Gen) and multicasts a Route Request message (RREQ), on all configured interfaces, containing information about the source and destination. The procedure for this is described in Section 8.1.1. Each generated RREQ results in an increment to the router's sequence number. The AODVv2 router generating an RREQ is referred to as RREQ_Gen.

Buffering might be configured for IP packets awaiting a route for forwarding by RREQ_Gen, if sufficient memory is available. Buffering of IP packets might have both positive and negative effects. Real-time traffic, voice, and scheduled delivery may suffer if packets are buffered and subjected to delays, but TCP connection establishment will benefit if packets are queued while route discovery is performed [Koodli01]. Recommendations for appropriate buffer methods are out of scope for this specification. Determining which packets to discard first when the buffer is full is a matter of policy at each AODVv2 router. Note that using different or no buffer methods does not affect interoperability.

RREQ_Gen awaits reception of a Route Reply message (RREP) containing a route toward TargAddr. This can be achieved by monitoring the entry in the Multicast Route Message Table that corresponds to the generated RREQ. When CurrentTime exceeds RteMsg.Timestamp + RREQ_WAIT_TIME and no RREP has been received, RREQ_Gen will retry the route discovery.

To reduce congestion in a network, repeated attempts at route discovery for a particular target address utilize a binary exponential backoff: for each additional attempt, the time to wait for receipt of the RREP is multiplied by 2. If the requested route is not learned within the wait period, another RREQ is sent, up to a total of DISCOVERY_ATTEMPTS_MAX. This is the same technique used in AODV [RFC3561].

Through the use of bidirectional link monitoring and blacklists (see Section 7.2) uni-directional links on initial selected route will be ignored on subsequent route discovery attempts.

Route discovery is considered to have failed after `DISCOVERY_ATTEMPTS_MAX` and the corresponding wait time for an RREP response to the final RREQ. After the attempted route discovery has failed, `RREQ_Gen` waits at least `RREQ_HOLDDOWN_TIME` before attempting another route discovery to the same destination, in order to avoid repeatedly generating control traffic that is unlikely to discover a route. Any IP packets buffered for `TargAddr` are also dropped and a Destination Unreachable ICMP message (Type 3) with a code of 1 (Host Unreachable Error) is delivered to the source of the packet, so that the application knows about the failure.

If `RREQ_Gen` does receive a route message containing a route to `TargAddr` within the timeout, it processes the message according to Section 8. When a valid `LocalRoute` entry is created in the Local Route Set, the route is also installed in the Routing Information Base, and the router will begin sending the buffered IP packets. Any retry timers for the corresponding RREQ are then cancelled.

During route discovery, all routers on the path learn a route to both `OrigPrefix` and `TargPrefix`, so that routes are constructed in both directions. The route is optimized for the forward route.

7.7. Processing Received Route Information

All AODVv2 route messages contain a route. A Route Request (RREQ) contains a route to `OrigPrefix`, and a Route Reply (RREP) contains a route to `TargPrefix`. All AODVv2 routers that receive a route message are able to store the route contained within it in their Local Route Set. Incoming information is first checked to verify that it is both safe to use and offers an improvement to existing information, as explained in Section 7.7.1. If these checks pass, the Local Route Set MUST be updated according to Section 7.7.2.

In the processes below, `RteMsg` is used to denote the route message, `AdvRte` is used to denote the route contained within it, and `LocalRoute` denotes an existing entry in the Local Route Set which matches `AdvRte` on address, prefix length, and metric type.

`AdvRte` has the following properties:

- o `AdvRte.Address := RteMsg.OrigPrefix` (in RREQ) or `RteMsg.TargPrefix` (in RREP)

- o AdvRte.PrefixLength := RteMsg.OrigPrefixLen (in RREQ) or RteMsg.TargPrefixLen (in RREP). If no prefix length was included in RteMsg, prefix length is the address length, in bits, of RteMsg.OrigPrefix (in RREQ) or RteMsg.TargPrefix (in RREP)
- o AdvRte.SeqNum := RteMsg.OrigSeqNum (in RREQ) or RteMsg.TargSeqNum (in RREP)
- o AdvRte.NextHop := RteMsg.IPSourceAddress (an address of the sending interface of the router from which the RteMsg was received)
- o AdvRte.MetricType := RteMsg.MetricType
- o AdvRte.Metric := RteMsg.Metric
- o AdvRte.Cost := Cost(R) using the cost function associated with the route's metric type, i.e. $Cost(R) = AdvRte.Metric + Cost(L)$, as described in Section 6, where L is the link from the advertising router

7.7.1. Evaluating Route Information

An incoming advertised route (AdvRte) is compared to existing LocalRoutes to determine whether the advertised route is to be used to update the AODVv2 Local Route Set. The incoming route information MUST be processed as follows:

1. Search for LocalRoutes in the Local Route Set matching AdvRte's address, prefix length and metric type
 - * If no matching LocalRoute exists, AdvRte MUST be used to update the Local Route Set and no further checks are required.
 - * If matching LocalRoutes are found, continue to Step 2.
2. Compare sequence numbers using the technique described in Section 5.4
 - * If AdvRte is more recent than all matching LocalRoutes, AdvRte MUST be used to update the Local Route Set and no further checks are required.
 - * If AdvRte is stale, AdvRte MUST NOT be used to update the Local Route Set. Ignore AdvRte for further processing.
 - * If the sequence numbers are equal, continue to Step 3.

3. Check that AdvRte is safe against routing loops compared to all matching LocalRoutes (see Section 6)
 - * If LoopFree(AdvRte, LocalRoute) returns FALSE, ignore AdvRte for further processing. AdvRte MUST NOT be used to update the Local Route Set because using the incoming information might cause a routing loop.
 - * If LoopFree(AdvRte, LocalRoute) returns TRUE, continue to Step 4.
4. Compare route costs
 - * If AdvRte is better than all matching LocalRoutes, it MUST be used to update the Local Route Set because it offers improvement.
 - * If AdvRte is equal in cost and LocalRoute is valid, AdvRte SHOULD NOT be used to update the Local Route Set because it will offer no improvement.
 - * If AdvRte is worse and LocalRoute is valid, ignore AdvRte for further processing. AdvRte MUST NOT be used to update the Local Route Set because it does not offer any improvement.
 - * If AdvRte is not better (i.e., it is worse or equal) but LocalRoute is Invalid, AdvRte SHOULD be used to update the Local Route Set because it can safely repair the existing Invalid LocalRoute.

If the advertised route is to be used to update the Local Route Set, the procedure in Section 7.7.2 MUST be followed. If not, non-optimal routes will remain in the Local Route Set.

For information on how to apply these changes to the Routing Information Base, see Section 5.5.

7.7.2. Applying Route Updates

After determining that AdvRte is to be used to update the Local Route Set (as described in Section 7.7.1), the following procedure applies.

If AdvRte is learned from an RREQ message, the link to the next hop neighbor may not be confirmed as bidirectional (see Section 5.3). If there is no existing matching route in the Local Route Set, AdvRte MUST be installed to allow a corresponding RREP to be sent. If a matching entry already exists, AdvRte offers potential improvement, if the link to the neighbor can be confirmed as bidirectional.

The route update is applied as follows:

1. If no existing entry in the Local Route Set matches AdvRte's address, prefix length and metric type, continue to Step 4 and create a new entry in the Local Route Set.
2. If two matching LocalRoutes exist in the Local Route Set, one is a valid route, and one is an Unconfirmed route, AdvRte may offer further improvement to the Unconfirmed route, or may offer an update to the valid route.
 - * If AdvRte.NextHop's Neighbor.State is Heard, the advertised route may offer improvement to the existing valid route, if the link to the next hop can be confirmed as bidirectional. Continue processing from Step 5 to update the existing Unconfirmed LocalRoute.
 - * If AdvRte.NextHop's Neighbor.State is Confirmed, the advertised route offers an update or improvement to the existing valid route. Continue processing from Step 5 to update the existing valid LocalRoute.
3. If only one matching LocalRoute exists in the Local Route Set:
 - * If AdvRte.NextHop's Neighbor.State is Confirmed, continue processing from Step 5 to update the existing LocalRoute.
 - * If AdvRte.NextHop's Neighbor.State is Heard, AdvRte may offer improvement the existing LocalRoute, if the link to AdvRte.NextHop can be confirmed as bidirectional.
 - * If LocalRoute.State is Unconfirmed, AdvRte is an improvement to an existing Unconfirmed route. Continue processing from Step 5 to update the existing LocalRoute.
 - * If LocalRoute.State is Invalid, AdvRte can replace the existing LocalRoute. Continue processing from Step 5 to update the existing LocalRoute.
 - * If LocalRoute.State is Active or Idle, AdvRte SHOULD be stored as an additional entry in the Local Route Set, with LocalRoute.State set to Unconfirmed. Continue processing from Step 4 to create a new LocalRoute.
4. Create an entry in the Local Route Set and initialize as follows:
 - * LocalRoute.Address := AdvRte.Address

- * LocalRoute.PrefixLength := AdvRte.PrefixLength
 - * LocalRoute.MetricType := AdvRte.MetricType
5. Update the LocalRoute as follows:
 - * LocalRoute.SeqNum := AdvRte.SeqNum
 - * LocalRoute.NextHop := AdvRte.NextHop
 - * LocalRoute.NextHopInterface := interface on which RteMsg was received
 - * LocalRoute.Metric := AdvRte.Cost
 - * LocalRoute.LastUsed := CurrentTime
 - * LocalRoute.LastSeqNumUpdate := CurrentTime
 6. If a new LocalRoute was created, or if the existing LocalRoute.State is Invalid or Unconfirmed, update LocalRoute as follows:
 - * LocalRoute.State := Unconfirmed (if the next hop's Neighbor.State is Heard)
 - * LocalRoute.State := Idle (if the next hop's Neighbor.State is Confirmed)
 7. If an existing LocalRoute.State changed from Invalid or Unconfirmed to become Idle, any matching Unconfirmed LocalRoute with worse metric value SHOULD be expunged.
 8. If an existing LocalRoute was updated with a better metric value, any matching Unconfirmed LocalRoute with worse metric value SHOULD be expunged.
 9. If this update results in LocalRoute.State of Active or Idle, which matches a route request which is still in progress, the associated route request retry timers MUST be cancelled.

If this update to the Local Route Set results in two LocalRoutes to the same address, the best LocalRoute will be Unconfirmed. In order to improve the route used for forwarding, the router SHOULD try to determine if the link to the next hop of that LocalRoute is bidirectional, by using that LocalRoute to forward future RREPs and request acknowledgements (see Section 8.2.1 and Section 8.3).

7.8. Suppressing Redundant Messages Using the Multicast Route Message Set

When route messages are flooded in a MANET, an AODVv2 router may receive several instances of the same message. Forwarding every one of these gives little additional benefit, and generates unnecessary signaling traffic and might generate unnecessary interference.

Each AODVv2 router stores information about recently received route messages in the AODVv2 Multicast Route Message Set (Section 5.6).

Entries in the Multicast Route Message Set SHOULD be maintained for at least `RteMsg_ENTRY_TIME` after the last Timestamp update in order to account for long-lived RREQs traversing the network. An entry MUST be deleted when the sequence number is no longer valid, i.e., after `MAX_SEQNUM_LIFETIME`. Memory-constrained devices MAY remove the entry before this time.

Received route messages are tested against previously received route messages, and if determined to be redundant, forwarding or response can be avoided.

To determine if a received message is redundant:

1. Search for an entry in the Multicast Route Message Set with the same `OrigPrefix`, `OrigPrefixLen`, `TargPrefix`, `Interface` and `MetricType`
 - * If there is no entry, the message is not redundant.
 - * If there is an entry, continue to Step 2.
2. Compare sequence numbers using the technique described in Section 5.4
 - * Use `OrigSeqNum` of the entry for comparison.
 - * If the entry has an older sequence number than the received message, the message is not redundant.
 - * If the entry has a newer sequence number than the received message, the message is redundant.
 - * If the entry has the same sequence number, continue to Step 3.
3. Compare the metric values

- * If the entry has a Metric value that is worse than or equal to the metric in the received message, the message is redundant.
- * If the entry has a Metric value that is better than the metric in the received message, the message is not redundant.

If the message is redundant, update the entry as follows:

- o RteMsg.Timestamp := CurrentTime
- o RteMsg.RemoveTime := CurrentTime + MAX_SEQNUM_LIFETIME

since matching route messages are still traversing the network and this entry should be maintained. This message MUST NOT be forwarded or responded to.

If the message is not redundant, create an entry or update the existing entry.

To update a Multicast Route Message Set entry, set:

- o RteMsg.OrigPrefix := OrigPrefix from the message
- o RteMsg.OrigPrefixLen := the prefix length associated with OrigPrefix
- o RteMsg.TargPrefix := TargPrefix from the message
- o RteMsg.OrigSeqNum := the sequence number associated with OrigPrefix, if RteMsg is an RREQ
- o RteMsg.TargSeqNum := the sequence number associated with TargPrefix, if RteMsg is an RREP
- o RteMsg.Metric := the metric value associated with OrigPrefix in a received RREQ
- o RteMsg.MetricType := the metric type associated with RteMsg.Metric
- o RteMsg.Timestamp := CurrentTime
- o RteMsg.RemoveTime := CurrentTime + MAX_SEQNUM_LIFETIME

Where the message is determined not redundant before Step 3, it MUST be forwarded or responded to. When a message is determined to be not redundant in Step 3, it MAY be suppressed to avoid extra control traffic. However, since the processing of the message will result in an update to the Local Route Set, the message SHOULD be forwarded or

responded to, to ensure other routers have up-to-date information and the best metrics. If the message is not forwarded, the best route may not be found. Forwarding or response is to be performed using the processes outlined in Section 8.

7.9. Suppressing Redundant Route Error Messages using the Route Error Set

In order to avoid flooding the network with RERR messages when a stream of IP packets to an unreachable address arrives, an AODVv2 router SHOULD avoid creating duplicate messages by determining whether an equivalent RERR has recently been sent. This is achieved with the help of the Route Error Set (see Section 5.7).

To determine if a RERR should be created:

1. Search for an entry in the Route Error Set where:

- * RerrMsg.UnreachableAddress == UnreachableAddress to be reported
- * RerrMsg.PktSource == PktSource to be included in the RERR

If a matching entry is found, no further processing is required and the RERR SHOULD NOT be sent.

2. If no matching entry is found, a new entry with the following properties is created, and the RERR is created and sent as described in Section 8.4.1:

- * RerrMsg.Timeout := CurrentTime + RERR_TIMEOUT
- * RerrMsg.UnreachableAddress == UnreachableAddress to be reported
- * RerrMsg.PktSource == PktSource to be included in the RERR

7.10. Local Route Set Maintenance

Route maintenance involves monitoring LocalRoutes in the Local Route Set, updating LocalRoute.State to handle route timeouts and reporting routes that become Invalid.

7.10.1. LocalRoute State Changes

During normal operation, AODVv2 does not require any explicit timeouts to manage the lifetime of a route. At any time, any LocalRoute MAY be examined and updated according to the rules below.

If timers are not used to prompt updates of `LocalRoute.State`, the `LocalRoute.State` MUST be checked before IP packet forwarding and before any operation based on `LocalRoute.State`.

Route timeout behaviour is as follows:

- o An Unconfirmed route MUST be expunged at `MAX_SEQNUM_LIFETIME` after `LocalRoute.LastSeqNumUpdate`.
 - o An Idle route MUST become Active when used to forward an IP packet. If the route is not used to forward an IP packet within `MAX_IDLETIME`, `LocalRoute.State` MUST become Invalid.
 - o An Invalid route SHOULD remain in the Local Route Set, since `LocalRoute.SeqNum` is used to classify future information about `LocalRoute.Address` as stale or fresh.
 - o In all cases, if the time since `LocalRoute.LastSeqNumUpdate` exceeds `MAX_SEQNUM_LIFETIME`, `LocalRoute.SeqNum` must be set to
1. This is required to ensure that any AODVv2 routers following the initialization procedure can safely begin routing functions using a new sequence number. A `LocalRoute` with `LocalRoute.State` set to Active or Idle can remain in the Local Route Set after the sequence number has been set to 0, for example if the route is reliably carrying traffic. If `LocalRoute.State` is Invalid, or later becomes Invalid, the `LocalRoute` MUST be expunged from the Local Route Set.

LocalRoutes can become Invalid before a timeout occurs:

- o If an external mechanism reports a link as broken, all `LocalRoutes` using that link for `LocalRoute.NextHop` MUST immediately have `LocalRoute.State` set to Invalid.
- o `LocalRoute.State` MUST immediately be set to Invalid if a Route Error (RERR) message is received where:
 - * The sender is `LocalRoute.NextHop` or `PktSource` is a Router Client address
 - * There is an `Address` in `AddressList` which matches `LocalRoute.Address`, and:
 - + The prefix length associated with this `Address`, if any, matches `LocalRoute.PrefixLength`

- + The sequence number associated with this Address, if any, is newer or equal to `LocalRoute.SeqNum` (see Section 5.4)
- + The metric type associated with this Address matches `LocalRoute.MetricType`

LocalRoutes are also updated when `Neighbor.State` is updated:

- o While the value of `Neighbor.State` is set to `Heard`, any routes in the Local Route Set using that neighbor as a next hop MUST have `LocalRoute.State` set to `Unconfirmed`.
- o When the value of `Neighbor.State` is set to `Confirmed`, the `Unconfirmed` routes in the Local Route Set using that neighbor as a next hop MUST have `LocalRoute.State` set to `Idle`. Any other matching LocalRoutes with metric values worse than `LocalRoute.Metric` MUST be expunged from the Local Route Set.
- o When the value of `Neighbor.State` is set to `Blacklisted`, any valid routes in the Local Route Set using that neighbor for their next hop MUST have `LocalRoute.State` set to `Invalid`.
- o When a Neighbor Set entry is removed, all routes in the Local Route Set using that neighbor as next hop MUST have `LocalRoute.State` set to `Invalid`.

Memory constrained devices MAY choose to expunge routes from the AODVv2 Local Route Set at other times, but MUST adhere to the following rules:

- o An Active route MUST NOT be expunged, as it is in use. If deleted, IP traffic forwarded to this router will prompt generation of a Route Error message, and it will be necessary for a Route Request to be generated by the originator's router to re-establish the route.
- o An Idle route SHOULD NOT be expunged, as it is still valid for forwarding IP traffic. If deleted, this could result in dropped IP packets and a Route Request could be generated to re-establish the route.
- o Any Invalid route MAY be expunged. Least recently used Invalid routes SHOULD be expunged first, since the sequence number information is less likely to be useful.
- o An Unconfirmed route MUST NOT be expunged if it was installed within the last `RREQ_WAIT_TIME`, because it may correspond to a route discovery in progress. A Route Reply message might be

received which needs to use the LocalRoute.NextHop information. Otherwise, it MAY be expunged.

7.10.2. Reporting Invalid Routes

When LocalRoute.State changes from Active to Invalid as a result of a broken link or a received Route Error (RERR) message, other AODVv2 routers MUST be informed by sending an RERR message containing details of the invalidated route.

An RERR message MUST also be sent when an AODVv2 router receives an RREP message to forward, but the LocalRoute to the OrigPrefix in the RREP has been lost or is marked as Invalid.

An RERR message MUST also be sent when an AODVv2 router receives an RREP message to forward, but the LocalRoute to the OrigAddr in the RREP has been lost or is marked as Invalid.

The packet or message triggering the RERR MUST be discarded.

Generation of an RERR message is described in Section 8.4.1.

8. AODVv2 Protocol Messages

AODVv2 defines four message types: Route Request (RREQ), Route Reply (RREP), Route Reply Acknowledgement (RREP_Ack), and Route Error (RERR).

Each AODVv2 message is defined as a set of data. Rules for the generation, reception and forwarding of each message type are described in the following sections. Section 9 discusses how the data is mapped to [RFC5444] Message TLVs, Address Blocks, and Address TLVs.

8.1. Route Request (RREQ) Message

Route Request messages are used in route discovery operations to request a route to a specified target address. RREQ messages have the following contents:

msg_hop_limit
AddressList
PrefixLengthList (optional)
OrigSeqNum, (optional) TargSeqNum
MetricType
OrigMetric

Figure 1: RREQ message contents

msg_hop_limit

The remaining number of hops allowed for dissemination of the RREQ message.

AddressList

Contains OrigPrefix, from the Router Client entry which includes OrigAddr, the source address of the IP packet for which a route is requested, and TargPrefix, set to TargAddr, the destination address of the IP packet for which a route is requested.

PrefixLengthList

Contains OrigPrefixLen, i.e., the length, in bits, of the prefix associated with the Router Client entry which includes OrigAddr. If omitted, the prefix length is equal to OrigAddr's address length in bits.

OrigSeqNum

The sequence number associated with OrigPrefix.

TargSeqNum

A sequence number associated with an existing Invalid route to TargAddr. This MAY be included if available.

MetricType

The metric type associated with OrigMetric.

OrigMetric

The metric value associated with the route to OrigPrefix, as seen from the sender of the message.

8.1.1.1. RREQ Generation

An RREQ is generated when an IP packet needs to be forwarded for a Router Client, and no valid route currently exists for the packet's destination in the Routing Information Base.

Before creating an RREQ, the router SHOULD check the Multicast Route Message Set to see if an RREQ has recently been sent for the requested destination. If so, and the wait time for a reply has not yet been reached, the router SHOULD continue to await a response without generating a new RREQ. If the timeout has been reached, a new RREQ MAY be generated. If buffering is configured, incoming IP packets awaiting this route SHOULD be buffered until the route discovery is completed.

If the limit for the rate of AODVv2 control message generation has been reached, no message SHOULD be generated.

To generate the RREQ, the router (referred to as RREQ_Gen) follows this procedure:

1. Set `msg_hop_limit` := `MAX_HOPCOUNT`
2. Set `AddressList` := {`OrigPrefix`, `TargPrefix`}
3. For the `PrefixLengthList`:
 - * If `OrigAddr` is part of an address range configured as a Router Client, set `PrefixLengthList` := {`RouterClient.PrefixLength`, `null`}.
 - * Otherwise, omit `PrefixLengthList`.
4. For `OrigSeqNum`:
 - * Increment the router Sequence Number as specified in Section 5.4.
 - * Set `OrigSeqNum` := router Sequence Number.
5. For `TargSeqNum`:
 - * If an Invalid route exists in the Local Route Set matching `TargAddr` using longest prefix matching and has a valid sequence number, set `TargSeqNum` := `LocalRoute.SeqNum`.

- * If no Invalid route exists in the Local Route Set matching TargAddr, or the route doesn't have a sequence number, omit TargSeqNum.
6. Include MetricType and set the type accordingly
 7. Find the Router Client Set Entry where RouterClient.IPAddress == OrigPrefix:
 - * Set OrigMetric := RouterClient.Cost

This AODVv2 message is used to create a corresponding [RFC5444] message (see Section 9) which is handed to the RFC5444 multiplexer for further processing. By default, the multiplexer is instructed to multicast the message to LL-MANET- Routers on all interfaces configured for AODVv2 operation. The RREP MUST be sent over LocalRoute[OrigPrefix].NextHopInterface.

8.1.2. RREQ Reception

Upon receiving a Route Request, an AODVv2 router performs the following steps:

1. Check and update the Neighbor Set according to Section 7.3
 - * If the sender has Neighbor.State set to Blacklisted, ignore this RREQ for further processing.
2. Verify that the message contains the required data: msg_hop_limit, OrigPrefix, TargPrefix, OrigSeqNum, and OrigMetric, and that OrigPrefix and TargPrefix are valid addresses
 - * If not, ignore this RREQ for further processing.
3. Check that the MetricType is supported and configured for use
 - * If not, ignore this RREQ for further processing.
4. Verify that the cost of the advertised route will not exceed the maximum allowed metric value for the metric type (Metric <= MAX_METRIC[MetricType] - Cost(L))
 - * If it will, ignore this RREQ for further processing.
5. Process the route to OrigPrefix as specified in Section 7.7

6. Check if the information in the message is redundant by comparing to entries in the Multicast Route Message Set, following the procedure in Section 7.8
 - * If redundant, ignore this RREQ for further processing.
 - * If not redundant, create a new entry in the Multicast Route Message Set and continue processing.
7. Check if the TargPrefix matches an entry in the Router Client Set
 - * If so, generate an RREP as specified in Section 8.2.1.
 - * If not, continue to RREQ forwarding.

8.1.3. RREQ Forwarding

By forwarding an RREQ, a router advertises that it will forward IP packets to the OrigPrefix contained in the RREQ according to the information enclosed. The router MAY choose not to forward the RREQ, for example if the router is heavily loaded or low on energy and therefore unwilling to advertise routing capability for more traffic. This could, however, decrease connectivity in the network or result in non-optimal paths.

The RREQ SHOULD NOT be forwarded if the limit for the rate of AODVv2 control message generation has been reached.

The procedure for RREQ forwarding is as follows:

1. Set `msg_hop_limit := received msg_hop_limit - 1`
2. If `msg_hop_limit` is now zero, do not continue the forwarding process
3. Set `OrigMetric := LocalRoute[OrigPrefix].Metric`

This modified message is handed to the [RFC5444] multiplexer for further processing. By default, the multiplexer is instructed to multicast the message to LL-MANET-Routers on all interfaces configured for AODVv2 operation.

8.2. Route Reply (RREP) Message

When a Route Request message is received, requesting a route to a target address (TargAddr) which is configured as part of a Router Client entry, a Route Reply message is sent in response. The RREP offers a route to TargPrefix.

RREP messages have the following contents:

msg_hop_limit
AddressList
PrefixLengthList (optional)
TargSeqNum
MetricType
TargMetric

Figure 2: RREP message contents

msg_hop_limit

The remaining number of hops allowed for dissemination of the RREP message.

AddressList

Contains OrigPrefix and TargPrefix, the prefixes of the source and destination addresses of the IP packet for which a route is requested.

PrefixLengthList

Contains TargPrefixLen, i.e., the length, in bits, of the prefix associated with the Router Client entry which includes TargAddr. If omitted, the prefix length is equal to TargAddr's address length, in bits.

TargSeqNum

The sequence number associated with TargPrefix.

MetricType

The metric type associated with TargMetric.

TargMetric

The metric value associated with the route to TargPrefix, as seen from the sender of the message.

8.2.1. RREP Generation

A Route Reply message is generated when a Route Request for a Router Client of the AODVv2 router arrives. This is the case when

RteMsg.TargPrefix matches an entry in the Router Client Set of the AODVv2 router.

Before creating an RREP, the router SHOULD check if CONTROL_TRAFFIC_LIMIT has been reached. If so, the RREP SHOULD NOT be created.

The RREP will follow the path of the route to OrigPrefix. If the best route to OrigPrefix in the Local Route Set is Unconfirmed, the link to the next hop neighbor is not yet confirmed as bidirectional (as described in Section 7.2). In this case an RREP_Ack MUST also be sent as described in Section 8.3, in order to request an acknowledgement message from the next hop router to prove that the link is bidirectional. If the best route to OrigPrefix in the Local Route Set is valid, the link to the next hop neighbor is already confirmed as bidirectional, and no acknowledgement is required.

Implementations MAY allow a number of retries of the RREP if a requested acknowledgement is not received within RREP_Ack_SENT_TIMEOUT, doubling the timeout with each retry, up to a maximum of RREP_RETRIES, using the same exponential backoff described in Section 7.6 for RREQ retries. The acknowledgement MUST be considered to have failed after the wait time for an RREP_Ack response to the final RREP.

To generate the RREP, the router (also referred to as RREP_Gen) follows this procedure:

1. Set msg_hop_limit := MAX_HOPCOUNT - msg_hop_limit from the received RREQ message
2. Set Address List := {OrigPrefix, TargPrefix}
3. For the PrefixLengthList:
 - * If TargAddr is part of an address range configured as a Router Client, set PrefixLengthList := {null, RouterClient.PrefixLength}.
 - * Otherwise, omit PrefixLengthList.
4. For the TargSeqNum:
 - * Increment the router Sequence Number as specified in Section 5.4.
 - * Set TargSeqNum := router Sequence Number.

5. Include MetricType and set the type to match the MetricType in the received RREQ message
6. Set TargMetric := RouterClient.Cost for the Router Client entry which includes TargAddr

This AODVv2 message is used to create a corresponding [RFC5444] message (see Section 9) which is handed to the RFC5444 multiplexer for further processing. The multiplexer is instructed to unicast the RREP to LocalRoute[OrigPrefix].NextHop. The RREP MUST be sent over LocalRoute[OrigPrefix].NextHopInterface.

8.2.2. RREP Reception

Upon receiving a Route Reply, an AODVv2 router performs the following steps:

1. Verify that the message contains the required data: msg_hop_limit, OrigPrefix, TargPrefix, TargSeqNum, and TargMetric, and that OrigPrefix and TargPrefix are valid addresses
 - * If not, ignore this RREP for further processing.
2. Check that the MetricType is supported and configured for use
 - * If not, ignore this RREP for further processing. <!--
3. If this RREP does not correspond to an RREQ generated or forwarded in the last RREQ_WAIT_TIME, ignore for further processing. -->
4. If the Multicast Route Message Set does not contain an entry where:
 - o RteMsg.OrigPrefix == RREP.OrigPrefix
 - o RteMsg.OrigPrefixLen == RREP.OrigPrefixLen
 - o RteMsg.TargAddr exists within RREP.TargPrefix
 - o RteMsg.OrigSeqNum <= RREP.OrigSeqNum
 - o RteMsg.MetricType == RREP.MetricType
 - o RteMsg.Timestamp > CurrentTime - RREQ_WAIT_TIME
 - o RteMsg.Interface == The interface on which the RREP was received

ignore this RREP for further processing, since it does not correspond to a previously sent RREQ.

1. Update the Neighbor Set according to Section 7.3
2. Verify that the cost of the advertised route does not exceed the maximum allowed metric value for the metric type ($\text{Metric} \leq \text{MAX_METRIC}[\text{MetricType}] - \text{Cost}(L)$)
 - * If it does, ignore this RREP for further processing.
3. Process the route to TargPrefix as specified in Section 7.7
4. Check if the message is redundant by comparing to entries in the Multicast Route Message Set (Section 7.8)
 - * If redundant, ignore this RREP for further processing.
 - * If not redundant, save the information in the Multicast Route Message Set to identify future redundant RREP messages and continue processing.
5. Check if the OrigPrefix matches an entry in the Router Client Set
 - * If so, no further processing is necessary.
 - * If not, continue to Step 10.
6. Check if a valid (Active or Idle) or Unconfirmed LocalRoute exists to OrigPrefix
 - * If so, continue to RREP forwarding.
 - * If not, a Route Error message SHOULD be transmitted toward TargPrefix according to Section 8.4.1 and the RREP SHOULD be discarded and not forwarded.

8.2.3. RREP Forwarding

A received Route Reply message is forwarded toward OrigPrefix. By forwarding an RREP, a router advertises that it will forward IP packets to TargPrefix.

The RREP SHOULD NOT be forwarded if CONTROL_TRAFFIC_LIMIT has been reached. Otherwise, the router MUST forward the RREP.

The procedure for RREP forwarding is as follows:

The RREP_Ack request MUST be sent unicast to the LocalRoute[OrigPrefix].NextHop via LocalRoute[OrigPrefix].NextHopInterface. The multiplexer MAY be instructed to send the RREP_Ack in the same [RFC5444] packet as the RREP.

The Neighbor Set entry for LocalRoute[OrigPrefix].NextHop MUST also be updated to indicate that an RREP_Ack is required (see Section 7.3).

8.3.2. RREP_Ack Reception

Upon receiving an RREP_Ack, an AODVv2 router performs the following steps:

1. Check if an AckReq element is included:
 - * If so, create an RREP_Ack Response as described in Section 8.3.3. No further processing is required.
 - * If not, continue to step 2.
2. Check if the RREP_Ack was expected:
 - * Check if the Neighbor Set contains an entry where:
 - + Neighbor.IPAddress == IP.SourceAddress of the RREP_Ack message
 - + Neighbor.State == Heard
 - + Neighbor.Timeout < CurrentTime
 - + Neighbor.Interface matches the interface on which the RREP_Ack was received
 - * If it does, the router sets Neighbor.Timeout to INFINITY_TIME, and processing continues to Step 3.
 - * Otherwise no actions are required and processing ends.
3. Update the Neighbor Set according to Section 7.3, including updating routes using this Neighbor as LocalRoute.NextHop.

The sequence numbers of the routes not available through RERR_Gen (where known).

MetricTypeList

The metric types associated with the routes not available through RERR_Gen.

8.4.1. RERR Generation

A Route Error message is generated when an AODVv2 router (also referred to as RERR_Gen) needs to report that a destination is not reachable. There are three events that cause this response:

- o When an IP packet that has been forwarded from another router, but cannot be forwarded further because there is no valid route in the Routing Information Base for its destination, the source of the packet needs to be informed that the route to the destination of the packet does not exist. The RERR generated MUST include PktSource set to the source address of the IP packet, and MUST contain only one unreachable address in the AddressList, i.e., the destination address of the IP packet. RERR_Gen MUST discard the IP packet that triggered generation of the RERR. The prefix length, sequence number and metric type SHOULD be included if known from an existing Invalid LocalRoute to the unreachable address.
- o When an RREP message cannot be forwarded because the LocalRoute to OrigPrefix has been lost or is Invalid, RREP_Gen needs to be informed that the route to OrigPrefix does not exist. The RERR generated MUST include PktSource set to the TargPrefix of the RREP, and MUST contain only one unreachable address in the AddressList, the OrigPrefix from the RREP. RERR_Gen MUST discard the RREP message that triggered generation of the RERR. The prefix length, sequence number and metric type SHOULD be included if known from an Invalid LocalRoute to the unreachable address.
- o When a link breaks, multiple LocalRoutes may become Invalid, and the RERR generated MAY contain multiple unreachable addresses. The RERR MUST include MetricTypeList. PktSource is omitted. All previously Active LocalRoutes that used the broken link MUST be reported. The AddressList, PrefixLengthList, SeqNumList, and MetricTypeList will contain entries for each LocalRoute which has become Invalid. An RERR message is only sent if an Active LocalRoute becomes Invalid, though an AODVv2 router can also include Idle LocalRoutes that become Invalid if the configuration parameter ENABLE_IDLE_IN_RERR is set (see Section 11.3).

The RERR SHOULD NOT be generated if CONTROL_TRAFFIC_LIMIT has been reached. The RERR also SHOULD NOT be generated if it is a duplicate, as determined by Section 7.9.

Incidentally, if an AODVv2 router receives an ICMP error packet to or from the address of one of its Router Clients, it forwards the ICMP packet in the same way as any other IP packet, and will not generate any RERR message based on the contents of the ICMP packet.

To generate the RERR, the router follows this procedure:

1. If necessary, include PktSource and set the value as given above
2. For each LocalRoute that needs to be reported:
 - * Insert LocalRoute.Address into the AddressList.
 - * Insert LocalRoute.PrefixLength into PrefixLengthList, if known and not equal to the address length.
 - * Insert LocalRoute.SeqNum into SeqNumList, if known.
 - * Insert LocalRoute.MetricType into MetricTypeList.

The AODVv2 message is used to create a corresponding [RFC5444] message (see Section 9).

If the RERR is sent in response to an undeliverable IP packet or RREP message, i.e., if PktSource is included, the RERR SHOULD be sent unicast to the next hop on the route to PktSource. It MUST be sent over the same interface on which the undeliverable IP packet was received. If there is no route to PktSource, the RERR SHOULD be multicast to LL-MANET-Routers. If the RERR is sent in response to a broken link, i.e., PktSource is not included, the RERR is, by default, multicast to LL-MANET-Routers.

8.4.2. RERR Reception

Upon receiving a Route Error, an AODVv2 router performs the following steps:

1. Verify that the message contains the required data: at least one unreachable address
 - * If not, ignore this RERR for further processing.
2. For each address in the AddressList, check that:

- * The address is valid (routable and unicast)
- * The MetricType is supported and configured for use
- * There is a LocalRoute with the same MetricType matching the address using longest prefix matching
- * Either the LocalRoute's next hop is the sender of the RERR and the next hop interface is the interface on which the RERR was received, or PktSource is present in the RERR and is a Router Client address
- * The unreachable address' sequence number is either unknown, or is greater than the LocalRoute's sequence number

If any of the above are false the address does not match a LocalRoute and MUST NOT be processed or regenerated in a RERR.

If all of the above are true, the LocalRoute which matches the address is no longer valid. If the LocalRoute was previously Active, it MUST be reported in a regenerated RERR. If the LocalRoute was previously Idle, it MAY be reported in a regenerated RERR, if ENABLE_IDLE_IN_RERR is configured. The Local Route Set MUST be updated according to these rules:

- * If the LocalRoute's prefix length is the same as the unreachable address' prefix length, set LocalRoute.State to Invalid.
 - * If the LocalRoute's prefix length is longer than the unreachable address' prefix length, the LocalRoute MUST be expunged from the Local Route Set, since it is a sub-route of the route which is reported to be Invalid.
 - * If the prefix length is different, create a new LocalRoute with the unreachable address, and its prefix length and sequence number, and set LocalRoute.State to Invalid. These Invalid routes are retained to avoid processing stale messages.
 - * Update the sequence number on the existing LocalRoute, if the reported sequence number is determined to be newer using the comparison technique described in Section 5.4.
3. If there are previously Active LocalRoutes that MUST be reported, as identified in step 2.:
 - * Regenerate the RERR as detailed in Section 8.4.3.

8.4.3. RERR Regeneration

The Route Error message SHOULD NOT be regenerated if CONTROL_TRAFFIC_LIMIT has been reached.

The procedure for RERR regeneration is as follows:

1. If PktSource was included in the original RERR, and PktSource is not a Router Client, copy it into the regenerated RERR
2. For each LocalRoute that needs to be reported as identified in Section 8.4.1:
 - * Insert LocalRoute.Address into the AddressList.
 - * Insert LocalRoute.PrefixLength into PrefixLengthList, if known and not equal to the address length.
 - * Insert LocalRoute.SeqNum into SeqNumList, if known.
 - * Insert LocalRoute.MetricType into MetricTypeList.

The AODVv2 message is used to create a corresponding [RFC5444] message (see Section 9). If the RERR contains PktSource, the regenerated RERR SHOULD be sent unicast to the next hop on the LocalRoute to PktSource. It MUST be sent over the same interface on which the undeliverable IP packet was received. If there is no route to PktSource, or PktSource is a Router Client, it SHOULD be multicast to LL-MANET-Routers. If the RERR is sent in response to a broken link, the RERR is, by default, multicast to LL-MANET-Routers.

9. RFC 5444 Representation

AODVv2 specifies that all control messages between routers MUST use the Generalized Mobile Ad Hoc Network Packet/Message Format [RFC5444], and therefore AODVv2's route messages comprise data which is mapped to message elements in [RFC5444].

[RFC5444] provides a multiplexed transport for multiple protocols. An [RFC5444] implementation MAY choose to optimize the content of certain elements during message creation to reduce control message overhead.

A brief summary of the [RFC5444] format:

1. A packet contains zero or more messages

2. A message contains a Message Header, one Message TLV Block, zero or more Address Blocks, and one Address Block TLV Block per Address Block
3. The Message TLV Block MAY contain zero or more Message TLVs
4. An Address Block TLV Block MAY include zero or more Address Block TLVs
5. Each TLV value in an Address Block TLV Block can be associated with all of the addresses, or with a contiguous set of addresses, or with a single address in the Address Block

AODVv2 does not require access to the [RFC5444] packet header.

In the message header, AODVv2 uses <msg-type>, <msg-hop-limit> and <msg-addr-length>. The <msg-addr-length> field indicates the length of any addresses in the message, using <msg-addr-length> := (address length in octets - 1), i.e. 3 for IPv4 and 15 for IPv6.

The addresses in an Address Block MAY appear in any order, and values in a TLV in the Address Block TLV Block must be associated with the correct address in the Address Block by the [RFC5444] implementation. To indicate which value is associated with each address, the AODVv2 message representation uses lists where the order of the addresses in the AODVv2 AddressList matches the order of values in other data lists, e.g., the order of SeqNums in the SeqNumList in an RERR. [RFC5444] maps this information to Address Block TLVs associated with the relevant addresses in the Address Block.

Each address included in the Address Block is identified as OrigPrefix, TargPrefix, PktSource, or Unreachable Address by including an ADDRESS_TYPE TLV in the Address Block TLV Block.

The following sections show how AODVv2 data is represented in [RFC5444] messages. AODVv2 defines (in Section 11.8) a number of new TLVs.

Where the extension type of a TLV is set to zero, this is the default [RFC5444] value and the extension type will not be included in the message.

9.1. Route Request Message Representation

9.1.1.1. Message Header

Data	Header Field	Value
None	<msg-type>	RREQ
msg_hop_limit	<msg-hop-limit>	MAX_HOPCOUNT, reduced by number of hops traversed so far by the message.

9.1.1.2. Message TLV Block

AODVv2 does not define any Message TLVs for an RREQ message.

9.1.1.3. Address Block

An RREQ contains OrigPrefix and TargPrefix, and each of these addresses has an associated prefix length. If the prefix length has not been included in the AODVv2 message, it is equal to the address length in bits.

Data	Address Block
OrigPrefix/OrigPrefixLen	<address> + <prefix-length>
TargPrefix/TargPrefixLen	<address> + <prefix-length>

9.1.1.4. Address Block TLV Block

Address Block TLVs are always associated with one or more addresses in the Address Block. The following sections show the TLVs that apply to each address.

9.1.4.1. Address Block TLVs for OrigPrefix

Data	TLV Type	Extension Type	Value
None	ADDRESS_TYPE	0	ORIGPREFIX
OrigSeqNum	SEQ_NUM	0	Sequence number of RREQ_Gen, the router which initiated route discovery.
OrigMetric /MetricType	PATH_METRIC	MetricType	Metric value for the route to OrigPrefix, using MetricType.

9.1.4.2. Address Block TLVs for TargPrefix

Data	TLV Type	Extension Type	Value
None	ADDRESS_TYPE	0	TARGPREFIX
TargSeqNum	SEQ_NUM	0	The last known TargSeqNum for TargPrefix.

9.2. Route Reply Message Representation

9.2.1. Message Header

Data	Header Field	Value
None	<msg-type>	RREP
msg_hop_limit	<msg-hop-limit>	MAX_HOPCOUNT - msg_hop_limit from the corresponding RREQ, reduced by number of hops traversed so far by the message.

9.2.2. Message TLV Block

AODVv2 does not define any Message TLVs for an RREP message.

9.2.3. Address Block

An RREP contains OrigPrefix and TargPrefix, and each of these addresses has an associated prefix length. If the prefix length has not been included in the AODVv2 message, it is equal to the address length in bits.

Data	Address Block
OrigPrefix/OrigPrefixLen	<address> + <prefix-length>
TargPrefix/TargPrefixLen	<address> + <prefix-length>

9.2.4. Address Block TLV Block

Address Block TLVs are always associated with one or more addresses in the Address Block. The following sections show the TLVs that apply to each address.

9.2.4.1. Address Block TLVs for OrigPrefix

Data	TLV Type	Extension Type	Value
None	ADDRESS_TYPE	0	ORIGPREFIX

9.2.4.2. Address Block TLVs for TargPrefix

Data	TLV Type	Extension Type	Value
None	ADDRESS_TYPE	0	TARGPREFIX
TargSeqNum	SEQ_NUM	0	Sequence number of RREP_Gen, the router which created the RREP.
TargMetric /MetricType	PATH_METRIC	MetricType	Metric value for the route to TargPrefix, using MetricType.

9.3. Route Reply Acknowledgement Message Representation

9.3.1. Message Header

Data	Header Field	Value
None	<msg-type>	RREP_Ack

9.3.2. Message TLV Block

AODVv2 defines an AckReq Message TLV, included when an acknowledgement of this message is required, in order to monitor adjacency, as described in Section 7.2.

Data	TLV Type	Extension Type	Value
AckReq	ACK_REQ	0	None

9.3.3. Address Block

AODVv2 does not define an Address Block for an RREP_Ack message.

9.3.4. Address Block TLV Block

AODVv2 does not define any Address Block TLVs for an RREP_Ack message.

9.4. Route Error Message Representation

Route Error Messages MAY be split into multiple [RFC5444] messages when the desired contents would exceed the MTU. However, all of the resulting messages MUST have the same message header as described below. If PktSource is included in the AODVv2 message, it MUST be included in all of the resulting [RFC5444] messages.

9.4.1. Message Header

Data	Header Field	Value
None	<msg-type>	RERR

9.4.2. Message TLV Block

AODVv2 does not define any Message TLVs for an RERR message.

9.4.3. Address Block

The Address Block in an RERR MAY contain PktSource, the source address of the IP packet triggering RERR generation, as detailed in Section 8.4. The prefix length associated with PktSource is equal to the address length in bits.

Address Block always contains one address per route that is no longer valid, and each address has an associated prefix length. If a prefix length has not been included for this address, it is equal to the address length in bits.

Data	Address Block
PktSource	<address> + <prefix-length> for PktSource
AddressList/PrefixLengthList	<address> + <prefix-length> for each unreachable address in AddressList

9.4.4. Address Block TLV Block

Address Block TLVs are always associated with one or more addresses in the Address Block. The following sections show the TLVs that apply to each type of address in the RERR.

9.4.4.1. Address Block TLVs for PktSource

Data	TLV Type	Extension Type	Value
PktSource	ADDRESS_TYPE	0	PKTSOURCE

9.4.4.2. Address Block TLVs for Unreachable Addresses

Data	TLV Type	Extension Type	Value
None	ADDRESS_TYPE	0	UNREACHABLE
SeqNumList	SEQ_NUM	0	Sequence number associated with invalid route to the unreachable address.
MetricTypeList	PATH_METRIC	MetricType	None. Extension Type set to MetricType of the route to the unreachable address.

10. Simple External Network Attachment

Figure 5 shows a stub (i.e., non-transit) network of AODVv2 routers which is attached to an external network via a single External Network Access Router (ENAR). The interface to the external network MUST NOT be configured in the InterfaceSet.

As in any externally-attached network, AODVv2 routers and Router Clients that wish to be reachable from the external network MUST have IP addresses within the ENAR's routable and topologically correct prefix (e.g., 191.0.2.0/24 in Figure 5). This AODVv2 network and networks attached to routers within it will be advertised to the external network using procedures which are out of scope for this specification.

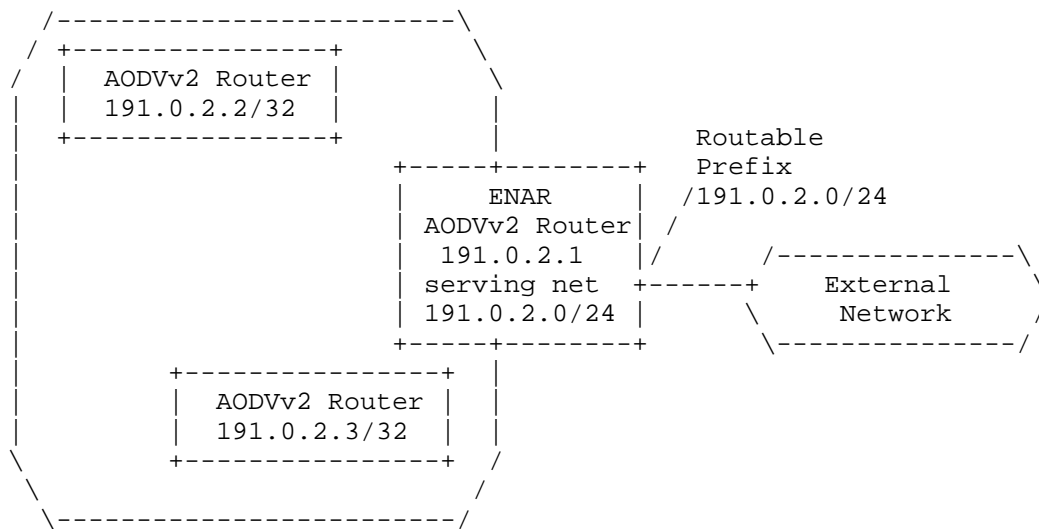


Figure 5: Simple External Network Attachment Example

When an AODVv2 router within the AODVv2 MANET wants to discover a route toward an address on the external network, it uses the normal AODVv2 route discovery for that IP Destination Address. The ENAR MUST respond to RREQ on behalf of all external network destinations, e.g., destinations not on the configured 191.0.2.0/24 network. The ENAR MAY respond with a TargPrefix and TargPrefixLen that represent a prefix including more addresses than just TargAddr, but MUST NOT respond with a TargPrefix and TargPrefixLen which includes any of the networks configured as part of the AODVv2 network. This does result in some inefficiencies in the way external routes are discovered. Sending a Route Request for a gateway is not currently supported.

RREQs for addresses inside the AODVv2 network, e.g. destinations on the configured 191.0.2.0/24 network, are handled using the standard processes described in Section 8. Note that AODVv2 does not support RREQs for prefixes that do not equal address length, but RREPs do advertise the prefix on which TargAddr resides.

When an IP packet from an address on the external network destined for an address in the AODVv2 MANET reaches the ENAR, if the ENAR does not have a route toward that destination in its Routing Information Base, it will perform normal AODVv2 route discovery for that destination.

Configuring the ENAR as a default router is outside the scope of this specification.

11. Configuration

AODVv2 uses various parameters which can be grouped into the following categories:

- o Timers
- o Protocol constants
- o Administrative parameters and controls

This section show the parameters along with their definitions and default values (if any).

Note that several fields have limited size (bits or bytes). These sizes and their encoding may place specific limitations on the values that can be set.

11.1. Timers

AODVv2 requires certain timing information to be associated with Local Route Set entries and message replies. The default values are as follows:

Name	Default Value
ACTIVE_INTERVAL	5 second
MAX_IDLETIME	200 seconds
MAX_BLACKLIST_TIME	200 seconds
MAX_SEQNUM_LIFETIME	300 seconds
RERR_TIMEOUT	3 seconds
RteMsg_ENTRY_TIME	12 seconds
RREQ_WAIT_TIME	2 seconds
RREP_Ack_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. Ideally, for networks with frequent topology changes the AODVv2 parameters SHOULD be adjusted using experimentally determined values or dynamic adaptation. For example, in networks with infrequent topology changes MAX_IDLETIME MAY be set to a much larger value. If the

values were configured differently, the following consequences may be observed:

- o If `MAX_SEQNUM_LIFETIME` was configured differently across the network, and any of the routers lost their sequence number or rebooted, this could result in their next route messages being classified as stale at any AODVv2 router using a greater value for `MAX_SEQNUM_LIFETIME`. This would delay route discovery from and to the re-initializing router.
- o Routers with lower values for `ACTIVE_INTERVAL + MAX_IDLETIME` will invalidate routes more quickly and free resources used to maintain them. This can affect bursty traffic flows which have quiet periods longer than `ACTIVE_INTERVAL + MAX_IDLETIME`. A route which has timed out due to perceived inactivity is not reported. When the bursty traffic resumes, it would cause a RERR to be generated, and the traffic itself would be dropped. This route would be removed from all upstream routers, even if those upstream routers had larger `ACTIVE_INTERVAL` or `MAX_IDLETIME` values. A new route discovery would be required to re-establish the route, causing extra routing protocol traffic and disturbance to the bursty traffic.
- o Routers with lower values for `MAX_BLACKLIST_TIME` would allow neighboring routers to participate in route discovery sooner than routers with higher values. This could result in failed route discoveries if un-blacklisted links are still uni-directional. Since RREQs are retried, this would not affect success of route discovery unless this value was so small as to un-blacklist the router before the RREQ is retried. This value need not be consistent across the network since it is used for maintaining a 1-hop blacklist. However it MUST be greater than `RREQ_WAIT_TIME`.
- o Routers with lower values for `RERR_TIMEOUT` may create more RERR messages than routers with higher values. This value should be large enough that a RERR will reach all routers using the route reported within it before the timer expires, so that no further data traffic will arrive, and no duplicated RERR messages will be generated.
- o Routers with lower values for `RteMsg_ENTRY_TIME` may not consider received redundant multicast route messages as redundant, and may forward these messages unnecessarily.
- o Routers with lower values for `RREQ_WAIT_TIME` may send more frequent RREQ messages and wrongly determine that a route does not exist, if the delay in receiving an RREP is greater than this interval.

- o Routers with lower values for RREP_Ack_SENT_TIMEOUT may wrongly determine links to neighbors to be unidirectional if an RREP_Ack is delayed longer than this timeout.
- o Routers with lower values for RREQ_HOLDDOWN_TIME will retry failed route discoveries sooner than routers with higher values. This may be an advantage if the network topology is frequently changing, or may unnecessarily cause more routing protocol traffic.

MAX_SEQNUM_LIFETIME MUST be configured to have the same values for all AODVv2 routers in the network.

11.2. 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 section describing their use.

Name	Default	Description
DISCOVERY_ATTEMPTS_MAX	3	Section 7.6
RREP_RETRIES	2	Section 8.2.1
MAX_METRIC[MetricType]	[TBD]	Section 6
MAX_METRIC[HopCount]	255	Section 6 and Section 8
MAX_HOPCOUNT	20	Limit to number of hops an RREQ or RREP message can traverse
INFINITY_TIME	[TBD]	Maximum expressible clock time (Section 7.7.2)

Table 3: AODVv2 Constants

MAX_HOPCOUNT cannot be larger than 255.

MAX_METRIC[MetricType] MUST always be the maximum expressible metric value of type MetricType. Field lengths associated with metric values are found in Section 11.5.

These protocol constants MUST have the same values for all AODVv2 routers in the ad hoc network. If the values were configured differently, the following consequences may be observed:

- o DISCOVERY_ATTEMPTS_MAX: Routers with higher values are likely to be more successful at finding routes, at the cost of additional control traffic.

- o RREP_RETRIES: Routers with lower values are more likely to blacklist neighbors when there is a temporary fluctuation in link quality.
- o MAX_METRIC[MetricType]: No interoperability problems due to variations on different routers, but routers with lower values may exhibit overly restrictive behavior during route comparisons.
- o MAX_HOPCOUNT: Routers with a value too small would not be able to discover routes to distant addresses.
- o INFINITY_TIME: No interoperability problems due to variations on different routers, but if a lower value is used, route state management may exhibit overly restrictive behavior.

11.3. Local Settings

The following table lists AODVv2 parameters which SHOULD be administratively configured for each router:

Name	Default Value	Description
InterfaceSet		Section 5.1
Router Client Set		Section 5.2
BUFFER_SIZE_PACKETS	2	Section 7.6
BUFFER_SIZE_BYTES	MAX_PACKET_SIZE [TBD]	Section 7.6
CONTROL_TRAFFIC_LIMIT	[TBD - 50 pkts/sec?]	Section 8

Table 4: Configuration for Local Settings

11.4. Network-Wide Settings

The following administrative controls MAY be used to change the operation of the network. The same settings SHOULD be used across the network. Inconsistent settings at different routers in the network will not result in protocol errors, but poor performance may result.

Name	Default	Description
ENABLE_IDLE_IN_RERR	Disabled	Section 8.4.1

Table 5: Configuration for Network-Wide Settings

11.5. MetricType Allocation

The metric types used by AODVv2 are identified according to Table 6. All implementations MUST use these values.

Name of MetricType	Type	Metric Value Size
Unassigned	0	Undefined
Hop Count	1	1 octet
Unallocated	2 - 254	TBD
Reserved	255	Undefined

Table 6: AODVv2 Metric Types

11.6. RFC 5444 Message Type Allocation

This specification defines four Message Types, to be allocated from the Experimental range of the "Message Types" namespace defined in [RFC5444], as specified in Table 7.

Name of Message	Type
Route Request (RREQ)	224
Route Reply (RREP)	225
Route Error (RERR)	226
Route Reply Acknowledgement (RREP_Ack)	227

Table 7: AODVv2 Message Types

If the AODVv2 experiment proves to be successful, types from the 0-223 range can be allocated in the future.

11.7. RFC 5444 Message TLV Types

This specification defines one Message TLV Type, to be allocated from the Message-Type-specific "Message TLV Types" namespace defined in [RFC5444], as specified in Table 8.

Name of TLV	Type	Length (octets)	Reference
ACK_REQ	128 (TBD)	0	Section 7.2

Table 8: AODVv2 Message TLV Types

11.8. RFC 5444 Address Block TLV Type Allocation

This specification defines three Address Block TLV Types, to be allocated from the Message-Type-specific "Address Block TLV Types" namespace defined in [RFC5444], as specified in Table 9.

Name of TLV	Type	Length (octets)	Reference
PATH_METRIC	129 (TBD)	depends on MetricType	Section 8
SEQ_NUM	130 (TBD)	2	Section 8
ADDRESS_TYPE	131 (TBD)	1	Section 9

Table 9: AODVv2 Address Block TLV Types

11.9. ADDRESS_TYPE TLV Values

These values are used in the [RFC5444] Address Type TLV discussed in Section 9. All implementations MUST use these values.

Address Type	Value
ORIGPREFIX	0
TARGPREFIX	1
UNREACHABLE	2
PKTSOURCE	3
UNSPECIFIED	255

Table 10: AODVv2 Address Types

12. IANA Considerations

This document has no IANA actions.

13. Security Considerations

This section describes various security considerations and potential avenues to secure AODVv2 routing. The main objective of the AODVv2 protocol is for each router to communicate reachability information about addresses for which it is responsible, and for routes it has learned from other AODVv2 routers.

Networks using AODVv2 to maintain connectivity and establish routes on demand may be vulnerable to certain well-known types of threats, which will be detailed in the following. Some of the threats described can be mitigated or eliminated. Tools to do so will be described also.

With the exception of metric values, AODVv2 assures the integrity of all RteMsg data end-to-end through the use of ICVs (see Section 13.4.2).

The on-demand nature of AODVv2 route discovery automatically reduces the vulnerability to route disruption. Since control traffic for updating route tables is diminished, there is less opportunity for attack and failure.

13.1. Availability

Threats to AODVv2 which reduce availability are considered below.

13.1.1. Denial of Service

Flooding attacks using RREQ amount to a (BLIND) denial of service for route discovery: By issuing RREQ messages for targets that don't exist, an attacker can flood the network, blocking resources and drowning out legitimate traffic. By triggering the generation of CONTROL_TRAFFIC_LIMIT amount of messages (for example by sending RREQs for many non-existent destinations), an attacker can prevent legitimate messages from being generated. The effect of this attack is dampened by the fact that duplicate RREQ messages are dropped (preventing the network from DDoSing itself). Processing requirements for AODVv2 messages are typically quite small, however AODVv2 routers receiving RREQs do allocate resources in the form of Neighbor Set, Local Route Set and Multicast Route Message Set entries. The attacker can maximize their impact on set growth by changing OrigPrefix or OrigPrefixLen for each RREQ. If a specific node is to be targeted, this attack may be carried out in a

DISTRIBUTED fashion, either by compromising its direct neighbors or by specifying the target's address with TargPrefix and TargPrefixLen. Note that it might be more economical for the attacker to simply jam the medium; an attack which AODVv2 cannot defend itself against.

Mitigation:

- o If AODVv2 routers always verify that the sender of the RERR message is trusted, this threat is reduced. Processing requirements would typically be dominated by calculations to verify integrity. This has the effect of reducing (but by no means eliminating) AODVv2's vulnerability to denial of service attacks.
- o Authentication of senders can prevent unauthenticated routers from launching a Denial of Service attack on another AODVv2 router. However, this does not protect the network if an attacker has access to an already authenticated router.

13.1.2. Malicious RERR messages

RERR messages are designed to cause removal of installed routes. A malicious node could send an RERR message with false information to attempt to get other routers to remove a route to one or more specific destinations, therefore disrupting traffic to the advertised destinations.

Routes will be deleted if an RERR is received, withdrawing a route for which the sender is the receiver's next hop, and when the RERR includes the MetricType of the installed route, and includes either no sequence number for the route, or includes a greater sequence number than the sequence number stored with that route in the receiver's Local Route Set. Routes will also be deleted if a received RERR contains a PktSource address corresponding to a Router Client.

The information necessary to construct a malicious RERR could be learned by eavesdropping, either by listening to AODVv2 messages or by watching data packet flows.

When the RERR is multicast, it can be received by many routers in the ad hoc network, and will be regenerated when processing results in an active route being removed. This threat could have serious impact on applications communicating by way of the sender of the RERR message.

- o The set of routers which use the malicious router as a next hop may be targeted with a malicious RERR with no PktSource address included, if the RERR contains routes for which the malicious router is a next hop from the receiving router. However, since

the sender of the RERR message is either malicious or broken, it is better that it is not used as a next hop for these routes anyway.

- o A single router which does not use the malicious router as part of its route may be targeted with a malicious RERR with a PktSource address included.
- o Replayed RERR messages could be used to disrupt active routes.

Mitigation:

- o Protection against eavesdropping of AODVv2 messages would mitigate this attack to some extent, but eavesdropping of data packets can also be used to deduce the information about which routes could be targeted.
- o Protection against a malicious router becoming part of a route will mitigate the attack where a set of routers are targeted. This will not protect against the attack if a PktSource address is included.
- o By only regenerating RERR messages where active routes are removed, the spread of the malicious RERR is limited.
- o Including sequence numbers in RERR messages offers protection against attacks using replays of these RERR messages.
- o If AODVv2 routers always verify that the sender of the RERR message is trusted, this threat is reduced.

13.1.3. False Confirmation of Link Bidirectionality

Links could be erroneously treated as bidirectional if malicious unsolicited or spoofed RREP messages were to be accepted. This would result in a route being installed which could not in fact be used to forward data to the destination, and may divert data packets away from the intended destination.

There is a window of RREQ_WAIT_TIME after an RREQ is sent, in which any malicious router could send an RREP in response, in order for the link to the malicious router to be deemed as bidirectional.

Mitigation:

- o Ignoring unsolicited RREP and RREP_Ack messages partially mitigates against this threat.

- o If AODVv2 routers always verify that the sender of the RERR message is trusted, this threat is reduced.

13.1.4. Message Deletion

A malicious router could decide not to forward an RREQ or RREP or RERR message. Not forwarding a RERR or RREP message would disrupt route discovery. Not regenerating a RERR message would result in the source of data packets continuing to maintain and use the route, and further RERR messages being generated by the sender of the non-regenerated RERR. A malicious router could intentionally disrupt traffic flows by not allowing the source of data traffic to re-discover a new route when one breaks.

Failing to send an RREP_Ack would also disrupt route establishment, by not allowing the reverse route to be validated. Return traffic which needs that route will prompt a new route discovery, wasting resources and incurring a slight delay but not disrupting the ability for applications to communicate.

Mitigation:

- o None. also note that malicious router would have to wait for a route to break before it could perform this attack.

13.2. Confidentiality

Passive inspection (eavesdropping) of AODVv2 control messages could enable unauthorized devices to gain information about the network topology, since exchanging such information is the main purpose of AODVv2.

Eavesdropping of data traffic could allow a malicious device to obtain information about how data traffic is being routed. With knowledge of source and destination addresses, malicious messages could be constructed to disrupt normal operation.

13.3. Integrity

Integrity of route information can be compromised in the following types of attack:

13.3.1. Message Insertion

Valid route set entries can be replaced or modified by maliciously constructed AODVv2 messages, destroying existing routes and the network's integrity. Any router may pose as another router by sending RREQ, RREP, RREP_Ack and RERR messages in its name.

- o Sending an RREQ message with false information can disrupt traffic to OrigPrefix, if the sequence number attached is not stale compared to any existing information about OrigPrefix. Since RREQ is multicast and likely to be received by all routers in the ad hoc network, this threat could have serious impact on applications communicating with OrigPrefix. The actual threat to disrupt routes to OrigPrefix is reduced by the AODVv2 mechanism of marking RREQ-derived routes as "Unconfirmed" until the link to the next hop is confirmed.
- o Sending an RREP message with false information can disrupt traffic to TargPrefix. Since RREP is unicast, and ignored if a corresponding RREQ was not recently sent, this threat is minimized, and is restricted to receivers along the path from OrigAddr to TargAddr.
- o Sending an RREP_Ack response message with false information can cause the route to an originator address to be erroneously accepted even though the route would contain a unidirectional link and thus not be suitable for most traffic. Since the RREP_Ack response is unicast, and ignored if a RREP_Ack was not sent recently to the sender of this RREP_Ack response, this threat is minimized and is strictly local to the RREP transmitter expecting the acknowledgement. Unsolicited RREP_Acks are ignored.
- o Sending an RERR message with false information is discussed in Section 13.1.2.

Mitigation:

- o If AODVv2 routers always verify that the sender of a message is trusted, this threat is reduced.

13.3.2. Message Modification - Man in the Middle

Any AODVv2 router can forward messages with modified data.

Mitigation:

- o If AODVv2 routers verify the integrity of AODVv2 messages, then the threat of disruption is minimized. A man in the middle with no knowledge of the key used to calculate an integrity check value may modify a message but the message will be rejected when it fails an integrity check.

13.3.3. Replay Attacks

Replaying of RREQ or RREP messages would be of less use to an attacker, since they would be dropped immediately due to their stale sequence number. RERR messages may or may not include sequence numbers and are therefore susceptible to replay attacks. RREP_Ack messages do not include sequence numbers and are therefore susceptible to replay attacks.

Mitigation:

- o Use of timestamps or sequence numbers prevents replay attacks.

13.4. Protection Mechanisms

13.4.1. Confidentiality and Authentication

Encryption MAY be used for AODVv2 messages. If the routers share a packet-level security association, the message data can be encrypted prior to message transmission. The establishment of such security associations is outside the scope of this specification. Encryption will not only protect against unauthorized devices obtaining information about network topology (eavesdropping) but will ensure that only trusted routers participate in routing operations.

13.4.2. Integrity and Trust using ICVs

Cryptographic Integrity Check Values (ICVs) can be used to ensure integrity of received messages, protecting against man in the middle attacks. Further, by using ICVs, only those routers with knowledge of a shared secret key are allowed to participate in routing information exchanges. [RFC7182] defines ICV TLVs for use with [RFC5444].

The data contained in AODVv2 routing protocol messages MUST be verified using Integrity Check Values, to avoid the use of message data if the message has been tampered with.

13.4.3. Replay Protection using Timestamps

Replay attacks MUST be prevented by using timestamps or sequence numbers in messages. [RFC7182] defines a TIMESTAMP TLV for use with [RFC5444].

The data contained in AODVv2 routing protocol messages MUST be protected with a TIMESTAMP value to ensure the protection against replaying of the message. Sequence numbers can be used as timestamps, since they are known to be strictly increasing.

13.4.4. Application to AODVv2

AODVv2 implementations MUST support ICV and TIMESTAMP TLVs, unless the implementation is intended solely for an environment in which security is unnecessary. AODVv2 deployments SHOULD be configured to use these TLVs to secure messages.

Implementations of AODVv2 MUST support ICV TLVs using type-extensions 1 and 2, hash-function HASH_FUNCTION, and cryptographic function CRYPTOGRAPHIC_FUNCTION. An ICV MUST be included with every message. The ICV value MAY be truncated as specified in [RFC7182].

Since the msg-hop-limit and PATH_METRIC values are mutable when included in AODVv2 messages, these values MUST be set to zero before calculating an ICV. This means that these values are not protected end-to-end and are therefore susceptible to manipulation. This form of attack is described in Section 13.3.2.

Implementations of AODVv2 MUST support a TIMESTAMP TLV using type-extension 0. The timestamp used is a sequence number, and therefore the length of the <TIMESTAMP-value> field matches the AODVv2 sequence number defined in Section 5.4. The TIMESTAMP TLV MUST be included in RREP_Ack and RERR messages.

When more than one message is included in an RFC5444 packet, using a single ICV Packet TLV or single TIMESTAMP Packet TLV is more efficient than including ICV and TIMESTAMP Message TLVs in each message created. If the RFC5444 multiplexer is capable of adding the Packet TLVs, it SHOULD be instructed to include the Packet TLVs in packets containing AODVv2 messages. However, if the multiplexer is not capable of adding the Packet TLVs, the TLVs MUST be included as Message TLVs in each AODVv2 message in the packet.

After message generation but before transmission, the ICV and TIMESTAMP TLVs MUST be added according to each message type as detailed in the following sections. The following steps list the procedure to be performed:

1. If the TIMESTAMP is to be included, depending on AODVv2 message type as specified below, add the TIMESTAMP TLV.
 - o When a TIMESTAMP Packet TLV is being added, the Packet TLV Block size field MUST be updated.
 - o When a TIMESTAMP Message TLV is being added, the Message TLV Block size field MUST be updated.

1. The considerations in Section 8 and section 9 of [RFC7182] are followed, removing existing ICV TLVs and adjusting the size and flags fields as appropriate:
 - o When an ICV Packet TLV is being added, existing ICV Packet TLVs MUST be removed and the Packet TLV Block size MUST be updated. If the Packet TLV Block now contains no TLVs, the `phastlv` bit in the `<pkt-flags>` field in the Packet Header MUST be cleared.
 - o When an ICV Message TLV is being added, existing ICV Message TLVs are removed and the Message TLV Block Size MUST be updated.
1. Mutable fields in the message MUST have their mutable values set to zero before calculating the ICV.
 - o If the `msg-hop-limit` field is included in the [RFC5444] message header, `msg-hop-limit` MUST be set to zero before calculating the ICV.
 - o If a `PATH_METRIC` TLV is included, any values present in the TLV MUST be set to zero before calculating the ICV value.
1. Depending on the message type, the ICV is calculated over the appropriate fields (as specified in sections Section 13.4.4.1, Section 13.4.4.2, Section 13.4.4.3 and Section 13.4.4.4) to include the fields `<hash-function>`, `<cryptographic-function>`, `<key-id-length>`, and, if present, `<key-id>` (in that order), followed by the entire packet or message. This value MAY be truncated (as specified in [RFC7182]).
2. Add the ICV TLV, updating size fields as necessary.
3. The changes made in Step 2 and Step 3 are reversed to re-add any existing ICV TLVs, re-adjust the relevant size and flags fields, and set the `msg-hop-limit` and `PATH_METRIC` TLV values.

On message reception, and before message processing, verification of the received message MUST take place:

1. The considerations in Section 8 and Section 9 of [RFC7182] are followed, removing existing ICV TLVs and adjusting the size and flags fields as appropriate.
 - o When verifying the ICV value in an ICV Packet TLV, all ICV Packet TLVs present in the Packet TLV Block MUST be removed before calculating the ICV, and the Packet TLV Block size MUST be updated. If there are no remaining Packet TLVs, the Packet TLV

Block MUST be removed and the phastlv bit in the <pkt-flags> field MUST be cleared.

- o When verifying the ICV value in an ICV Message TLV, all ICV Message TLVs present in the Message TLV Block MUST be removed before calculating the ICV, and the Message TLV Block size MUST be updated.
- 1. Mutable fields in the message MUST have their mutable values set to zero before calculating the ICV.
- o If the msg-hop-limit field is included in the [RFC5444] message header, msg-hop-limit MUST be set to zero before calculating the ICV.
- o If a PATH_METRIC TLV is included, any values present in the TLV MUST be set to zero before calculating the ICV value.
- 1. The ICV is calculated following the considerations in Section 12.2 of [RFC7182], to include the fields <hash-function>, <cryptographic-function>, <key-id-length>, and, if present, <key-id> (in that order), followed by the entire packet or message.
- o If the received ICV value is truncated, the calculated ICV value MUST also be truncated (as specified in [RFC7182]), before comparing.
- o If the ICV value calculated from the received message or packet does not match the value of <ICV-data> in the received message or packet, the validation fails and the AODVv2 message MUST be discarded and NOT processed or forwarded.
- o If the ICV values do match, the values set to zero before calculating the ICV are reset to the received values, and processing continues to Step 4.
- 1. Verification of a received TIMESTAMP value MUST be performed. The procedure depends on message type as specified in the following sub sections.
- o If the TIMESTAMP value in the received message is not valid, the AODVv2 message MUST be discarded and NOT processed or forwarded.
- o If the TIMESTAMP value is valid, processing continues as defined in Section 7.

13.4.4.1. RREQ Generation and Reception

Since OrigPrefix is included in the RREQ, the ICV can be calculated and verified using the [RFC5444] contents. The ICV TLV has type extension := 1. Inclusion of an ICV TLV message integrity and endpoint authentication, because trusted routers MUST hold the shared key in order to calculate the ICV value, both to include when creating a message, and to validate the message by checking that the ICV is correct.

Since RREQ_Gen's sequence number is incremented for each new RREQ, replay protection is already afforded and no extra TIMESTAMP TLV is required.

After message generation and before message transmission:

1. Add the ICV TLV as described above.

On message reception and before message processing:

1. Verify the received ICV value as described above.
2. Verification of the sequence number is handled according to Section 7.

13.4.4.2. RREP Generation and Reception

Since TargPrefix is included in the RREP, the ICV can be calculated and verified using the [RFC5444] contents. The ICV TLV has type extension := 1. Inclusion of an ICV provides message integrity and endpoint authentication, because trusted routers MUST hold a valid key in order to calculate the ICV value, both to include when creating a message, and to validate the message by checking that the ICV is correct.

Since RREP_Gen's sequence number is incremented for each new RREP, replay protection is already afforded and no extra TIMESTAMP TLV is required.

After message generation and before message transmission:

1. Add the ICV TLV as described above.

On message reception and before message processing:

1. Verify the received ICV value as described above.

2. Verification of the sequence number is handled according to Section 7.

13.4.4.3. RREP_Ack Generation and Reception

Since no sequence number is included in the RREP_Ack, a TIMESTAMP TLV MUST be included to protect against replay attacks. The value in the TIMESTAMP TLV is set as follows:

- o For RREP_Ack request, use Neighbor.AckSeqNum.
- o For RREP_Ack response, use the sequence number from the TIMESTAMP TLV in the received RREP_Ack request.

Since no addresses are included in the RREP_Ack, and the receiver of the RREP_Ack uses the source IP address of a received RREP_Ack to identify the sender, the ICV MUST be calculated using the message contents and the IP source address. The ICV TLV has type extension := 2 in order to accomplish this. This provides message integrity and endpoint authentication, because trusted routers MUST hold the correct key in order to calculate the ICV value.

After message generation and before message transmission:

1. Add the TIMESTAMP TLV and ICV TLV as described above.

On message reception and before message processing:

1. Verify the received ICV value as described above.
2. Verify the received TIMESTAMP value by comparing the sequence number in the value field of the TIMESTAMP TLV as follows:
 - o For a received RREP_Ack request, there is no need to verify the timestamp value. Proceed to message processing as defined in Section 7.
 - o For a received RREP_Ack response, compare with the Neighbor.AckSeqNum of the Neighbor Set entry for sender of the RREP_Ack request.
 - o If the sequence number does not match, the AODVv2 message MUST be discarded. Otherwise, Neighbor.AckSeqNum is incremented by 1 and processing continues according to Section 7.

13.4.4.4. RERR Generation and Reception

Since the sender's sequence number is not contained in the RERR, a `TIMESTAMP` TLV **MUST** be included to protect against replay attacks. The value in the `TIMESTAMP` TLV is set by incrementing and using `RERR_Gen`'s sequence number.

Since the receiver of the RERR **MUST** use the source IP address of the RERR to identify the sender, the ICV **MUST** be calculated using the message contents and the IP source address. The ICV TLV has type extension := 2 in order to accomplish this. This provides message integrity and endpoint authentication, because trusted routers **MUST** hold the shared key in order to calculate the ICV value.

After message generation and before message transmission:

1. Add the `TIMESTAMP` TLV and ICV TLV as described above.

On message reception and before message processing:

1. Verify the received ICV value as described above.
2. Verify the received `TIMESTAMP` value by comparing the sequence number in the value field of the `TIMESTAMP` TLV with the `Neighbor.HeardRERRSeqNum`. If the sequence number in the message is lower than the stored value, the AODVv2 message **MUST** be discarded. Otherwise, the `Neighbor.HeardRERRSeqNum` **MUST** be set to the received value and processing continues according to Section 7.

13.5. Key Management

The method of distribution of shared secret keys is out of the scope of this protocol. Key management is not specified for the following reasons:

Against [RFC4107], an analysis as to whether automated or manual key management should be used shows a compelling case for automated management. In particular:

- o a potentially large number of routers may have to be managed, belonging to several organisations, for example in vehicular applications.
- o a stream cipher is likely to be used, such as an AES variant.

- o long term session keys might be used by more than two parties, including multicast operations. AODVv2 makes extensive use of multicast.
- o there may be frequent turnover of devices.

On reviewing the case for manual key management against the same document, it can be seen that manual management might be advantageous in environments with limited bandwidth or high round trip times. AODVv2 lends itself to sparse ad hoc networks where transmission conditions may indeed be limited, depending on the bearers selected for use.

However, [RFC4107] assumes that the connectivity between endpoints is already available. In AODVv2, no route is available to a given destination until a router client requests that user traffic be transmitted. It is required to secure the signalling path of the routing protocol that will establish the path across which key exchange functions might subsequently be applied, which is clearly the reverse of the expected functionality. A different strategy is therefore required.

There are two possible solutions. In each case, it is assumed that a defence in depth security posture is being adopted by the system integrator, such that each function in the network as a whole is appropriately secured or defended as necessary, and that there is not complete reliance on security mechanisms built in to AODVv2. Such additional mechanisms could include a suitable wireless device security technology, so that wireless devices are authenticated and secured by their peers prior to exchanging user data, which in this case would include AODVV2 signalling traffic as a payload, and mechanisms which verify the authenticity and/or integrity of application-layer user data transported once a route has been established.

1. In the case that no AODVv2 routers have any detailed prior knowledge of any other AODVv2 router, but does have knowledge of the credentials of other organisations in which the router has been previously configured to trust, it is possible for an AODVv2 router to send an initialisation vector as part of an exchange, which could be verified against such credentials. Such an exchange could make use of Identity-Based Signatures ([I-D.ietf-manet-ibs]), based on Elliptic Curve-Based Certificateless Signatures for Identity-Based Encryption [RFC6507], which eliminate the need for a handshake process to establish trust.

2. If it is impossible to use Identity-Based Signatures, and the risk to the AODVv2 signalling traffic is considered to be low due to the use of security countermeasures elsewhere in the system, a simple pre-placed shared secret could be used between routers, which is used as-is or is used to generate some ephemeral secret based on another known variable, such as time of day if that is universally available at a level of accuracy sufficient to make such a system viable.

14. 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 and Ian Chakeres for their long time authorship of AODV. Additional thanks to Derek Atkins, Emmanuel Baccelli, Abdussalam Baryun, Ramon Caceres, Justin Dean, Christopher Dearlove, Fatemeh Ghassemi, Ulrich Herberg, Henner Jakob, Ramtin Khosravi, Luke Klein-Berndt, Lars Kristensen, Tronje Krop, Koojana Kuladinithi, Kedar Namjoshi, Keyur Patel, Alexandru Petrescu, Henning Rogge, Fransisco Ros, Pedro Ruiz, Christoph Sommer, Romain Thouvenin, Richard Trefler, Jiazi Yi, Seung Yi, Behnaz Yousefi, and Cong Yuan, for their reviews of AODVv2 and DYMO, as well as numerous specification suggestions.

15. References

15.1. 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>>.
- [RFC3561] Perkins, C., Belding-Royer, E., and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing", RFC 3561, DOI 10.17487/RFC3561, July 2003, <<http://www.rfc-editor.org/info/rfc3561>>.
- [RFC5444] Clausen, T., Dearlove, C., Dean, J., and C. Adjih, "Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format", RFC 5444, DOI 10.17487/RFC5444, February 2009, <<http://www.rfc-editor.org/info/rfc5444>>.
- [RFC5498] Chakeres, I., "IANA Allocations for Mobile Ad Hoc Network (MANET) Protocols", RFC 5498, DOI 10.17487/RFC5498, March 2009, <<http://www.rfc-editor.org/info/rfc5498>>.

- [RFC7182] Herberg, U., Clausen, T., and C. Dearlove, "Integrity Check Value and Timestamp TLV Definitions for Mobile Ad Hoc Networks (MANETs)", RFC 7182, DOI 10.17487/RFC7182, April 2014, <<http://www.rfc-editor.org/info/rfc7182>>.

15.2. Informative References

- [I-D.ietf-manet-ibs]
Dearlove, C., "Identity-Based Signatures for MANET Routing Protocols", draft-ietf-manet-ibs-05 (work in progress), March 2016.
- [Koodli01]
Koodli, R. and C. Perkins, "Fast handovers and context transfers in mobile networks", Proceedings of the ACM SIGCOMM Computer Communication Review 2001, Volume 31 Issue 5, 37-47, October 2001.
- [Perkins94]
Perkins, C. and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers", Proceedings of the ACM SIGCOMM '94 Conference on Communications Architectures, Protocols and Applications, London, UK, pp. 234-244, August 1994.
- [RFC2501] Corson, S. and J. Macker, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations", RFC 2501, DOI 10.17487/RFC2501, January 1999, <<http://www.rfc-editor.org/info/rfc2501>>.
- [RFC4107] Bellovin, S. and R. Housley, "Guidelines for Cryptographic Key Management", BCP 107, RFC 4107, DOI 10.17487/RFC4107, June 2005, <<http://www.rfc-editor.org/info/rfc4107>>.
- [RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, DOI 10.17487/RFC4193, October 2005, <<http://www.rfc-editor.org/info/rfc4193>>.
- [RFC4728] Johnson, D., Hu, Y., and D. Maltz, "The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4", RFC 4728, DOI 10.17487/RFC4728, February 2007, <<http://www.rfc-editor.org/info/rfc4728>>.
- [RFC6130] Clausen, T., Dearlove, C., and J. Dean, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)", RFC 6130, DOI 10.17487/RFC6130, April 2011, <<http://www.rfc-editor.org/info/rfc6130>>.

[RFC6507] Groves, M., "Elliptic Curve-Based Certificateless Signatures for Identity-Based Encryption (ECCSI)", RFC 6507, DOI 10.17487/RFC6507, February 2012, <<http://www.rfc-editor.org/info/rfc6507>>.

Appendix A. AODVv2 Draft Updates

This section lists the changes between AODVv2 revisions ...-15.txt and ...-16.txt.

- o Changed 'regeneration' language in favor of 'forwarding'.
- o Reintroduced use of msg-hop-limit in 5444 message header.
- o Use OrigPrefix rather than OrigAddr and TargPrefix rather than TargAddr where appropriate
- o Removed validity time
- o Removed AckReq from RREP messages, use two-way RREP_ack to check for bidirectionality
- o Unicast RREP messages
- o Removed orphaned references
- o Clarified language
- o Improved Sequence Number instructions
- o Changed 'Unknown' terminology to 'Heard'
- o Extended experiment description
- o Added detailed description of which steps to take when calculating and evaluating ICVs, particularly how to zero out the metric value

Authors' Addresses

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

Phone: +1-408-330-4586
Email: charliep@computer.org

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

Email: ratliffstan@gmail.com

John Dowdell
Airbus Defence and Space
Celtic Springs
Newport, Wales NP10 8FZ
United Kingdom

Email: john.dowdell@airbus.com

Lotte Steenbrink
HAW Hamburg, Dept. Informatik
Berliner Tor 7
D-20099 Hamburg
Germany

Email: lotte.steenbrink@haw-hamburg.de

Victoria Mercieca
Airbus Defence and Space
Celtic Springs
Newport, Wales NP10 8FZ
United Kingdom

Email: victoria.mercieca@airbus.com

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.

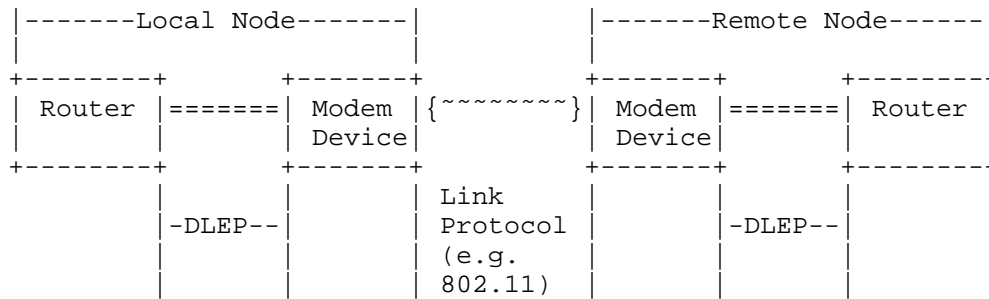


Figure 1: DLEP Network

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.

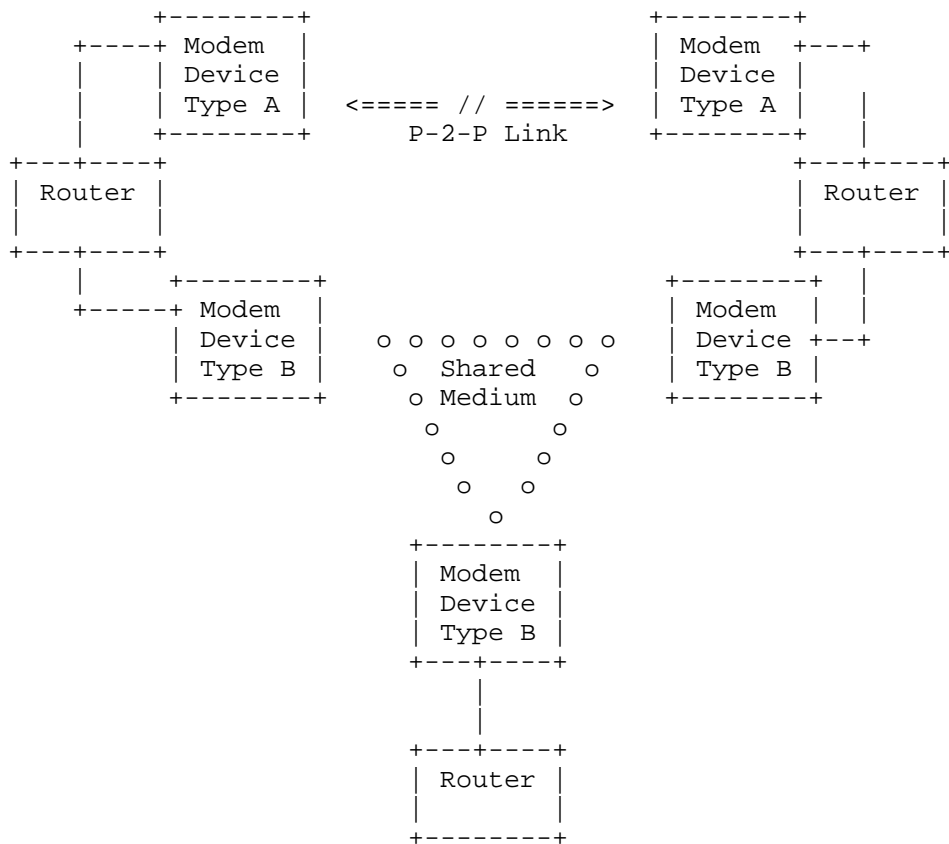


Figure 2: DLEP Network with Multiple Modem Devices

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., datarate, 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 data rate 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 data rate/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:

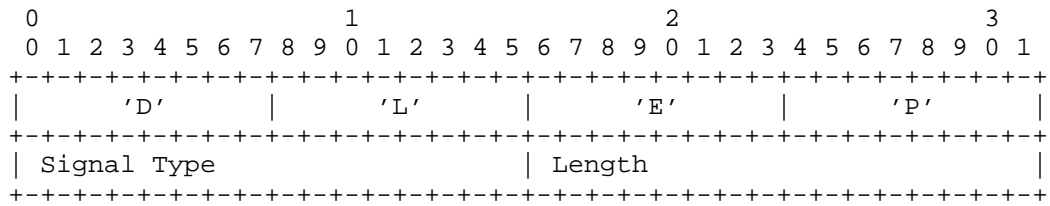


Figure 3: DLEP Signal Header

"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:

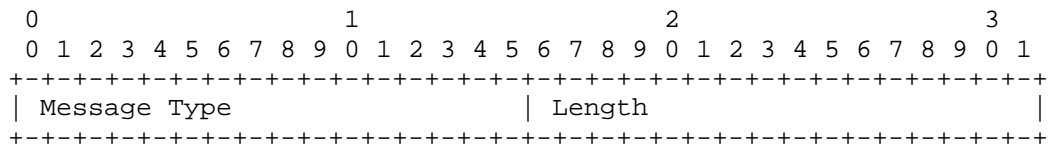


Figure 4: DLEP Message Header

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:

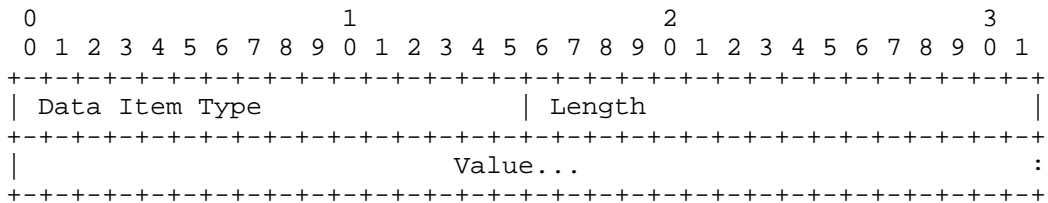


Figure 5: DLEP Generic Data Item

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	2	3	4	5	6	7	8	9
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

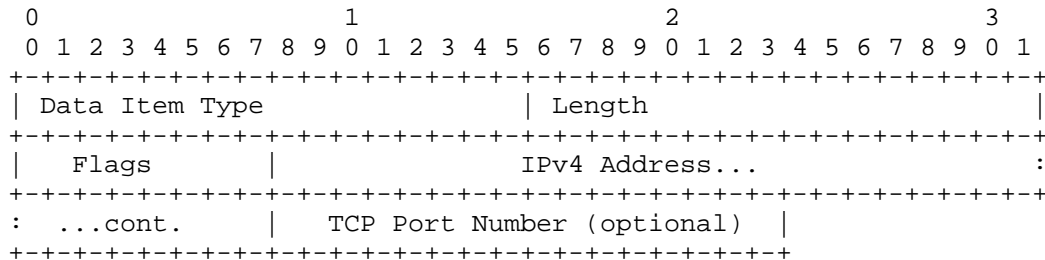
132	Terminate	Timed Out	destination. For example, in the Link Characteristic Response Message (Section 12.19). The session has timed out.
133-239	Terminate	<Reserved>	Reserved for future extensions.
240-254	Terminate	<Private Use>	Available for experiments.
255	Terminate	<Reserved>	Reserved.

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:



Data Item Type: 2

Length: 5 (or 7 if TCP Port included)

Flags: Flags field, defined below.

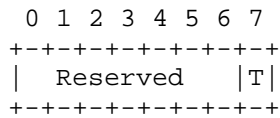
IPv4 Address: The IPv4 address listening on the modem.

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:



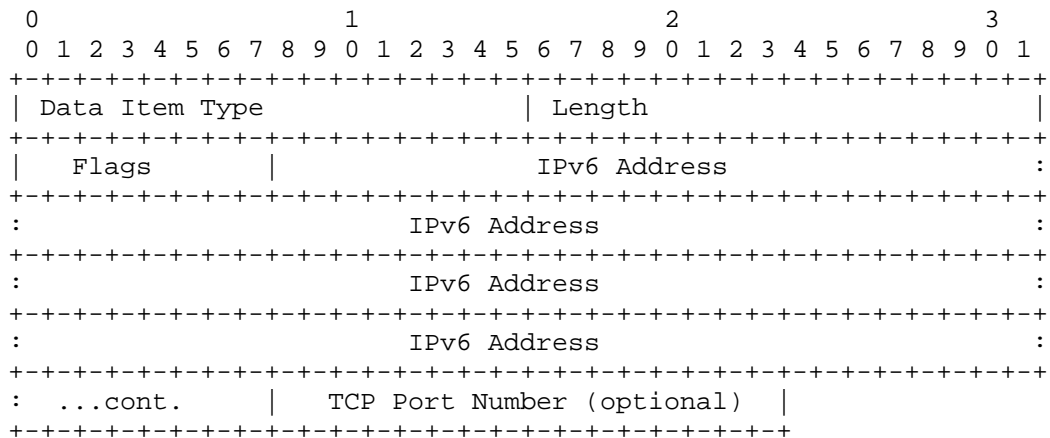
T: Use TLS flag, indicating whether the TCP connection to the given address and port requires the use of TLS [RFC5246] (1), or not (0).

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:



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	2	3	4	5	6	7	8	9
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

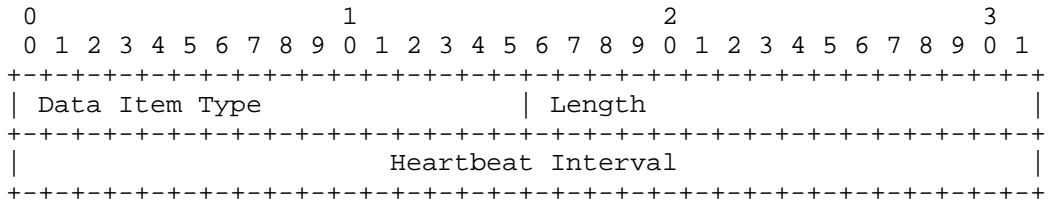
S: Secured Medium flag, used by a modem to indicate if the shared RF medium implements access control (1), or not (0). The Secured Medium flag only has meaning in Signals and Messages sent by a modem.

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:



Data Item Type: 5

Length: 4

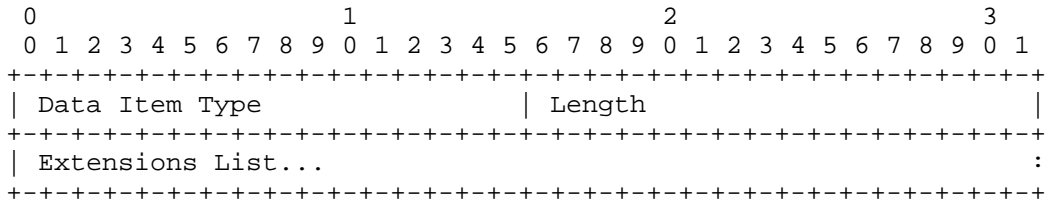
Heartbeat Interval: The interval in milliseconds, expressed as a 32-bit unsigned integer, for Heartbeat Messages. This value MUST NOT be 0.

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:



Data Item Type: 6

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

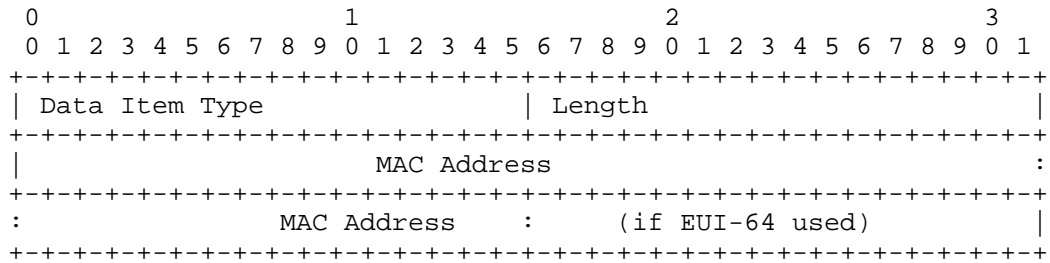
Extension List: A list of extensions supported, identified by their 2-octet value as listed in the extensions registry.

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).



Data Item Type: 7

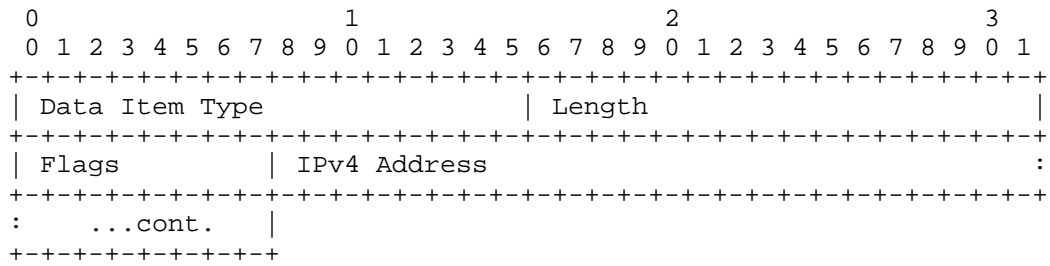
Length: 6 for EUI-48 format, or 8 for EUI-64 format

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:



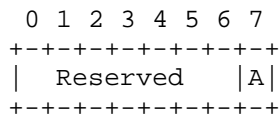
Data Item Type: 8

Length: 5

Flags: Flags field, defined below.

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

The Flags field is defined as:



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.

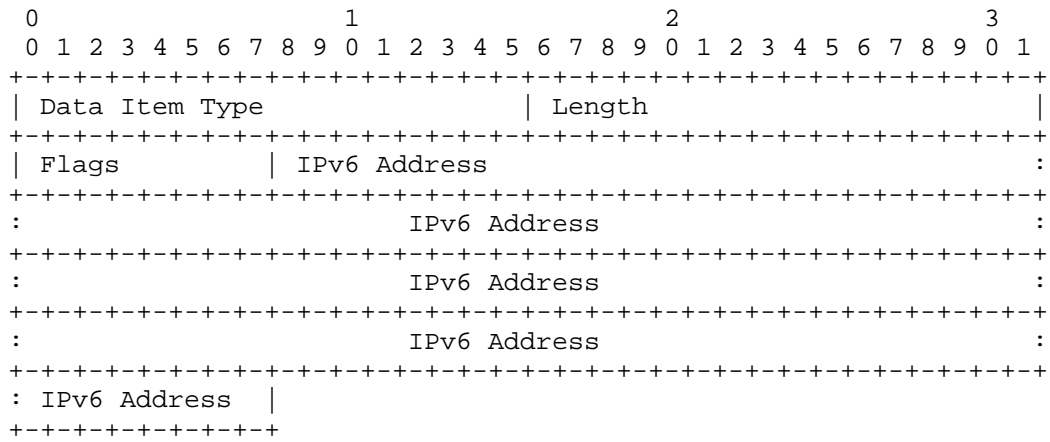
If no response message is appropriate, for example, the Destination Update Message, then the implementation MUST continue with session processing.

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:



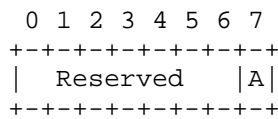
Data Item Type: 9

Length: 17

Flags: Flags field, defined below.

IPv6 Address: IPv6 Address of the destination or peer.

The Flags field is defined as:



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.

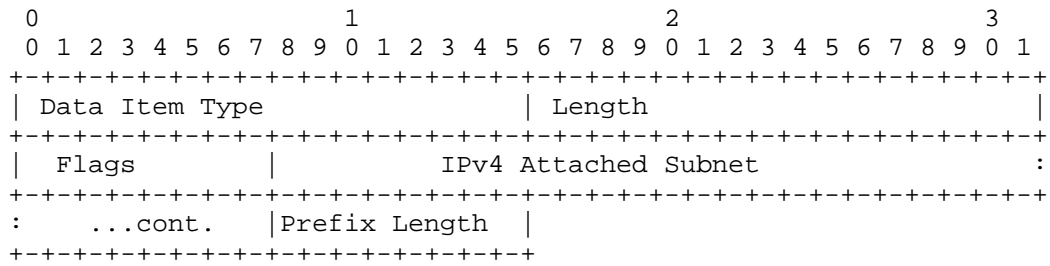
If no response message is appropriate, for example, the Destination Update Message, then the implementation MUST continue with session processing.

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:



Data Item Type: 10

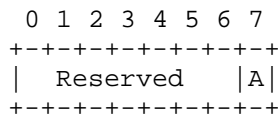
Length: 6

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:



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.

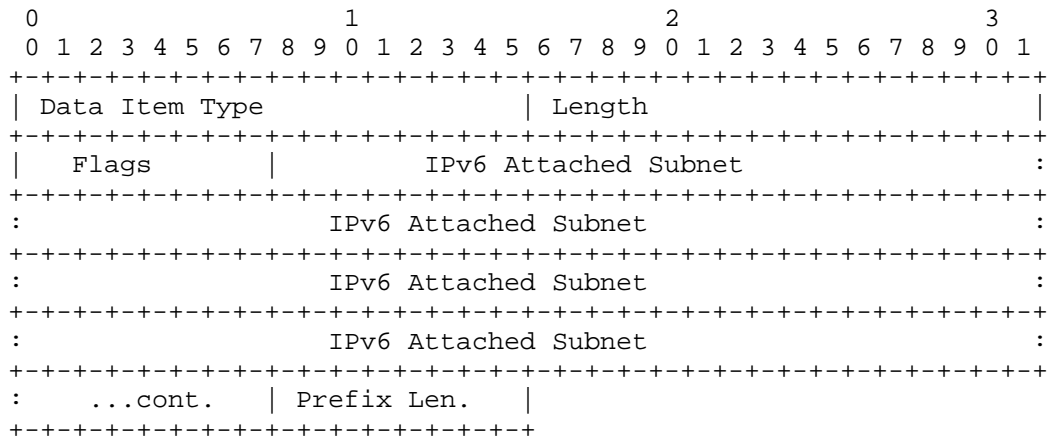
If no response message is appropriate, for example, the Destination Update Message, then the implementation MUST continue with session processing.

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:



Data Item Type: 11

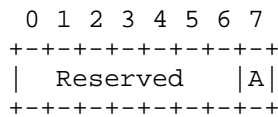
Length: 18

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:



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.

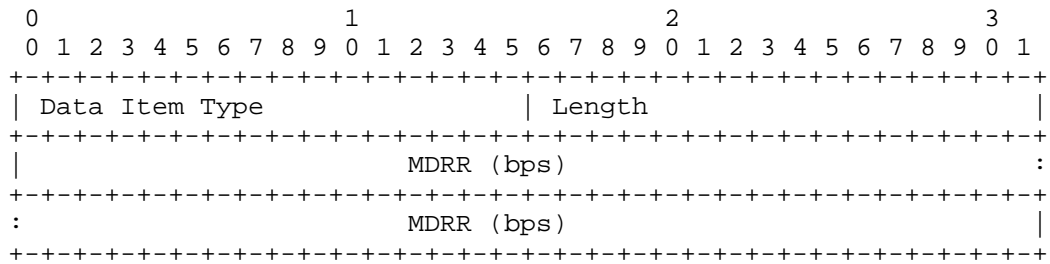
If no response message is appropriate, for example, the Destination Update Message, then the implementation MUST continue with session processing.

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:



Data Item Type: 12

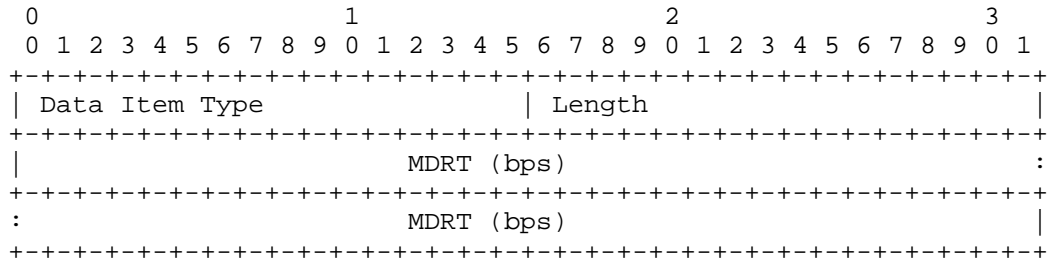
Length: 8

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:



Data Item Type: 13

Length: 8

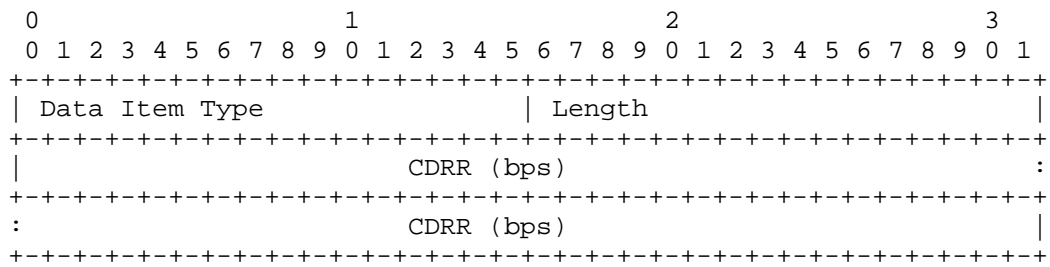
Maximum Data Rate (Transmit): A 64-bit unsigned integer, representing the maximum theoretical data rate, in bits per second (bps), that can be achieved while transmitting on the link.

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:



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.

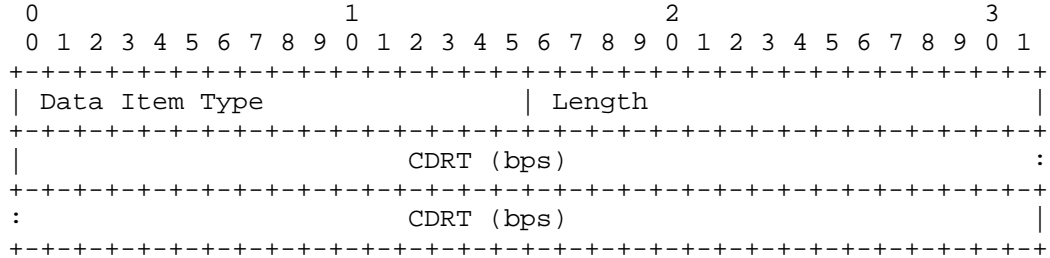
If there is no distinction between Current Data Rate (Receive) and Maximum Data Rate (Receive) (Section 13.12), Current Data Rate (Receive) MUST be set equal to the Maximum Data Rate (Receive). The Current Data Rate (Receive) MUST NOT exceed the Maximum Data Rate (Receive).

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:



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.

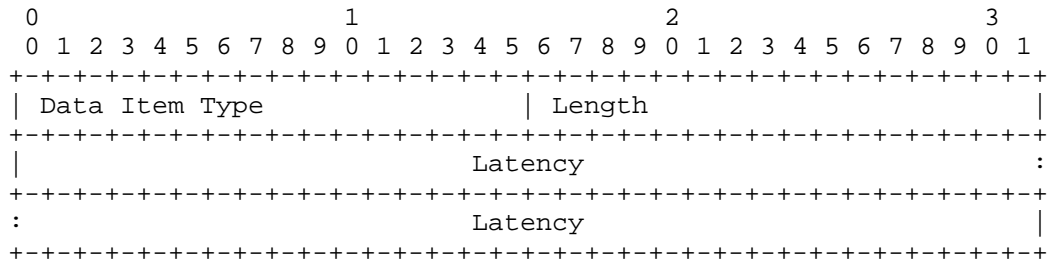
If there is no distinction between Current Data Rate (Transmit) and Maximum Data Rate (Transmit) (Section 13.13), Current Data Rate (Transmit) MUST be set equal to the Maximum Data Rate (Transmit). The Current Data Rate (Transmit) MUST NOT exceed the Maximum Data Rate (Transmit).

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:



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
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.

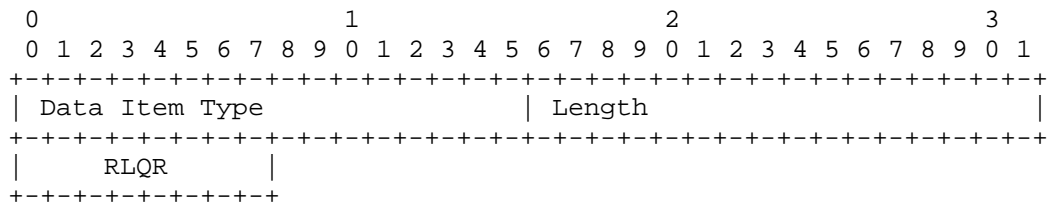
If a device cannot calculate Resources, this Data Item MUST NOT be issued.

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:



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.

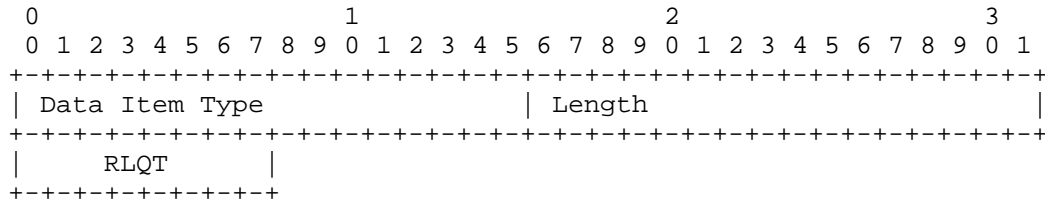
If a device cannot calculate the Relative Link Quality (Receive), this Data Item MUST NOT be issued.

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:



Data Item Type: 19

Length: 1

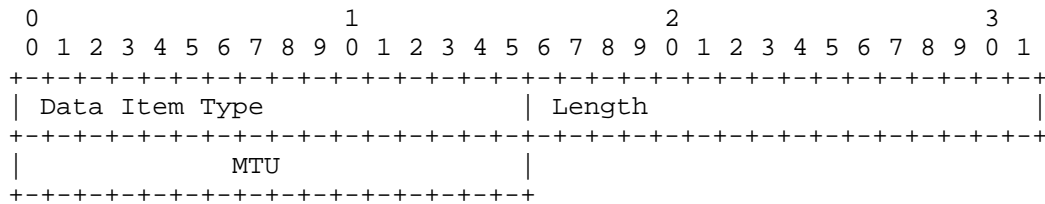
Relative Link Quality (Transmit): A non-dimensional unsigned 8-bit integer, 0-100, representing relative quality of the link for transmitting traffic. Any value greater than 100 MUST be considered as invalid.

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:



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>>.

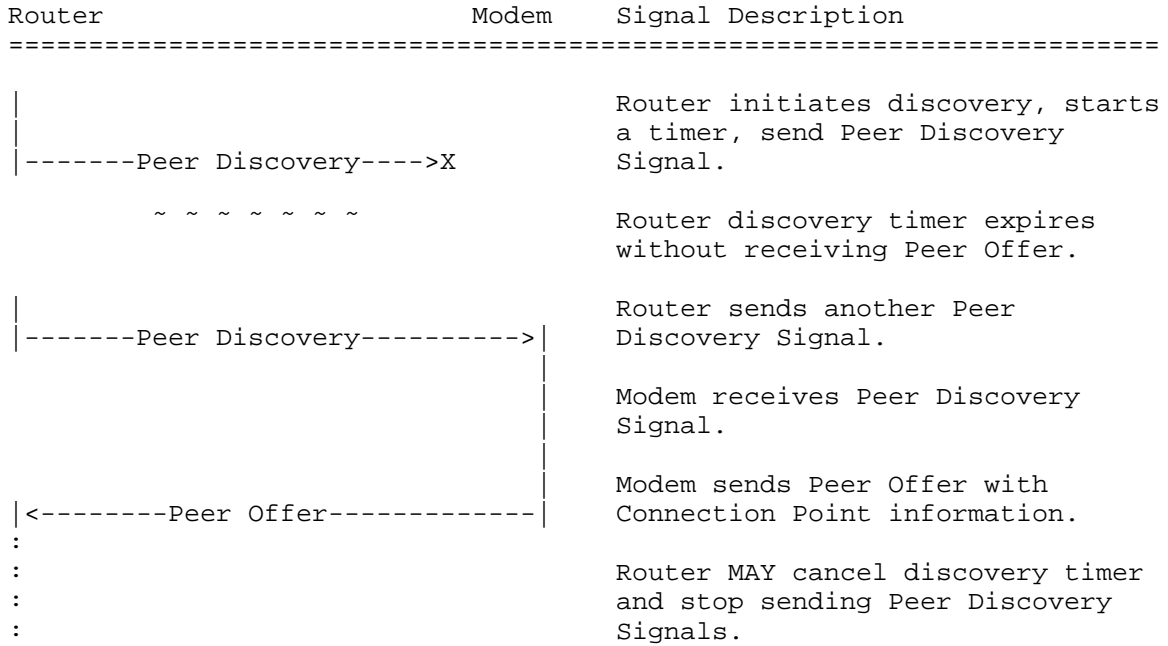
- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <<http://www.rfc-editor.org/info/rfc3629>>.
- [RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., Ed., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", 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



Appendix B. Peer Level Message Flows

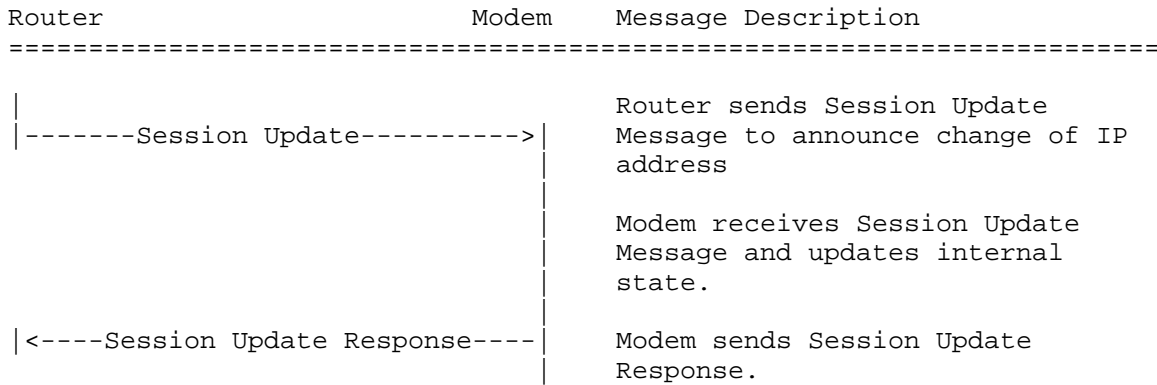
B.1. Session Initialization

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.
<<=====>>		
:	:	

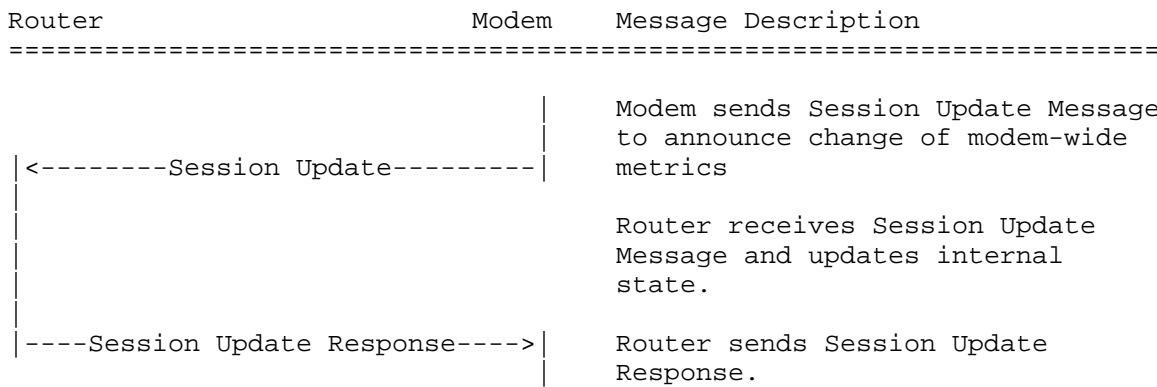
B.2. Session Initialization - Refused

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
		Router sends Session Termination Message with Status Data Item.
-----Session Termination----->		
-----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
		Modem sends Session Termination Message with Status Data Item.
<-----Session Termination-----		
		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.

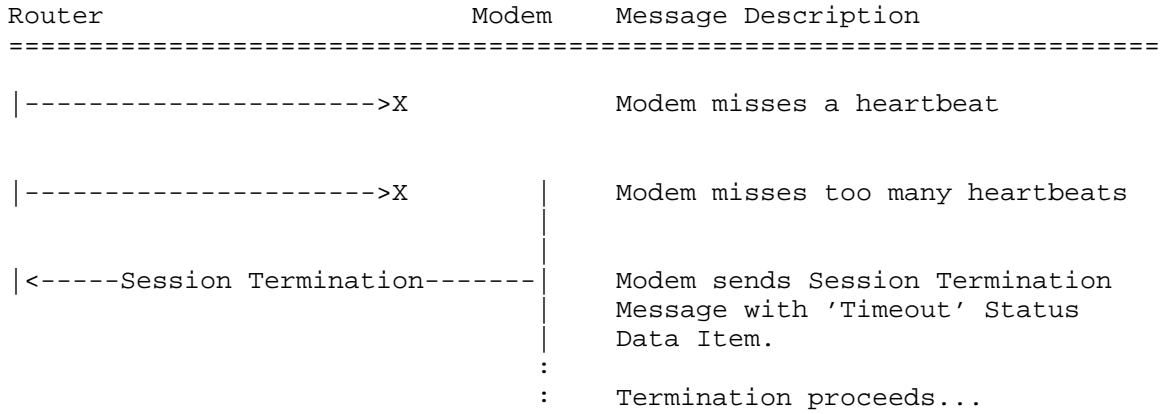
B.7. Session Heartbeats

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



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

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.-----		
		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.
<-----Destination Update-----		
		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.
 --Link Characteristics Request-->		Router requires different Characteristics for the destination, and sends Link Characteristics Request Message.
<---Link Characteristics Resp.--		Modem attempts to adjust link properties to meet the received request, and sends a Link Characteristics Response Message with the new values.

Authors' Addresses

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

Email: sratliff@idirect.net

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

Email: sjury@cisco.com

Darryl Satterwhite
 Broadcom

Email: dsatterw@broadcom.com

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

Email: rick.taylor@airbus.com

Bo Berry

Mobile Ad hoc Networking (MANET)
Internet-Draft
Updates: RFC6130, OLSRv2
(if approved)
Intended status: Standards Track
Expires: September 6, 2014

C. Dearlove
BAE Systems ATC
T. Clausen
LIX, Ecole Polytechnique
March 5, 2014

Optimized Link State Routing Protocol version 2 (OLSRv2) and MANET
Neighborhood Discovery Protocol (NHDP) Extension TLVs
draft-ietf-manet-nhdp-olsrv2-tlv-extension-05

Abstract

This specification describes extensions to definitions of TLVs used by the Optimized Link State Routing Protocol version 2 (OLSRv2) and the MANET Neighborhood Discovery Protocol (NHDP), to increase their abilities to accommodate protocol extensions. This document updates OLSRv2 and RFC6130.

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 6, 2014.

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. Terminology	3
3. Applicability Statement	3
4. TLV Values	4
4.1. Unrecognized TLV Values	4
4.2. TLV Value Lengths	5
4.3. Undefined TLV Values	6
4.3.1. NHDP TLVs: LOCAL_IF, LINK_STATUS and OTHER_NEIGHB	6
4.3.2. OLSRv2 TLVs: MPR and NBR_ADDR_TYPE	6
4.3.3. Unspecified TLV Values	7
5. IANA Considerations	8
5.1. LOCAL_IF Address Block TLVs	8
5.1.1. Create New Registry	8
5.1.2. Modification to Existing Registry	9
5.2. LINK_STATUS Address Block TLVs	10
5.2.1. Create New Registry	10
5.2.2. Modification to Existing Registry	11
5.3. OTHER_NEIGHB Address Block TLVs	12
5.3.1. Create New Registry	12
5.3.2. Modification to Existing Registry	13
5.4. MPR Address Block TLVs	13
5.4.1. Create New Registry	13
5.4.2. Modification to Existing Registry	14
5.5. NBR_ADDR_TYPE Address Block TLVs	15
5.5.1. Create New Registry	15
5.5.2. Modification to Existing Registry	16
6. Security Considerations	16
7. Acknowledgments	17
8. References	17
8.1. Normative References	17
8.2. Informative References	17
Authors' Addresses	18

1. Introduction

The MANET Neighborhood Discovery Protocol (NHDP) [RFC6130] and the Optimized Link State Routing Protocol, version 2 (OLSRv2) [OLSRv2] are protocols for use in mobile ad hoc networks (MANETs) [RFC2501], based on the Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format [RFC5444].

This document updates [RFC6130] and [OLSRv2], specifically their use of TLV (Type-Length-Value) elements, to increase the extensibility of these protocols, and to enable some improvements in their implementation.

This specification reduces the latitude of implementations of [OLSRv2] and [RFC6130] to consider some messages, which will not be created by implementations simply following those specifications, as a reason to consider the message as "badly formed", and thus as a reason to reject the message. This gives greater latitude to the creation of extensions of these protocols, in particular extensions that will interoperate with unextended implementations of those protocols. As part of that, it indicates how TLVs (Type-Length-Value elements) [RFC5444] with unexpected value fields must be handled, and adds some additional options to those TLVs.

Note that TLVs with unknown type or type extension are already specified as to be ignored by [RFC6130] and [OLSRv2], and also are not a reason to reject a message.

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].

Additionally, this document uses the terminology of [RFC5444], [RFC6130], and [OLSRv2].

3. Applicability Statement

This document updates the specification of the protocols [OLSRv2] and [RFC6130].

Specifically, this specification updates [RFC6130] and [OLSRv2] in the following way:

- o Removes the latitude of rejecting a message with a TLV with a known type, but with an unexpected TLV Value field, for the TLV Types defined in [RFC6130] and [OLSRv2].
- o Specifies the handling of a TLV Value field with unexpected length.
- o Sets up IANA registries for TLV Values for the Address Block TLVs:
 - * LOCAL_IF, defined in [RFC6130].
 - * LINK_STATUS, defined in [RFC6130].
 - * OTHER_NEIGHB, defined in [RFC6130].
 - * MPR, defined in [OLSRv2], now considered as a bit field.
 - * NBR_ADDR_TYPE, defined in [OLSRv2], now considered as a bit field.
- o Defines a well-known TLV Value for "UNSPECIFIED" for the Address Block TLV Types LOCAL_IF, LINK_STATUS, and OTHER_NEIGHB, all defined in [RFC6130].

4. TLV Values

NHDP [RFC6130] and OLSRv2 [OLSRv2] define a number of TLVs within the framework of [RFC5444]. These TLVs define the meaning of only some of the contents that can be found in a TLV Value field. This limitation may be either only defining certain TLV Values, or considering only some lengths of the TLV Value fields (or single value field in a multi value Address-Block TLV). This specification describes how NHDP [RFC6130] and OLSRv2 [OLSRv2] are to handle TLVs with other TLV Value fields.

4.1. Unrecognized TLV Values

NHDP and OLSRv2 specify that, in addition to well-defined reasons (in the respective protocol specifications), an implementation of these protocols MAY recognize a message as "badly formed" and therefore "invalid for processing" for other reasons (Section 12.1 of [RFC6130] and Section 16.3.1 of [OLSRv2]). These sections could be interpreted as allowing rejection of a message because a TLV Value field is unrecognized. This specification removes that latitude:

- o An implementation MUST NOT reject a message because it contains an unrecognized TLV value. Instead, any unrecognised TLV Value field

MUST be processed or ignored by an unextended implementation of NHDP or OLSRv2, as described in the following sections.

- o Hence, this specification removes the 7th, 10th, and 11th bullets in Section 12.1 of [RFC6130].

It should be stressed that this is not a change to [RFC6130] or [OLSRv2], except with regard to not allowing this to be a reason for rejection of a message. [RFC6130] or [OLSRv2] are specified in terms such as "if an address is associated with a value of LOST by a LINK_STATUS TLV". Association with an unrecognized value has no effect on any implementation strictly following such a specification.

4.2. TLV Value Lengths

The TLVs specified in [RFC6130] and [OLSRv2] may be either single-value or multi-value TLVs. In either case, the length of each item of information encoded in the TLV Value field is the "single-length", defined and calculated as in section 5.4.1 in [RFC5444]. All TLVs specified in [RFC6130] and [OLSRv2] have a one or two octet single-length. These are considered the expected single-lengths of such a received TLV.

Other single-length TLV Value fields may be introduced by extensions to [RFC6130] and [OLSRv2]. This document specifies how implementations of [RFC6130] and [OLSRv2], or extensions thereof, MUST behave on receiving TLVs of the TLV types defined in [RFC6130] and [OLSRv2], but with TLV Value fields with other single-length values.

The following principles apply:

- o If the received single-length is greater than the expected single-length, then the excess octets MUST be ignored.
- o If the received single-length is less than the expected single-length, then the absent octets MUST be considered to have all bits cleared (0).

Exceptions:

- o A received CONT_SEQ_NUM with a single-length < 2 SHOULD be considered an error.

4.3. Undefined TLV Values

[RFC6130] and [OLSRv2] define a number of TLVs, but for some of these TLVs specify meanings for only some TLV Values. This document establishes IANA registries for these TLV Values, with initial registrations reflecting those used by [RFC6130] and [OLSRv2], and as specified in Section 4.3.3.

There are different cases of TLV Values with different characteristics. These cases are considered in this section.

4.3.1. NHDP TLVs: LOCAL_IF, LINK_STATUS and OTHER_NEIGHB

For the Address-Block TLVs LOCAL_IF, LINK_STATUS and OTHER_NEIGHB TLVs, defined in [RFC6130], only a limited number of values are specified for each. These are converted, by this specification, into extensible registries with initial registrations for values defined and used by [RFC6130] - see Section 5.

An implementation of [RFC6130], receiving a LOCAL_IF, LINK_STATUS, or OTHER_NEIGHB TLV with any TLV Value other than the values which are defined in [RFC6130] MUST ignore that TLV Value, as well as any corresponding attribute association to the address.

4.3.2. OLSRv2 TLVs: MPR and NBR_ADDR_TYPE

The Address-Block TLVs MPR and NBR_ADDR_TYPE, defined in [OLSRv2], are similar to those defined in [RFC6130] in having only limited values specified (1, 2 and 3): 1 and 2, represent presence of two different attributes associated to an address, and 3 represents "both 1 and 2".

These TLV Value fields, are by this specification, converted to bit fields, and MUST be interpreted as such. As the existing definitions of values 1, 2, and 3 behave in that manner, it is likely that this will involve no change to an implementation, but any test of (for example) Value = 1 or Value = 3 MUST be converted to a test of (for example) Value bitand 1 = 1, where "bitand" denotes a bitwise and operation.

This specification creates registries for recording reservations of the individual bits in these bitfields, with initial registrations for values defined and used by [OLSRv2] - see Section 5.

Other TLVs defined by [OLSRv2] are not affected by this specification.

4.3.3. Unspecified TLV Values

The registries defined in Section 5 for the LOCAL_IF, LINK_STATUS and OTHER_NEIGHB TLVs each include an additional TLV Value UNSPECIFIED. This TLV Value represents a defined value that, like currently undefined TLV Values, indicates that no information is associated with this address, but will always have this meaning. Such a TLV Value may be used to enable the creation of more efficient multivalued Address Block TLVs, or to simplify an implementation.

The similar requirement for the MPR and NBR_ADDR_TYPES TLVs is already satisfied by the TLV Value zero, provided that each bit in the TLV Value is defined as set ('1') when indicating the presence of an attribute, or clear ('0') when indicating the absence of an attribute; this is therefore required for registrations from the relevant registries, see Section 5.

For the LINK_METRIC TLV, this is already possible by clearing the most significant bits (0 to 3) of the first octet of the TLV Value. It is RECOMMENDED that in this case the remaining bits of the TLV Value are either all clear ('0') or all set ('1').

5. IANA Considerations

IANA is requested to take a total of ten actions as set out in the following sections.

5.1. LOCAL_IF Address Block TLVs

5.1.1. Create New Registry

IANA maintains a registry called "Mobile Ad hoc NETWORK (MANET) Parameters". IANA is requested to create a new sub-registry called "LOCAL_IF TLV Values".

IANA is requested to populate this registry as specified in Table 1.

Value	Name	Description	Reference
0	THIS_IF	The network address is associated with this local interface of the sending router	[This.I-D]
1	OTHER_IF	The network address is associated with another local interface of the sending router	[This.I-D]
2-223 224-254	UNSPECIFIED	Unallocated: Expert Review Experimental Use	[This.I-D]
255		No information about this network address is provided	[This.I-D]

Table 1: LOCAL_IF TLV Values

New assignments are to be made by Expert Review [RFC5226].

The Designated Experts are required to use the guidelines specified in [RFC6130] and [OLSRv2]. IANA is not expected to record this fact in the registry.

5.1.2. Modification to Existing Registry

IANA maintains a registry called "Mobile Ad hoc NETwork (MANET) Parameters" with a sub-registry called "LOCAL_IF Address Block TLV Type Extensions". This sub-registry currently has an entry for value 0. IANA is requested to replace the entry in the Description column for this value with the text "The value is to be interpreted according to the registry LOCAL_IF TLV Values". The resulting table should look as specified in Table 2.

Type Extension	Description	Reference
0	The value is to be interpreted according to the registry LOCAL_IF TLV Values	[RFC6130] [This.I-D]
1-255	Unassigned	[This.I-D]

Table 2: LOCAL_IF Address Block TLV Type Extensions Modifications

5.2. LINK_STATUS Address Block TLVs

5.2.1. Create New Registry

IANA maintains a registry called "Mobile Ad hoc NETWORK (MANET) Parameters". IANA is requested to create a new sub-registry called "LINK_STATUS TLV Values".

IANA is requested to populate this registry as specified in Table 3.

Value	Name	Description	Reference
0	LOST	The link on this interface from the router with that network address has been lost	[This.I-D]
1	SYMMETRIC	The link on this interface from the router with that network address has the status of symmetric	[This.I-D]
2	HEARD	The link on this interface from the router with that network address has the status of heard	[This.I-D]
3-223 224-254	UNSPECIFIED	Unallocated: Expert Review Experimental Use	[This.I-D]
255		No information about this network address is provided	[This.I-D]

Table 3: LINK_STATUS TLV Values

New assignments are to be made by Expert Review [RFC5226].

The Designated Experts are required to use the guidelines specified in [RFC6130] and [OLSRv2]. IANA is not expected to record this fact in the registry.

5.2.2. Modification to Existing Registry

IANA maintains a registry called "Mobile Ad hoc NETwork (MANET) Parameters" with a sub-registry called "LINK_STATUS Address Block TLV Type Extensions". This sub-registry currently has an entry for value 0. IANA is requested to replace the entry in the Description column for this value with the text "The value is to be interpreted according to the registry LINK_STATUS TLV Values". The resulting table should look as specified in Table 4.

Type Extension	Description	Reference
0	The value is to be interpreted according to the registry LINK_STATUS TLV Values	[RFC6130] [This.I-D]
1-255	Unassigned	[This.I-D]

Table 4: LINK_STATUS Address Block TLV Type Extensions Modifications

5.3. OTHER_NEIGHB Address Block TLVs

5.3.1. Create New Registry

IANA maintains a registry called "Mobile Ad hoc NETWORK (MANET) Parameters". IANA is requested to create a new sub-registry called "OTHER_NEIGHB TLV Values".

IANA is requested to populate this registry as specified in Table 5.

Value	Name	Description	Reference
0	LOST	The neighbor relationship with the router with that network address has been lost	[This.I-D]
1	SYMMETRIC	The neighbor relationship with the router with that network address is symmetric	[This.I-D]
2-223		Unallocated: Expert Review	
224-254		Experimental Use	[This.I-D]
255	UNSPECIFIED	No information about this network address is provided	[This.I-D]

Table 5: OTHER_NEIGHB Address Block TLV Values

New assignments are to be made by Expert Review [RFC5226].

The Designated Experts are required to use the guidelines specified in [RFC6130] and [OLSRv2]. IANA is not expected to record this fact in the registry.

5.3.2. Modification to Existing Registry

IANA maintains a registry called "Mobile Ad hoc NETWORK (MANET) Parameters" with a sub-registry called "OTHER_NEIGHB Address Block TLV Type Extensions". This sub-registry currently has an entry for value 0. IANA is requested to replace the entry in the Description column for this value with the text "The value is to be interpreted according to the registry OTHER_NEIGHB TLV Values". The resulting table should look as specified in Table 6.

Type Extension	Description	Reference
0	The value is to be interpreted according to the registry OTHER_NEIGHB TLV Values	[RFC6130] [This.I-D]
1-255	Unassigned	[This.I-D]

Table 6: OTHER_NEIGHB Address Block TLV Type Extensions Modifications

5.4. MPR Address Block TLVs

5.4.1. Create New Registry

IANA maintains a registry called "Mobile Ad hoc NETWORK (MANET) Parameters". IANA is requested to create a new sub-registry called "MPR TLV Bit Values".

IANA is requested to populate this registry as specified in Table 7.

Bit	Value	Name	Description	Reference
7	0x01	Flooding	The neighbor with that network address has been selected as flooding MPR	[This.I-D]
6	0x02	Routing	The neighbor with that network address has been selected as routing MPR	[This.I-D]
0-5			Unallocated: Expert Review	

Table 7: MPR Address Block TLV Bit Values

New assignments are to be made by Expert Review [RFC5226].

The Designated Experts are required to use the guidelines specified in [RFC6130] and [OLSRv2]. Additionally, the Designated Experts are required to ensure that the following sense is preserved:

- o For each bit in the field, a set bit (1) means that the address has the designated property, while an unset bit (0) means that no information about the designated property is provided. In particular, an unset bit must not be used to convey any specific information about the designated property. IANA is not expected to record these facts in the registry.

5.4.2. Modification to Existing Registry

IANA maintains a registry called "Mobile Ad hoc NETwork (MANET) Parameters" with a sub-registry called "MPR Address Block TLV Type Extensions". This sub-registry currently has an entry for value 0. IANA is requested to replace the entry in the Description column for this value with the text "The value is to be interpreted according to the registry MPR TLV Bit Values". The resulting table should look as specified in Table 8.

Type Extension	Description	Reference
0	The value is to be interpreted according to the registry MPR TLV Bit Values	[OLSRv2] [This.I-D]
1-255	Unassigned	[This.I-D]

Table 8: MPR Address Block TLV Type Extensions Modifications

5.5. NBR_ADDR_TYPE Address Block TLVs

5.5.1. Create New Registry

IANA maintains a registry called "Mobile Ad hoc NETWORK (MANET) Parameters". IANA is requested to create a new sub-registry called "NBR_ADDR_TYPE Address Block TLV Bit Values".

IANA is requested to populate this registry as specified in Table 9.

Bit	Value	Name	Description	Reference
7	0x01	ORIGINATOR	The network address is an originator address reachable via the originating router	[This.I-D]
6	0x02	ROUTABLE	The network address is a routable address reachable via the originating router	[This.I-D]
0-5			Unallocated: Expert Review	

Table 9: NBR_ADDR_TYPE Address Block TLV Bit Values

New assignments are to be made by Expert Review [RFC5226].

The Designated Experts are required to use the guidelines specified in [RFC6130] and [OLSRv2]. Additionally, the Designated Experts are required to ensure that the following sense is preserved:

- o For each bit in the field, a set bit (1) means that the address has the designated property, while an unset bit (0) means that no information about the designated property is provided. In particular, an unset bit must not be used to convey any specific information about the designated property. IANA is not expected to record these facts in the registry.

5.5.2. Modification to Existing Registry

IANA maintains a registry called "Mobile Ad hoc NETwork (MANET) Parameters" with a sub-registry called "NBR_ADDR_TYPE Address Block TLV Type Extensions". This sub-registry currently has an entry for value 0. IANA is requested to replace the entry in the Description column for this value with the text "The value is to be interpreted according to the registry NBR_ADDR_TYPE TLV Bit Values". The resulting table should look as specified in Table 10.

Type Extension	Description	Reference
0	The value is to be interpreted according to the registry NBR_ADDR_TYPE Address Block TLV Bit Values	[OLSRv2] [This.I-D]
1-255	Unassigned	[This.I-D]

Table 10: NBR_ADDR_TYPE Address Block TLV Type Extensions Modifications

6. Security Considerations

The presented updates to [RFC6130] and [OLSRv2]:

- o Create IANA registries for retaining TLV values for TLVs, already defined in the already published specifications of the two protocols, and with initial registrations for the TLV values defined by these specifications. This does not give rise to any additional security considerations.
- o Enable protocol extensions to be able to register TLV values in the created IANA registries. Such extensions MUST specify appropriate security considerations.
- o Create, in some registries, a registration for "UNSPECIFIED" values, for more efficient use of multi-value Address Block TLVs. The interpretation of an address being associated with a TLV of a given type and with the value "UNSPECIFIED" is identical to that address not being associated with a TLV of that type. Thus, this update does not give rise to any additional security considerations.
- o Reduces the latitude of implementations of the two protocols to reject a message as "badly formed", due to the value field of a

TLV being unexpected. These protocols are specified in terms such as "if an address is associated with a value of LOST by a LINK_STATUS TLV". Association with an unknown value (or a value newly defined to mean no link status information) has no effect on such a specification. Thus, this update does not give rise to any additional security considerations.

- o Do not introduce any opportunities for attacks on the protocols through signal modification that are not already present in the two protocols.

7. Acknowledgments

The authors would like to gratefully acknowledge the following people for intense technical discussions, early reviews, and comments on the specification (listed alphabetically): Ulrich Herberg (Fujitsu Laboratories of America) and Henning Rogge (Frauenhofer FKIE).

The authors would also like to express their gratitude to Adrian Farrel, for his assistance and contributions to successful and timely completion of this specification.

8. References

8.1. Normative References

- [OLSRv2] Clausen, T., Dearlove, C., Jacquet, P., and U. Herberg, "The Optimized Link State Routing Protocol version 2", work in progress draft-ietf-manet-olsrv2-19, March 2013.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC5444] Clausen, T., Dearlove, C., Dean, J., 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.

8.2. Informative References

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

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

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
URI: <http://www.baesystems.com/>

Thomas Heide Clausen
LIX, Ecole Polytechnique

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

Mobile Ad hoc Networking (MANET)
Internet-Draft
Updates: 7188, 7631
(if approved)
Intended status: Experimental
Expires: March 31, 2016

C. Dearlove
BAE Systems Applied Intelligence
Laboratories
T. Clausen
LIX, Ecole Polytechnique
September 28, 2015

Multi-Topology Extension for the Optimized Link State Routing Protocol
version 2 (OLSRv2)
draft-ietf-manet-olsrv2-multitopology-07

Abstract

This specification describes an extension to the Optimized Link State Routing Protocol version 2 (OLSRv2) to support multiple routing topologies, while retaining interoperability with OLSRV2 routers that do not implement this extension.

This specification updates RFCs 7188 and 7631 by modifying and extending TLV registries and descriptions.

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 March 31, 2016.

Copyright Notice

Copyright (c) 2015 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
1.1.	Motivation and Experimentation	4
2.	Terminology and Notation	5
3.	Applicability Statement	6
4.	Protocol Overview and Functioning	6
5.	Parameters	8
6.	Information Bases	9
6.1.	Local Attached Network Set	9
6.2.	Link Sets	9
6.3.	2-Hop Sets	9
6.4.	Neighbor Set	9
6.5.	Router Topology Set	10
6.6.	Routable Address Topology Set	10
6.7.	Attached Network Set	10
6.8.	Routing Sets	11
7.	TLVs	11
7.1.	Message TLVs	11
7.1.1.	MPR_TYPES TLV	11
7.1.2.	MPR_WILLING TLV	11
7.2.	Address Block TLVs	12
7.2.1.	LINK_METRIC TLV	12
7.2.2.	MPR TLV	13
7.2.3.	GATEWAY TLV	13
8.	HELLO Messages	14
8.1.	HELLO Message Generation	14
8.2.	HELLO Message Processing	14
9.	TC Messages	15
9.1.	TC Message Generation	15
9.2.	TC Message Processing	16
10.	MPR Calculation	16
11.	Routing Set Calculation	16
12.	Management Considerations	17
13.	IANA Considerations	18
13.1.	Expert Review: Evaluation Guidelines	18
13.2.	Message TLV Types	18
13.3.	Address Block TLV Types	19
14.	Security Considerations	21
15.	Acknowledgments	21
16.	References	22
16.1.	Normative References	22
16.2.	Informative References	22
	Authors' Addresses	23

1. Introduction

The Optimized Link State Routing Protocol, version 2 [RFC7181] (OLSRv2) is a proactive link state routing protocol designed for use in mobile ad hoc networks (MANETs) [RFC2501]. One of the significant improvements of OLSRv2 over its Experimental precursor [RFC3626] is the ability of OLSRv2 to route over other than minimum hop routes, using a link metric.

A limitation that remains in OLSRv2 is that it uses a single link metric type for all routes. However in some MANETs it would be desirable to be able to route packets using more than one link metric type. This specification describes an extension to OLSRv2 that is designed to permit this, while maintaining maximal interoperability with OLSRv2 routers not implementing this extension.

The purpose of OLSRv2 can be described as to create and maintain a Routing Set, which contains all the necessary information to populate an IP routing table. In a similar way, the role of this extension can be described as to create and maintain multiple Routing Sets, one for each link metric type supported by the router maintaining the sets.

1.1. Motivation and Experimentation

Multi-topology routing is a natural extension to a link state routing protocol, as for example to OSPF (see [RFC4915]). However multi-topology routing for OLSRv2 does not yet benefit from extensive operational, or even experimental, experience. This specification is published to facilitate collecting such experience, with the intent that once suitable experimental evidence has been collected, an OLSRv2 Multi-Topology Routing Extension will be proposed for advancement onto Standards Track.

Any experiments using this protocol extension are encouraged. Reports from such experiments planned with pre-specified objectives and scenarios (including link, position and mobility information) are particularly encouraged. Results from such experiments, documenting the following, are of particular importance:

- o Operation in networks that contain both routers implementing this extension, and routers implementing only [RFC7181], in particular are there any unexpected interactions that can break the network?
- o Operation in networks with dynamic topologies, both due to mobility and due to link metric changes for reasons other than mobility.

- o Operation in realistic deployments, and details thereof, including in particular indicating how many concurrent topologies were required.
- o Behavior of routing sets, including measures of successful route establishment.

In addition, reports from experiments covering the following are also of value:

- o Which link metric types were useful, and how the metrics to associate with a given link were established.
- o How packet types were associated with link metric types (whether using DiffServ on an alternative mechanism).
- o Any data link layer issues, and any cross-layer issues, including whether NHDP link quality was used, and how.
- o Transport and higher layer issues observed, if any.
- o Resource requirements observed from running the protocol, including processing, storage, and bandwidth.
- o Network performance, including packet delivery results.
- o Any other implementation issues.

The first bullet in the latter list applies to unextended [RFC7181] as well as this extension, and potentially to other MANET routing protocols. This may also allow experimentation with link metric types that are not compromises to handle multiple traffic types.

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].

This specification uses the terminology of [RFC5444], [RFC6130] and [RFC7181], which is to be interpreted as described in those specifications.

Additionally, this specification uses the following terminology:

Router - A MANET router that implements [RFC7181].

MT-OLSRv2 - The protocol defined in this specification as an extension to [RFC7181].

This specification introduces the notation `map[A -> B]` to represent an associative mapping. The domain of this mapping (A) is, in this specification, always a set of link metric types that the router supports: either `IFACE_METRIC_TYPES` or `ROUTER_METRIC_TYPES`, as defined in Section 5. The codomain of this mapping (B) is a set of all possible values of an appropriate type, in this specification this type is always one of:

- o boolean (true or false),
- o willingness (a 4 bit unsigned integer from 0 to 15);
- o number of hops (an 8 bit unsigned integer from 0 to 255), or
- o link metric (either a representable link metric value, as described in [RFC7181], or `UNKNOWN_METRIC`).

3. Applicability Statement

The protocol described in this specification is applicable to a MANET for which OLSRv2 is otherwise applicable (see [RFC7181], Section 3), but in which multiple topologies are maintained, each characterized by a different choice of link metric type. It is assumed, but outside the scope of this specification, that the network layer is able to choose which topology to use for each packet, for example using the DiffServ Code Point (DSCP) defined in [RFC2474]. This selection of topology **MUST** be consistent, that is each router receiving a packet must make the same choice of link metric type, in order that each packet uses a single topology. This is necessary to avoid the possibility of a packet "looping" in the network.

4. Protocol Overview and Functioning

The purpose of this specification is to extend OLSRv2 [RFC7181] so as to enable a router to establish and maintain multiple routing topologies in a MANET, each topology associated with a link metric type. Routers in the MANET may each form part of some or all of these topologies, and each router will maintain a Routing Set for each topology that it forms part of, allowing separate routing of packets for each topology.

MT-OLSRv2 is designed to interoperate with OLSRv2; a MANET can be created containing both routers that implement MT-OLSRv2 (MT-OLSRv2 routers) and routers that do not implement MT-OLSRv2, and may be unaware of its existence (non-MT-OLSRv2 routers). MANETs may also be created that are known to contain only MT-OLSRv2 routers. In both cases, but especially the former, management may be required to ensure that the MANET will function as required, and does not, for example, unnecessarily fragment. (Such issues already arise in an OLSRv2-based MANET using multiple interfaces.)

OLSRv2 is an extension of NHDP [RFC6130]. However the extension in this specification does not modify NHDP, it only modifies Protocol Sets that are specific to OLSRv2, or elements in Protocol Tuples that were added by OLSRv2 and which are not included in nor used by NHDP. In addition it does not use or modify the link quality mechanism in [RFC6130].

Each router implementing this specification selects a set of link metric types for each of its OLSRv2 interfaces. If all routers in the MANET implement MT-OLSRv2, then there are no restrictions within this specification on how these sets of link metrics are selected. (However the issues described in the preceding paragraph still apply.) However in MANETs containing non-MT-OLSRv2 routers, the single link metric used by these non-MT-OLSRv2 routers must be included in the set of link metrics for each OLSRv2 interface of an MT-OLSRv2 router that may be heard on an OLSRv2 interface of a non-MT-OLSRv2 router in the MANET.

Each router then determines an incoming link metric for each link metric type selected for each of its OLSRv2 interfaces. These link metrics are distributed using link metric TLVs contained in all HELLO messages sent on OLSRv2 interfaces, and in all TC messages. Both HELLO and TC messages generated by an MT-OLSRv2 router (other than one using only the single metric type used by non-MT-OLSRv2 routers) include an MPR_TYPES Message TLV that indicates that this is an MT-OLSRv2 router and which metric types it supports (on the sending OLSRv2 interface for a HELLO message).

In addition to link and neighbor metric values for each link metric type, router MPR (multipoint relay) and MPR selector status, and advertised neighbor status, is maintained per supported neighbor metric type, for each symmetric 1-hop neighbor. Each router may choose a different willingness to be a routing MPR for each link metric type that it supports.

A network using MT-OLSRv2 will usually require greater management than one using unmodified OLSRv2. In particular, the use of multiple metric types across the MANET must be managed, by administrative

configuration or otherwise. As also for other decisions that may be made when using OLSRv2, a bad collective choice of metric type use will make the MANET anywhere from inefficient to non-functional, so care will be needed in selecting supported link metric types across the MANET.

The meanings of link metric types are at the discretion of the MANET operator, they could be used, for example, to represent packets of different types, packets in streams of different rates, or packets with different trust requirements. Note that packets will generally not be delivered to routers that do not support that link metric type, and the MANET, and the packets sent in it, will need to be managed accordingly (especially if containing any non-MT-OLSRv2 routers).

5. Parameters

The parameters used in [RFC7181], including from its normative references, are used in this specification with the following changes.

Each OLSRv2 interface will support a number of link metric types, corresponding to Type Extensions of the LINK_METRIC TLV defined in [RFC7181]. The router parameter LINK_METRIC_TYPE, used by routers that do not implement MT-OLSRv2, and used with that definition in this specification, is replaced in routers implementing MT-OLSRv2 by an interface parameter array IFACE_METRIC_TYPES and a router parameter array ROUTER_METRIC_TYPES. Each element in these arrays is a link metric type (i.e., a type extension used by the LINK_METRIC TLV [RFC7181]).

The interface parameter array IFACE_METRIC_TYPES contains the link metric types supported on that OLSRv2 interface. The router parameter array ROUTER_METRIC_TYPES is the union of all of the IFACE_METRIC_TYPES. Both arrays MUST be without repetitions.

If in a given deployment there may be any routers that do not implement MT-OLSRv2, then IFACE_METRIC_TYPES MUST first include LINK_METRIC_TYPE if that OLSRv2 interface may be able to communicate with any routers that do not implement MT-OLSRv2. In that case, ROUTER_METRIC_TYPES MUST also first include LINK_METRIC_TYPE.

In addition, the router parameter WILL_ROUTING is extended to an array of values, one each for each link metric type in the router parameter list ROUTER_METRIC_TYPES.

6. Information Bases

The Information Bases specified in [RFC7181], which extend those specified in [RFC6130], are further extended in this specification. With the exception of the Routing Set, the extensions in this specification are the replacement of single values (boolean, willingness, number of hops, or link metric) from [RFC7181] with elements representing multiple values (associative mappings from a set of metric types to their corresponding values). The following subsections detail these extensions.

Note that, as in [RFC7181], an implementation is free to organize its internal data in any manner it chooses, it needs only to behave as if it were organized as described in [RFC7181] and this specification.

6.1. Local Attached Network Set

Each element `AL_dist` becomes a map[ROUTER_METRIC_TYPES -> number of hops].

Each element `AL_metric` becomes a map[ROUTER_METRIC_TYPES -> link metric].

6.2. Link Sets

Each element `L_in_metric` becomes a map[IFACE_METRIC_TYPES -> link metric].

Each element `L_out_metric` becomes a map[IFACE_METRIC_TYPES -> link metric].

The elements of `L_in_metric` MUST be set following the same rules that apply to the setting of the single element `L_in_metric` in [RFC7181].

6.3. 2-Hop Sets

Each element `N2_in_metric` becomes a map[ROUTER_METRIC_TYPES -> link metric].

Each element `N2_out_metric` becomes a map[ROUTER_METRIC_TYPES -> link metric].

6.4. Neighbor Set

Each element `N_in_metric` becomes a map[ROUTER_METRIC_TYPES -> link metric].

Each element `N_out_metric` becomes a map[ROUTER_METRIC_TYPES -> link

metric].

Each element `N_will_routing` becomes a map[ROUTER_METRIC_TYPES -> willingness].

Each element `N_routing_mpr` becomes a map[ROUTER_METRIC_TYPES -> boolean].

Each element `N_mpr_selector` becomes a map[ROUTER_METRIC_TYPES -> boolean].

Each element `N_advertised` becomes a map[ROUTER_METRIC_TYPES -> boolean].

6.5. Router Topology Set

Each element `TR_metric` becomes a map[ROUTER_METRIC_TYPES -> link metric].

Note that some values of `TR_metric` may now take the value `UNKNOWN_METRIC`. When used to construct a Routing Set, where just the corresponding link metric value from this mapping is used, Router Topology Tuples whose corresponding value from `TR_metric` is `UNKNOWN_METRIC` are ignored.

6.6. Routable Address Topology Set

Each element `TA_metric` becomes a map[ROUTER_METRIC_TYPES -> link metric].

Note that some values of `TA_metric` may now take the value `UNKNOWN_METRIC`. When used to construct a Routing Set, where just the corresponding link metric value from this mapping is used, Routable Address Topology Tuples whose corresponding value from `TA_metric` is `UNKNOWN_METRIC` are ignored.

6.7. Attached Network Set

Each element `AN_dist` becomes a map[ROUTER_METRIC_TYPES -> number of hops].

Each element `AN_metric` becomes a map[ROUTER_METRIC_TYPES -> link metric].

Note that some values of `AN_metric` may now take the value `UNKNOWN_METRIC`. When used to construct a Routing Set, where just the corresponding link metric value from this mapping is used, Attached Network Tuples whose corresponding value from `AN_metric` is

UNKNOWN_METRIC are ignored.

6.8. Routing Sets

There is a separate Routing Set for each link metric type in ROUTER_METRIC_TYPES.

7. TLVs

This specification makes the following additions and extensions to the TLVs defined in [RFC7181].

7.1. Message TLVs

One new Message TLV is defined in this specification, and one existing Message TLV is extended by this specification.

7.1.1. MPR_TYPES TLV

The MPR_TYPES TLV is used in both HELLO messages sent over OLSRv2 interfaces and TC messages. A message MUST NOT contain more than one MPR_TYPES TLV.

The presence of this TLV in a message is used to indicate that the router supports MT-OLSRv2, in the same way that the presence of the MPR_WILLING TLV is used to indicate that the router supports OLSRv2, as specified in [RFC7181]. For this reason, the MPR_TYPES TLV has been defined with the same Type as the MPR_WILLING TLV, but with Type Extension = 1.

This TLV may take a Value field of any size. Each octet in its Value field will contain a link metric type that is supported, either on any OLSRv2 interface, when included in a TC message, or on the OLSRv2 interface on which an including HELLO message is sent. These octets MAY be in any order, except that if there may be any routers in the MANET not implementing MT-OLSRv2, then the first octet MUST be LINK_METRIC_TYPE.

7.1.2. MPR_WILLING TLV

The MPR_WILLING TLV, which is used in HELLO messages, is specified in [RFC7181], and extended in this specification as enabled by [RFC7188].

The interpretation of this TLV, specified by [RFC7181], and which uses all of its single octet Value field, is unchanged. That interpretation uses bits 0-3 of its Value field to specify its

willingness to be a flooding TLV, and bits 4-7 of its Value field to be a routing TLV. Those latter bits are, when using this specification, interpreted as its willingness to be a routing TLV using the link metric type LINK_METRIC_TYPE.

The extended use of this message TLV, as defined by this specification, defines additional 4 bit sub-fields of the Value field, starting with bits 4-7 of the first octet and continuing with bits 0-3 of the second octet, to represent willingness to be a routing MPR using the link metric types specified in this OLSRv2 interface's IFACE_METRIC_TYPES parameter, ordered as reported in the included MPR_TYPES Message TLV. Note that this means that the link metric type LINK_METRIC_TYPE will continue to occupy bits 4-7 of the first octet. (If there is no such TLV included, then the router does not support MT-OLSRv2, and only the first octet of the Value field will be used.)

If the number of link metric types in this OLSRv2 interface's IFACE_METRIC_TYPES parameter is even, then there will be an unused 4 bit sub-field in bits 4-7 of the last octet of a full sized Value field. These bits will not be used, they SHOULD all be cleared ('0').

If the Value field in an MPR_WILLING TLV is shorter than its full length, then, as specified in [RFC7188], missing Value octets, i.e., missing willingness values, are considered as zero, i.e., as WILL_NEVER. This is the correct behavior. (In particular it means that an OLSRv2 router that is not implementing MT-OLSRv2 will not act as a routing MPR for any link metric that it does not recognize.)

7.2. Address Block TLVs

New Type Extensions are defined for the LINK_METRIC TLV defined in [RFC7181], and the Value fields of the MPR TLV and the GATEWAY TLV, both defined in [RFC7181], are extended, as enabled by [RFC7188].

7.2.1. LINK_METRIC TLV

The LINK_METRIC TLV is used in HELLO messages and TC messages. This TLV is unchanged from the definition in [RFC7181].

Only a single Type Extension was specified by [RFC7181] (link metric type) 0 as defined by administrative action. This specification extends this range to 0-7. This specification will work with any combination of Type Extensions both within and without that range (assuming that the latter are defined as specified in [RFC7181]).

7.2.2. MPR TLV

The MPR TLV is used in HELLO messages, and indicates that an address with which it is associated is of a symmetric 1-hop neighbor that has been selected as an MPR.

The Value field of this address block TLV is, in [RFC7181], defined to be one octet long, with the values 1, 2 and 3 defined. [RFC7188] redefines this Value field to be a bitfield where bit 7 (the lsb) denotes flooding status, bit 6 denotes routing MPR status, and bits 5-0 are unallocated (respecting the semantics of the bits/values 1, 2 and 3 from [RFC7181]).

This specification, as enabled by [RFC7188], extends the MPR TLV to have a variable-length Value field. For interoperability with a router not implementing MT-OLSRv2, the two least significant bits of the first octet in the Value field of this TLV MUST be the TLV Value of the MPR TLV, generated according to [RFC7181].

Subsequent bits (in increasing significance within an octet, then continuing with the least significant bit in the next octet, if required) in the TLV Value field indicate which link metric types, for which the corresponding address is selected as a routing MPR, link metric types (including the first) being indicated in, and used in the same order as, the Value field of an MPR_TYPES Message TLV, excluding the link metric type LINK_METRIC_TYPE, which already occupies the second bit.

7.2.3. GATEWAY TLV

The GATEWAY TLV is used in TC messages to indicate that a network address is of an attached network.

The Value field of this address block TLV is, in [RFC7181] defined to be one octet long, containing the number of hops to that attached network.

This specification, as enabled by [RFC7181], allows the extension the GATEWAY TLV to have a variable-length Value field when the number of hops to each attached network is different for different link metric types. For interoperability with a router not implementing MT-OLSRv2, the first octet in the Value field of this TLV MUST be the TLV Value of the GATEWAY TLV generated according to [RFC7181].

Any subsequent octets in the TLV Value field indicate the number of hops to the attached network for each other link metric type, link metric types (including the first) being indicated in the Value field of an MPR_TYPES Message TLV.

Type	Value
GATEWAY	Number of hops to attached network for each link metric type.

Table 1: GATEWAY TLV definition

8. HELLO Messages

The following changes are made to the generation and processing of HELLO messages compared to that described in [RFC7181] by routers that implement MT-OLSRv2.

8.1. HELLO Message Generation

A generated HELLO message to be sent on an OLSRv2 interface (whose `IFACE_METRIC_TYPES` parameter will be that used) is extended by:

- o Adding an `MPR_TYPES` Message TLV. The Value octets will be the link metric types in `IFACE_METRIC_TYPES`. This TLV MAY be omitted if the only link metric type included would be `LINK_METRIC_TYPE`.
- o Extending the `MPR_WILLING` Message TLV Value field to report the willingness values from the `WILL_ROUTING` parameter list that correspond to the link metric types in `IFACE_METRIC_LIST`, in the same order as reported in the `MPR_TYPES` TLV, each value (also including one representing `WILL_FLOODING`) occupying 4 bits.
- o Including `LINK_METRIC` Address Block TLVs that report all values in `L_in_metric`, `L_out_metric`, `N_in_metric` and `N_out_metric` elements that are not equal to `UNKNOWN_METRIC`, with the TLV Type Extension being the link metric type, and otherwise following the rules for such inclusions specified in [RFC7181].
- o Including `MPR` Address Block TLVs such that for each link metric type in `IFACE_METRIC_TYPES`, and for the choice of flooding MPRs, the indicated addresses MUST be of the MPRs in an MPR set as specified for a single link metric type in [RFC7181].

8.2. HELLO Message Processing

On receipt of a HELLO message on an OLSRv2 interface, a router implementing MT-OLSRv2 MUST, in addition to the processing described in [RFC7181]:

1. If in this deployment there may be any routers that do not implement MT-OLSRv2, the HELLO message contains an MPR_TYPES Message TLV, and the first link metric type that it reports is not LINK_METRIC_TYPE, then the HELLO message MUST be silently discarded.
2. Determine the list of link metric types supported by the sending router on its corresponding OLSRv2 interface, either from an MPR_TYPES Message TLV or, if not present, the single link metric type LINK_METRIC_TYPE.
3. For those link metric types supported by both routers, set the appropriate L_out_metric, N_in_metric, N_out_metric, N_will_routing, N_mpr_selector, N_advertised, N2_in_metric and N2_out_metric values as described for single such elements in [RFC7181].
4. For any other metric types supported by the receiving router only (i.e. in IFACE_METRIC for the receiving OLSRv2 interface), set the elements listed in the previous point to their default values, i.e., UNKNOWN_METRIC, WILL_NEVER (not WILL_DEFAULT), or false.

9. TC Messages

The following changes are made to the generation and processing of TC messages compared to that described in [RFC7181] by routers that implement MT-OLSRv2.

9.1. TC Message Generation

A generated TC message is extended by:

- o Adding an MPR_TYPES TLV. The value octets will be the link metric types in ROUTER_METRIC_TYPES. This MAY be omitted if the only link metric type included would be LINK_METRIC_TYPE.
- o Including LINK_METRIC TLVs that report all values of N_out_metric that are not equal to UNKNOWN_METRIC, with the TLV Type Extension being the link metric type, and otherwise following the rules for such inclusions specified in [RFC7181].
- o When not all the same, including a number of hops per reported (in an MPR_TYPES Message TLV) link metric type in the Value field of each GATEWAY TLV included, in the same order as reported in the MPR_TYPES TLV.

9.2. TC Message Processing

On receipt of a TC message, a router implementing this extension MUST, in addition to the processing specified in [RFC7181]:

- o If in this deployment there may be any routers that do not implement MT-OLSRv2, the TC message contains an MPR_TYPES Message TLV, and the first link metric type that it reports is not LINK_METRIC_TYPE, then the TC message MUST be silently discarded.
- o Set the appropriate TR_metric, TA_metric, AN_dist and AN_metric elements using the rules for setting the single elements of those types specified in [RFC7181].
- o For any other metric types supported by the receiving router that do not have an advertised outgoing neighbor metric of that type, set the corresponding elements of TR_metric, TA_metric and AN_metric to UNKNOWN_METRIC. (The corresponding element of AN_dist may be set to any value.)

10. MPR Calculation

Routing MPRs are calculated for each link metric type in ROUTER_METRIC_TYPES. Links to symmetric 1-hop neighbors via OLSRv2 interfaces that do not support that link metric type are not considered. The determined status (routing MPR or not routing MPR) for each link metric type is recorded in the relevant element of N_routing_mpr.

Each router may make its own decision as to whether or not to use a link metric, or link metrics, for flooding MPR calculation, and if so which and how. This decision MUST be made in a manner that ensures that flooded messages will reach the same symmetric 2-hop neighbors as would be the case for a router not supporting MT-OLSRv2.

Note that it is possible that a 2-Hop Tuple in the Information Base for a given OLSRv2 interface does not support any of the link metric types that are in the router's corresponding IFACE_METRIC_TYPES, but nevertheless that 2-Hop Tuple MUST be considered when determining flooding MPRs.

11. Routing Set Calculation

A Routing Set is calculated for each link metric type in ROUTER_METRIC_TYPES. The calculation may be as for [RFC7181], except that where an element is now represented by a map, the value from the

map for the selected link metric type is used. Where this is a link metric of value UNKNOWN_METRIC, that protocol Tuple is ignored for the calculation.

12. Management Considerations

MT-OLSRv2 may require greater management than unextended OLSRv2. In particular a MANET using MT-OLSRv2 requires the following management considerations:

- o Deciding which link metric, and hence which Routing Set to use, for received packets, hence how to use the Routing Sets to configure the network layer (IP). All routers MUST make the same decision for the same packet. An obvious approach is to map each DiffServ Code Point (DSCP) [RFC2474] to a single link metric. (This may be a many to one mapping.)
- o Selecting which link metrics to support on each OLSRv2 interface and implementing that decision. (Different interfaces may have different physical and data link layer properties, and this may inform the selection of link metrics to support, and their values.) If the MANET may contain non-MT-OLSRv2 routers, which is also subject to management, then the rules for link metric assignment to OLSRv2 interfaces in this specification for that case MUST be followed.
- o Ensuring that the MANET is sufficiently connected, by ensuring that, for example, sufficiently many routers implement each metric type required (this being easier in, for example, a denser network). Note that if there is any possibility that there are any routers not implementing MT-OLSRv2, then the MANET will be connected, to the maximum extent possible, using the link metric type LINK_METRIC_TYPE, but this will only serve to deliver packets that use that link metric type.
- o Non-MT-OLSRv2 routers SHOULD be managed so as not produce packets that will be routed by a topology that they are not part of. However if they were to do so then such packets will be routed until either they reach their destination, or they reach an MT-OLSRv2 router. In the latter case the packet will then either be dropped (if that MT-OLSRv2 router is not part of that topology, or is not aware of the destination within that topology) or will be routed by that topology to the destination. Such a packet will not loop.
- o If a packet is created for a destination that is not part of the corresponding topology then it may or may not be delivered (if the

originating router is a non-MT-OLSRv2 router) or will not be transmitted (if the originating router is an MT-OLSRv2 router). Routers SHOULD be managed so that this does not occur.

13. IANA Considerations

This specification adds one new Message TLV, allocated as a new Type Extension to an existing Message TLV, using a new name. It also modifies the Value field of an existing Message TLV, and of an existing Address Block TLV. Finally, this specification makes additional allocations from the LINK_METRIC Address Block TLV Type registry.

13.1. Expert Review: Evaluation Guidelines

For the registry where an Expert Review is required, the designated expert SHOULD take the same general recommendations into consideration as are specified by [RFC5444] and [RFC7631].

13.2. Message TLV Types

This specification modifies the Message TLV Type 7, replacing Table 4 of [RFC7631] by Table 2, changing the description of the Type Extension MPR_WILLING and adding the Type Extension TLV_TYPES. Each of these TLVs MUST NOT be included more than once in a Message TLV Block.

Type Extension	Name	Description	Reference
0	MPR_WILLING	First (most significant) half octet of Value field specifies the originating router's willingness to act as a flooding MPR; subsequent half octets specify the originating router's willingness to act as a routing MPR, either for the link metric types reported in an MPR_TYPES TLV (in the same order), or (if no MPR_TYPES TLV is present) for the single administratively agreed link metric type	[RFC7181] [RFC7631] This specification
1	MPR_TYPES	The link metric types supported on this OLSRv2 interface of this router (one octet each).	This specification
2-223 224-255		Unassigned Reserved for Experimental Use	[RFC7181]

Table 2: Type 7 Message TLV Type Extensions

13.3. Address Block TLV Types

Table 7 of [RFC7188] is replaced by Table 3.

Bit	Value	Name	Description
First octet bit 7	First octet 0x01	Flooding	If set then the neighbor with that network address has been selected as flooding MPR
From first octet bit 6	From first octet 0x02	Routing	If set then the neighbor with that network address has been selected as routing MPR, either for the link metric types reported in an MPR_TYPES TLV (in the same order), or (if no MPR_TYPES TLV is present) then (first octet bit 6, value 0x02) for the single administratively agreed link metric type

Table 3: MPR TLV Bit Values

Table 14 of [RFC7631] is replaced by Table 4. The only changes are to the Description and the References for the GATEWAY TLV.

Type Extension	Name	Description	References
0	GATEWAY	Specifies that a given network address is reached via a gateway on the originating router. The number of hops is indicated by the Value field, one octet per link metric type reported in an MPR_TYPES Message TLV (in the same order) or (if no MPR_TYPES Message TLV is present) using a single octet	[RFC7181] This specification
1-223 224-255		Unassigned Reserved for Experimental Use	[RFC7631]

Table 4: Type 10 Address Block TLV Type Extensions

Table 13 of [RFC7181] is replaced by Table 5. The only change is to allocate 8 Type Extensions as assigned by administrative action, in order to support administratively determined multi-topologies.

Name	Type	Type Extension	Description	Allocation Policy
LINK_METRIC	7	0-7	Link metric meaning assigned by administrative action.	
LINK_METRIC	7	8-223	Unassigned.	Expert Review
LINK_METRIC	7	224-255	Unassigned.	Experimental Use

Table 5: Address Block TLV Type assignment: LINK_METRIC

14. Security Considerations

This extension to OLSRv2 allows a router to support more than one link metric type for each link advertised in HELLO and TC messages, and for routers to support different sets of types. Link metric values of additional types are reported by the inclusion of additional TLVs in the messages sent by a router, which will report known values of all supported types.

HELLO and TC message processing is then extended simply to record, for each supported type, all of the received link metric values for each link. Protocol internal processing (specifically MPR set and shortest path calculations) then operate as specified in [RFC7181] for each link metric type that the router supports.

Consequently the security considerations, including the security architecture and the mandatory security mechanisms, from [RFC7181] are directly applicable to MT-OLSRv2.

Furthermore, this extension does not introduce any additional vulnerabilities over those of [RFC7181], because each link metric type is used independently, and each one could have been the single link metric type supported by an implementation of [RFC7181] receiving the same information, as received information of an unsupported type is ignored by all routers.

15. Acknowledgments

The authors would like to thank (in alphabetical order): Juliusz Chroboczek (University of Paris Diderot), Alan Cullen (BAE Systems),

Susan Hares (Huawei) and Henning Rogge (FGAN) for discussions and suggestions.

16. References

16.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC5444] Clausen, T., Dearlove, C., Dean, J., 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.
- [RFC7181] Clausen, T., Dearlove, C., Jacquet, P., and U. Herberg, "The Optimized Link State Routing Protocol version 2", RFC 7181, April 2014.
- [RFC7188] Dearlove, C. and T. Clausen, "Optimized Link State Routing Protocol version 2 (OLSRv2) and MANET Neighborhood Discovery Protocol (NHDP) Extension TLVs", RFC 7188, April 2014.
- [RFC7631] Dearlove, C. and T. Clausen, "TLV Naming in the MANET Generalized Packet/Message Format", RFC 7631, January 2015.

16.2. Informative References

- [RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.
- [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.
- [RFC4915] Psenak, P., Mirtorabi, S., Roy, A., Nguyen, L., and P. Pillay-Esnault, "Multi-Topology (MT) Routing in OSPF",

RFC 4915, June 2007.

Authors' Addresses

Christopher Dearlove
BAE Systems Applied Intelligence Laboratories
West Hanningfield Road
Great Baddow, Chelmsford
United Kingdom

Phone: +44 1245 242194
Email: chris.dearlove@baesystems.com
URI: <http://www.baesystems.com/>

Thomas Heide Clausen
LIX, Ecole Polytechnique

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

Mobile Ad hoc Networking (MANET)
Internet-Draft
Updates: OLSRv2 (if approved)
Intended status: Standards Track
Expires: August 25, 2014

C. Dearlove
BAE Systems ATC
T. Clausen
LIX, Ecole Polytechnique
February 21, 2014

Routing MPR Optimization for the Optimized Link State Routing Protocol
version 2 (OLSRv2)
draft-ietf-manet-olsrv2-rmpr-optimization-01

Abstract

This specification updates the Optimized Link State Routing Protocol version 2 (OLSRv2) with an optimization to improve the selection of routing Multi-Point Relays. The optimization retains full interoperability between implementations of OLSRv2 with and without this optimization.

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 25, 2014.

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. Terminology 3
- 3. Applicability Statement 3
- 4. Routing MPR Selection 4
- 5. IANA Considerations 4
- 6. Security Considerations 4
- 7. Acknowledgments 5
- 8. References 5
 - 8.1. Normative References 5
 - 8.2. Informative References 5
- Authors' Addresses 5

1. Introduction

The Optimized Link State Routing Protocol, version 2 [OLSRv2] is a proactive link state routing protocol designed for use in mobile ad hoc networks (MANETs) [RFC2501]. This document improves one area of that specification.

One improvement included in OLSRv2, compared to its predecessor described in [RFC3626], is the use of link metrics, rather than minimum hop routing. A rationale for how link metrics were included in OLSRv2 is documented in [RFC6966-to-be]. However, one aspect of the use of link metrics described in [RFC6966-to-be], the removal of some unnecessarily selected routing Multi-Point Relays (MPRs), was not included in [OLSRv2]. This specification updates OLSRv2 to include this optimization.

Note that this optimization does not impact interoperability: implementations which do, and which do not, implement this optimization will interoperate seamlessly.

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].

Additionally, this document uses the terminology of [OLSRv2].

3. Applicability Statement

This specification updates [OLSRv2]. As such it is applicable to all implementations of this protocol. The optimization presented in this specification is simply permissive, it allows an additional optimization, and there is no requirement for any implementation to include it. However inclusion of this optimization is advised; it can, in some cases, create smaller and fewer messages, without ever having the opposite effect.

[OLSRv2] defines the properties for the selection of routing MPRs (multipoint relays) from among a router's symmetric 1-hop neighbors as properties that correspond first to that the selected MPRs consist of a set of symmetric 1-hop neighbors that cover all the symmetric 2-hop neighbors, and second that they do so retaining a minimum distance route (1-hop, if present, or 2-hop) to each symmetric 2-hop neighbor. The discussion in the latter part of Section 6.2 of

[RFC6966-to-be] indicates that this requirement is over-prescriptive for routing MPR selection. The update to [OLSRv2] described in this specification permits a router to use the described optimization, while still being considered compliant.

Note that, whether considered compliant or not, a router that implements the optimization, described in this specification, will interoperate successfully with routers not implementing this optimization.

4. Routing MPR Selection

A set of routing MPRs created as specified in [OLSRv2] MAY be optimized in the following manner. Note that this uses the notation of Section 18.2 of [OLSRv2]:

1. If there is a sequence x_0, \dots, x_n of elements of $N1$ such that:

- * x_0 is a routing MPR,
- * x_1, \dots, x_n have corresponding elements y_1, \dots, y_n of $N2$, and
- * $d1(x_0) + d2(x_0, y_1) + \dots + d2(x_{m-1}, y_m) < d1(x_m)$ for $m = 1, \dots, n$,

then x_1 to x_n may be removed from the set of routing MPRs, if selected.

Note that "corresponding elements" in $N1$ and $N2$ means that these elements represent the same router. All of this information is available from information gathered by NHDP [RFC6130].

5. IANA Considerations

This document has no actions for IANA.

[This section may be removed by the RFC Editor.]

6. Security Considerations

The update to [OLSRv2] does not introduce any new protocol signals, compared to the already published specifications of the protocol, nor does it change the processing of any received protocol signals.

This update to [OLSRv2] permits a compliant implementation of OLSRV2 to (potentially) eliminate some redundant information from the routing MPR sets otherwise generated by the algorithms described in [OLSRv2], and therefore also eliminate the need for including that information in generated TC messages. Because this information is not used when included, this update to [OLSRv2] does not present any additional security considerations, beyond those described in [OLSRv2].

7. Acknowledgments

The authors would like to gratefully acknowledge Philippe Jacquet (Alcatel-Lucent) for intense technical discussions and comments.

8. References

8.1. Normative References

- [OLSRv2] Clausen, T., Dearlove, C., Jacquet, P., and U. Herberg, "The Optimized Link State Routing Protocol version 2", work in progress draft-ietf-manet-olsrv2-19, March 2013.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6130] Clausen, T., Dean, J., and C. Dearlove, "Mobile Ad Hoc Network (MANET) Neighborhood Discovery Protocol (NHDP)", RFC 6130, April 2011.

8.2. 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.
- [RFC6966-to-be] Clausen, T., Dearlove, C., and P. Jacquet, "Rationale for the Use of Link Metrics in the Optimized Link State Routing Protocol Version 2 (OLSRv2)", RFC 6966, TBD 2013.

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
URI: <http://www.baesystems.com/>

Thomas Heide Clausen
LIX, Ecole Polytechnique

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

MANET
Internet-Draft
Intended status: Informational
Expires: August 18, 2014

H. Rogge
Fraunhofer FKIE
E. Baccelli
INRIA
February 14, 2014

Packet Sequence Number based directional airtime metric for OLSRv2
draft-rogge-baccelli-olsrv2-ett-metric-04

Abstract

This document specifies an directional airtime link metric for usage 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 August 18, 2014.

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. Terminology 3
- 3. Applicability Statement 4
- 4. Directional Airtime Metric Rational 5
- 5. Metric Functioning & Overview 6
- 6. Protocol Parameters 7
 - 6.1. Recommended Values 7
- 7. Protocol Constants 7
- 8. Data Structures 8
 - 8.1. Initial Values 9
- 9. Packets and Messages 9
 - 9.1. Definitions 9
 - 9.2. Requirements 9
 - 9.3. Link Loss Data Gathering 10
 - 9.3.1. Packet Sequence based link loss 10
 - 9.3.2. HELLO based Link Loss 11
 - 9.3.3. Other Measurement of Link Loss 11
 - 9.4. HELLO Message Processing 11
- 10. HELLO Timeout Processing 12
- 11. Metric Update 12
- 12. IANA Considerations 13
- 13. Security Considerations 13
- 14. Acknowledgements 13
- 15. References 14
 - 15.1. Normative References 14
 - 15.2. Informative References 14
- Appendix A. OLSR.org metric history 15
- Authors' Addresses 16

1. Introduction

One of the major shortcomings of OLSR [RFC3626] is the missing of a link cost metric between mesh nodes. Operational experience with mesh networks gathered since the standardization of OLSR has revealed that wireless networks links can have highly variable and heterogeneous properties. This makes a hopcount metric insufficient for effective mesh routing.

Based on this experience, OLSRv2 [OLSRV2] integrates the concept of link metrics directly into the core specification of the routing protocol. The OLSRv2 routing metric is an external process, it can be any kind of dimensionless additive cost function which reports to the OLSRv2 protocol.

Since 2004 the OLSR.org [OLSR.org] implementation of OLSR included an Estimated Transmission Count (ETX) metric [MOBICOM04] as a proprietary extension. While this metric is not perfect, it proved to be sufficient for a long time for Community Mesh Networks (Appendix A). But the increasing maximum data rate of IEEE 802.11 made the ETX metric less efficient than in the past, which is one reason to move to a different metric.

This document describes a Directional Airtime routing metric for OLSRv2, a successor of the OLSR.org routing metric for [RFC3626]. It takes both the loss rate and the link speed into account to provide a more accurate picture of the mesh network links.

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].

The terminology introduced in [RFC5444], [OLSRV2] and [RFC6130], including the terms "packet", "message" and "TLV" are to be interpreted as described therein.

Additionally, this document uses the following terminology and notational conventions:

QUEUE - a first in, first out queue of integers.

QUEUE[TAIL] - the most recent element in the queue.

add(QUEUE, value) - adds a new element to the TAIL of the queue.

remove(QUEUE) - removes the HEAD element of the queue

sum(QUEUE) - an operation which returns the sum of all elements in a QUEUE.

diff_segno(new, old) - an operation which returns the positive distance between two elements of the circular sequence number space defined in section 5.1 of [RFC5444]. Its value is either (new - old) if this result is positive, or else its value is (new - old + 65536).

MAX(a,b) - the maximum of a and b.

UNDEFINED - a value not in the normal value range of a variable. Might be -1 for this protocol.

airtime - the time a transmitted packet blocks the link layer, e.g., a wireless link.

ETX - Expected Transmission Count, a link metric proportional to the number of transmissions to successfully send an IP packet over a link.

ETT - Estimated Travel Time, a link metric proportional to the amount of airtime needed to transmit an IP packet over a link, not considering layer-2 overhead created by preamble, backoff time and queuing.

DAT - Directional Airtime Metric, the link metric described in this document, which is a directional variant of ETT. It does not take reverse path loss into account.

3. Applicability Statement

The Directional Airtime Metric was designed and tested in wireless IEEE 802.11 mesh networks. These networks employ link layer retransmission to increase the delivery probability and multiple unicast data rates.

The metric must learn about the unicast data rate towards each one-hop neighbor from an external process, either by configuration or by an external measurement process. This measurement could be done by gathering cross-layer data from the operating system or an external

daemon like DLEP [DLEP], but also by indirect layer-3 measurements like packet-pair.

If [RFC5444] control traffic is used to determine the link packet loss, the administrator should take care that link layer multicast transmission do not not have a higher reception probability than the slowest unicast transmission. It might be necessary to increase the data-rate of the multicast transmissions, e.g. set the multicast data-rate to 6 MBit/s if you use IEEE 802.11g only.

The metric can only handle a certain range of packet loss and unicast data-rate. Maximum packet loss is "ETX 4" (1 of 4 packets is successfully sent to the receiver, without link layer retransmissions), the unicast data-rate can be between 1024 Bit/s and 4 GBit/s. The metric has been designed for data-rates of 1 MBit/s and hundreds of MBit/s.

4. Directional Airtime Metric Rational

The Directional Airtime Metric has been inspired by the publications on the ETX [MOBICOM03] and ETT [MOBICOM04] metric, but has several key differences.

Instead of measuring the combined loss probability of a bidirectional transmission of a packet over a link in both directions, the Directional Airtime Metric measures the incoming loss rate and integrates the incoming link speed into the metric cost. There are multiple reasons for this decision:

- o OLSRV2 [OLSRV2] defines the link metric as directional costs between nodes.
- o Not all link layer implementations use acknowledgement mechanisms. Most link layer implementations who do use them use less airtime and a more robust modulation for the acknowledgement than the data transmission, which makes it more likely for the data transmission to be disrupted compared to the acknowledgement.
- o Incoming packet loss and link speed can be measured locally, symmetric link loss would need an additional signaling TLV in the [RFC6130] HELLO and would delay metric calculation by up to one HELLO interval.

The Directional Airtime Metric does not integrate the packet size into the link cost. Doing so is not feasible in most link-state routing protocol implementations. The routing decision of most operation systems don't take packet size into account. Multiplying

all link costs of a topology with the size of a data-plane packet would never change the dijkstra result anyways.

The queue based packet loss estimator has been tested extensively in the OLSR.org ETX implementation, see Appendix A. The output is the average of the packet loss over a configured time period.

5. Metric Functioning & Overview

The Directional Airtime Metric is calculated for each link set entry, as defined in [RFC6130] section 7.1.

The metric processes two kinds of data into the metric value, namely packet loss rate and link-speed. While the link-speed is taken from an external process, the current packet loss rate is calculated by keeping track of packet reception and packet loss events.

Multiple incoming packet loss/reception events must be combined into a loss rate to get a smooth metric. Experiments with exponential weighted moving average (EWMA) lead to a highly fluctuating or a slow converging metric (or both). To get a smoother and more controllable metric result, this metric uses two fixed length queues to measure and average the incoming packet events, one queue for received packets and one for the estimated number of packets sent by the other side of the link.

Because the rate of incoming packets is not uniform over time, the queue contains a number of counters, each representing a fixed time interval. Incoming packet loss and packet reception event are accumulated in the current queue element until a timer adds a new empty counter to both queues and remove the oldest counter from both.

In addition to the packet loss stored in the queue, this metric uses a timer to detect a total link-loss. For every NHDP HELLO interval in which the metric received no packet from a neighbor, it scales the number of received packets in the queue based on the total time interval the queue represents compared to the total time of the lost HELLO intervals.

The average packet loss ratio is calculated as the sum of the 'total packets' counters divided by the sum of the 'packets received' counters. This value is then divided through the current link-speed and then scaled into the range of metrics allowed for OLSRV2.

The metric value is then used as `L_in_metric` of the Link Set (as defined in section 8.1. of [OLSRV2]).

6. Protocol Parameters

This specification defines the following parameters, which can be changed without making the metric outputs incomparable with each other:

DAT_MEMORY_LENGTH - Queue length for averaging packet loss. All received and lost packets within the queue are used to calculate the cost of the link.

DAT_REFRESH_INTERVAL - interval in seconds between two metric recalculations as described in Section 11. This value SHOULD be smaller than a typical HELLO interval.

DAT_HELLO_TIMEOUT_FACTOR - timeout factor for HELLO interval at which point a HELLO is definitely considered lost. The value must be a floating point number between 1.0 and 2.0, large enough to take the delay and jitter for message aggregation into account.

DAT_SEQNO_RESTART_DETECTION - threshold in number of missing packets (based on received packet sequence numbers) at which point the router considers the neighbor has restarted. This parameter is only used for packet sequence number based loss estimation. This number MUST be larger than DAT_MAXIMUM_LOSS.

6.1. Recommended Values

The proposed values of the protocol parameters are for Community Mesh Networks, which mostly use immobile mesh nodes. Using this metric for mobile networks might require shorter DAT_REFRESH_INTERVAL and/or DAT_MEMORY_LENGTH.

DAT_MEMORY_LENGTH := 64

DAT_REFRESH_INTERVAL := 1

DAT_HELLO_TIMEOUT_FACTOR := 1.2

DAT_SEQNO_RESTART_DETECTION := 256

7. Protocol Constants

This specification defines the following constants, which cannot be changed without making the metric outputs incomparable:

DAT_MAXIMUM_LOSS - Fraction of the loss rate used in this routing metric. Loss rate will be between $0/\text{DAT_MAXIMUM_LOSS}$ and $(\text{DAT_MAXIMUM_LOSS}-1)/\text{DAT_MAXIMUM_LOSS}$: 4.

DAT_MINIMUM_BITRATE - Minimal bit-rate in Bit/s used by this routing metric: 1024.

8. Data Structures

This specification extends the Link Set Tuples of the Interface Information Base, as defined in [RFC6130] section 7.1, by the following additional elements for each link tuple when being used with this metric:

L_DAT_received is a QUEUE with DAT_MEMORY_LENGTH integer elements. Each entry contains the number of successfully received packets within an interval of DAT_REFRESH_INTERVAL.

L_DAT_total is a QUEUE with DAT_MEMORY_LENGTH integer elements. Each entry contains the estimated number of packets transmitted by the neighbor, based on the received packet sequence numbers within an interval of DAT_REFRESH_INTERVAL.

L_DAT_hello_time is the time when the next hello will be expected.

L_DAT_hello_interval is the interval between two hello messages of the links neighbor as signaled by the INTERVAL_TIME TLV [RFC5497] of NHDP messages [RFC6130].

L_DAT_lost_hello_messages is the estimated number of lost hello messages from this neighbor, based on the value of the hello interval.

L_DAT_rx_bitrate is the current bitrate of incoming unicast traffic for this neighbor.

Methods to obtain the value of L_DAT_rx_bitrate are out of the scope of this specification. Such methods may include static configuration via a configuration file or dynamic measurement through mechanisms described in a separate specification (e.g. [DLEP]). Any Link tuple with L_status = HEARD or L_status = SYMMETRIC MUST have a specified value of L_DAT_rx_bitrate if it is to be used by this routing metric.

When using packet sequence numbers to estimate the loss rate, the Link Set Tuples get another field:

L_DAT_last_pkt_seqno is the last received packet sequence number received from this link.

8.1. Initial Values

When generating a new tuple in the Link Set, as defined in [RFC6130] section 12.5 bullet 3, the values of the elements specified in Section 8 are set as follows:

- o L_DAT_received := 0, ..., 0. The queue always has DAT_MEMORY_LENGTH elements.
- o L_DAT_total := 0, ..., 0. The queue always has DAT_MEMORY_LENGTH elements.
- o L_DAT_last_pkt_seqno := UNDEFINED (no earlier packet received).
- o L_DAT_hello_time := EXPIRED (no earlier NHDP HELLO received).
- o L_DAT_hello_interval := UNDEFINED (no earlier NHDP HELLO received).
- o L_DAT_lost_hello_messages := 0 (no HELLO interval without packets).

9. Packets and Messages

9.1. Definitions

For the purpose of this section, note the following definitions:

- o "pkt_seqno" is defined as the [RFC5444] packet sequence number of the received packet.
- o "interval_time" is the time encoded in the INTERVAL_TIME message TLV of a received [RFC6130] HELLO message.

9.2. Requirements

An implementation of OLSRv2 using the metric specified by this document MUST include the following parts into its [RFC5444] output:

- o an INTERVAL_TIME message TLV in each HELLO message, as defined in [RFC6130] section 4.3.2.

9.3. Link Loss Data Gathering

While this metric was designed for measuring the packet loss based on the [RFC5444] packet sequence number, some implementations might not be able to add the packet sequence number to their output.

9.3.1. Packet Sequence based link loss

An implementation of OLSRV2, using the metric specified by this document with packet sequence based link loss, MUST include the following element into its [RFC5444] output:

- o an interface specific packet sequence number as defined in [RFC5444] section 5.1 which is incremented by 1 for each outgoing [RFC5444] packet on the interface.

For each incoming [RFC5444] packet, additional processing MUST be carried out after the packet messages have been processed as specified in [RFC6130] and [OLSRV2].

[RFC5444] packets without packet sequence number MUST NOT be processed in this way by this metric.

The router MUST update the Link Set Tuple corresponding to the originator of the packet:

1. If L_DAT_last_pkt_seqno = UNDEFINED, then:
 1. L_DAT_received[TAIL] := 1.
 2. L_DAT_total[TAIL] := 1.
2. Otherwise:
 1. L_DAT_received[TAIL] := L_DAT_received[TAIL] + 1.
 2. diff := seq_diff(pkt_seqno, L_DAT_last_pkt_seqno).
 3. If diff > DAT_SEQNO_RESTART_DETECTION, then:
 1. diff := 1.
 4. L_DAT_total[TAIL] := L_DAT_total[TAIL] + diff.
3. L_DAT_last_pkt_seqno := pkt_seqno.
4. If L_DAT_hello_interval != UNDEFINED, then:

1. `L_DAT_hello_time := current time + (L_DAT_hello_interval * DAT_HELLO_TIMEOUT_FACTOR).`

5. `L_DAT_lost_hello_messages := 0.`

9.3.2. HELLO based Link Loss

A metric might just use the incoming NHDP HELLO messages of a neighbor to calculate the link loss. Because this method uses fewer events to calculate the metric, the variance of the output will increase. It might be necessary to increase the value of `DAT_MEMORY_LENGTH` to compensate for this.

For each incoming HELLO message, after it has been processed as defined in [RFC6130] section 12, the Link Set Tuple as defined in section 7.1 corresponding to the incoming HELLO message must be updated.

1. `L_DAT_received[TAIL] := L_DAT_received[TAIL] + 1.`

2. `L_DAT_total[TAIL] := L_DAT_total[TAIL] + 1.`

3. `L_DAT_lost_hello_messages := 0.`

9.3.3. Other Measurement of Link Loss

Instead of using incoming [RFC5444] packets or [RFC6130] messages, the routing daemon can also use other sources to measure the link layer lossrate (e.g. [DLEP]).

To use a source like this with the DAT metric, the routing daemon has to add incoming total traffic (or the sum of received and lost traffic) and lost traffic to the queued elements in the extension of the Link Set Tuple defined in Section 8 corresponding to originator of the traffic.

The routing daemon should also set `L_DAT_lost_hello_messages` to zero every times new packages arrive.

9.4. HELLO Message Processing

For each incoming HELLO Message, after it has been processed as defined in [RFC6130] section 12, the Link Set Tuple corresponding to the incoming HELLO message must be updated.

Only HELLO messages with an `INTERVAL_TIME` message TLVs must be processed.

1. L_DAT_hello_interval := interval_time.

10. HELLO Timeout Processing

When L_DAT_hello_time has timed out, the following step MUST be done:

1. L_DAT_lost_hello_messages := L_DAT_lost_hello_messages + 1.
2. L_DAT_hello_time := L_DAT_hello_time + L_DAT_hello_interval.

11. Metric Update

Once every DAT_REFRESH_INTERVAL, all L_in_metric values in all Link Set entries MUST be recalculated:

1. sum_received := sum(L_DAT_total).
2. sum_total := sum(L_DAT_received).
3. If L_DAT_hello_interval != UNDEFINED and L_DAT_lost_hello_messages > 0, then:
 1. lost_time_proportion := L_DAT_hello_interval * L_DAT_lost_hello_messages / DAT_MEMORY_LENGTH.
 2. sum_received := sum_received * MAX (0, 1 - lost_time_proportion);
4. If sum_received < 1, then:
 1. L_in_metric := MAXIMUM_METRIC, as defined in [OLSRV2] section 5.6.1.
5. Otherwise:
 1. loss := sum_total / sum_received.
 2. If loss > DAT_MAXIMUM_LOSS, then:
 1. loss := DAT_MAXIMUM_LOSS.
 3. bitrate := L_DAT_rx_bitrate.
 4. If bitrate < DAT_MINIMUM_BITRATE, then:

1. `bitrate := DAT_MINIMUM_BITRATE.`
5. `L_in_metric := (2^24 / DAT_MAXIMUM_LOSS) * loss / (bitrate / DAT_MINIMUM_BITRATE).`
6. `remove(L_DAT_total)`
7. `add(L_DAT_total, 0)`
8. `remove(L_DAT_received)`
9. `add(L_DAT_received, 0)`

12. IANA Considerations

This document contains no actions for IANA.

13. Security Considerations

Artificial manipulation of metrics values can drastically alter network performance. In particular, advertising a higher `L_in_metric` value may decrease the amount of incoming traffic, while advertising lower `L_in_metric` may increase the amount of incoming traffic. By artificially increasing or decreasing the `L_in_metric` values it advertises, a rogue router may thus attract or repulse data traffic. A rogue router may then potentially degrade data throughput by not forwarding data as it should or redirecting traffic into routing loops or bad links.

An attacker might also inject packets with incorrect packet level sequence numbers, pretending to be somebody else. This attack could be prevented by the true originator of the RFC5444 packets by adding a [RFC6622] ICV Packet TLV and TIMESTAMP Packet TLV to each packet. This allows the receiver to drop all incoming packets which have a forged packet source, both packets generated by the attacker or replayed packets.

14. Acknowledgements

The authors would like to acknowledge the network administrators from Freifunk Berlin [FREIFUNK] and Funkfeuer Vienna [FUNKFEUER] for endless hours of testing and suggestions to improve the quality of the original ETX metric for the OLSR.org routing daemon.

This effort/activity is supported by the European Community Framework

Program 7 within the Future Internet Research and Experimentation Initiative (FIRE), Community Networks Testbed for the Future Internet ([CONFINE]), contract FP7-288535.

The authors would like to gratefully acknowledge the following people for intense technical discussions, early reviews and comments on the specification and its components (listed alphabetically): Teco Boot (Infinity Networks), Juliusz Chroboczek (PPS, University of Paris 7), Thomas Clausen, Christopher Dearlove (BAE Systems Advanced Technology Centre), Ulrich Herberg (Fujitsu Laboratories of America), Markus Kittenberger (Funkfeuer Vienna), Joseph Macker (Naval Research Laboratory) and Stan Ratliff (Cisco Systems).

15. References

15.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, BCP 14, March 1997.
- [RFC3626] Clausen, T. and P. Jacquet, "Optimized Link State Routing Protocol", RFC 3626, October 2003.
- [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.
- [RFC6622] Ulrich, U. and T. Clausen, "Integrity Check Value and Timestamp TLV Definitions for Mobile Ad Hoc Networks (MANETs)", RFC 6622, May 2012.
- [OLSRV2] Clausen, T., Jacquet, P., and C. Dearlove, "The Optimized Link State Routing Protocol version 2", draft-ietf-manet-olsrv2-19 , March 2013.

15.2. Informative References

- [CONFINE] "Community Networks Testbed for the Future Internet (CONFINE)", 2013, <<http://www.confine-project.eu>>.

- [DLEP] Ratliff, S., Berry, B., Harrison, G., Jury, S., and D. Satterwhite, "Dynamic Link Exchange Protocol (DLEP)", draft-ietf-manet-dlep-04 , March 2013.
- [MOBICOM03] De Couto, D., Aguayo, D., Bicket, J., and R. Morris, "A High-Throughput Path Metric for Multi-Hop Wireless Routing", Proceedings of the MOBICOM Conference , 2003.
- [MOBICOM04] Richard, D., Jitendra, P., and Z. Brian, "Routing in Multi-Radio, Multi-Hop Wireless Mesh Networks", Proceedings of the MOBICOM Conference , 2004.
- [OLSR.org] "The OLSR.org OLSR routing daemon", 2013, <<http://www.olsr.org/>>.
- [FREIFUNK] "Freifunk Wireless Community Networks", 2013, <<http://www.freifunk.net>>.
- [FUNKFEUER] "Austria Wireless Community Network", 2013, <<http://www.funkfeuer.at>>.

Appendix A. OLSR.org metric history

The Funkfeuer [FUNKFEUER] and Freifunk networks [FREIFUNK] are OLSR-based [RFC3626] or B.A.T.M.A.N. based wireless community networks with hundreds of routers in permanent operation. The Vienna Funkfeuer network in Austria, for instance, consists of 400 routers (around 600 routes) covering the whole city of Vienna and beyond, spanning roughly 40km in diameter. It has been in operation since 2003 and supplies its users with Internet access. A particularity of the Vienna Funkfeuer network is that it manages to provide Internet access through a city wide, large scale Wi-Fi mesh network, with just a single Internet uplink.

Operational experience of the OLSR project [OLSR.org] with these networks have revealed that the use of hop-count as routing metric leads to unsatisfactory network performance. Experiments with the ETX metric [MOBICOM03] were therefore undertaken in parallel in the Berlin Freifunk network as well as in the Vienna Funkfeuer network in 2004, and found satisfactory, i.e., sufficiently easy to implement and providing sufficiently good performance. This metric has now been in operational use in these networks for several years.

The ETX metric of a link is the estimated number of transmissions required to successfully send a packet (each packet equal to or smaller than MTU) over that link, until a link layer acknowledgement is received. The ETX metric is additive, i.e., the ETX metric of a path is the sum of the ETX metrics for each link on this path.

While the ETX metric delivers a reasonable performance, it doesn't handle well networks with heterogeneous links that have different bitrates. Since every wireless link, when using ETX metric, is characterized only by its packet loss ratio, the ETX metric prefers long-ranged links with low bitrate (with low loss ratios) over short-ranged links with high bitrate (with higher but reasonable loss ratios). Such conditions, when they occur, can degrade the performance of a network considerably by not taking advantage of higher capacity links.

Because of this the OLSR.org project has implemented the Directional Airtime Metric for OLSRv2, which has been inspired by the Estimated Travel Time (ETT) metric [MOBICOM04]. This metric uses an unidirectional packet loss, but also takes the bitrate into account to create a more accurate description of the relative costs or capabilities of mesh links.

Authors' Addresses

Henning Rogge
Fraunhofer FKIE

Email: henning.rogge@fkie.fraunhofer.de
URI: <http://www.fkie.fraunhofer.de>

Emmanuel Baccelli
INRIA

Email: Emmanuel.Baccelli@inria.fr
URI: <http://www.emmanuelbaccelli.org/>

