Mobile Ad hoc Networking (MANET)
Internet-Draft
Intended status: Standards Track
Expires: September 19, 2013

T. Clausen
LIX, Ecole Polytechnique
C. Dearlove
BAE Systems ATC
P. Jacquet
Alcatel-Lucent Bell Labs
U. Herberg
Fujitsu Laboratories of America
March 18, 2013

The Optimized Link State Routing Protocol version 2
draft-ietf-manet-olsrv2-18

Abstract

This specification describes version 2 of the Optimized Link State Routing (OLSRv2) protocol for Mobile Ad hoc NETworks (MANETs).

Status of This Memo

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

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 19, 2013.

Table of Contents

1. Introduction

The Optimized Link State Routing protocol version 2 (OLSRv2) is the successor to OLSR (version 1) as published in [RFC3626]. Compared to [RFC3626], OLSRv2 retains the same basic mechanisms and algorithms, enhanced by the ability to use a link metric other than hop count in the selection of shortest routes. OLSRv2 also uses a more flexible and efficient signaling framework, and includes some simplification of the messages being exchanged.

OLSRv2 is developed for mobile ad hoc networks (MANETs). It operates as a table driven, proactive protocol, i.e., it exchanges topology information with other routers in the network regularly. OLSRv2 is an optimization of the classic link state routing protocol. Its key concept is that of MultiPoint Relays (MPRs). Each router selects two sets of MPRs, each being a set of its neighbor routers that "cover" all of its symmetrically connected 2-hop neighbor routers. These two sets are of "flooding MPRs" and "routing MPRs", and are used to achieve flooding reduction and topology reduction, respectively.

Flooding reduction is achieved by control traffic being flooded through the network using hop by hop forwarding, but with a router only needing to forward control traffic that is first received directly from one of the routers that have selected it as a flooding MPR (its "flooding MPR selectors"). This mechanism, denoted "MPR flooding", provides an efficient mechanism for information distribution within the MANET by reducing the number of transmissions required [MPR].

Topology reduction is achieved by assigning a special responsibility to routers selected as routing MPRs when declaring link state information. A sufficient requirement for OLSRv2 to provide shortest routes to all destinations is that routers declare link state information for their routing MPR selectors, if any. Routers that are not selected as routing MPRs need not send any link state information. Based on this reduced link state information, routing MPRs are used as intermediate routers in multi-hop routes.

Thus the use of MPRs allows reduction of the number and the size of link state messages, and in the amount of link state information maintained in each router. When possible (in particular if using a hop count metric) the same routers may be picked as both flooding MPRs and routing MPRs.

A router selects both routing and flooding MPRs from among its one hop neighbors connected by "symmetric", i.e., bidirectional, links. Therefore, selecting routes through routing MPRs avoids the problems associated with data packet transfer over unidirectional links (e.g.,

the problem of not getting link layer acknowledgments at each hop, for link layers employing this technique).

OLSRv2 uses and extends the MANET NeighborHood Discovery Protocol (NHDP) defined in [RFC6130] and also uses the Generalized MANET Packet/Message Format [RFC5444], the TLVs specified in [RFC5497] and, optionally, message jitter as specified in [RFC5148]. These four other protocols and specifications were all originally created as part of OLSRv2, but have been specified separately for wider use.

OLSRv2 makes no assumptions about the underlying link layer. OLSRv2, through its use of [RFC6130], may use link layer information and notifications when available and applicable. In addition, OLSRv2 uses link metrics that may be derived from link layer or any other information. OLSRv2 does not specify the physical meaning of link metrics, but specifies a means by which new types of link metrics may be specified in the future, but used by OLSRv2 without modification.

OLSRv2, as OLSR [RFC3626], inherits its concept of forwarding and relaying from HIPERLAN (a MAC layer protocol) which is standardized by ETSI [HIPERLAN], [HIPERLAN2]. This document does not obsolete [RFC3626] which is left in place for further experimentation.

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].

All terms introduced in [RFC5444], including "packet", "Packet Header", "message", "Message Header", "Message Body", "Message Type", "message sequence number", "hop limit", "hop count", "Address Block", "TLV Block", "TLV", "Message TLV", "Address Block TLV", "type" (of TLV), "type extension" (of TLV), "value" (of TLV), "address", "address prefix", and "address object" are to be interpreted as described there.

All terms introduced in [RFC6130], including "interface", "MANET interface", "network address", "link", "symmetric link", "symmetric 1-hop neighbor", "symmetric 2-hop neighbor", "symmetric 1-hop neighborhood" "constant", "interface parameter", "router parameter", "Information Base", and "HELLO message" are to be interpreted as described there.

Additionally, this specification uses the following terminology:

Router:
A MANET router which implements this protocol.

OLSRv2 interface:
A MANET interface running this protocol. A router running this protocol MUST have at least one OLSRv2 interface.

Routable address:
A network address which may be used as the destination of a data packet. A router which implements this protocol will need to distinguish a routable address from a non-routable address by direct inspection of the network address, based on global scope address allocations by IANA and/or administrative configuration (consistently across the MANET). Broadcast and multicast addresses, and addresses which are limited in scope to less than the entire MANET, MUST NOT be considered as routable addresses. Anycast addresses may be considered as routable addresses.

Originator address:
An address which is unique (within the MANET) to a router. A router MUST select an originator address; it MAY choose one of its interface addresses as its originator address; it MAY select either a routable or non-routable address. A broadcast, multicast or anycast address MUST NOT be chosen as an originator address. If the router selects a routable address then this MUST be one which the router will accept as destination. An originator address MUST NOT have a prefix length, except for when included in an Address Block where it MAY be associated with a prefix of maximum prefix length (e.g., if the originator address is an IPv6 address, it MUST have either no prefix length, or have a prefix length of 128).

Message originator address:
The originator address of the router which created a message, as deduced from that message by its recipient. For all messages used in this specification, including HELLO messages defined in [RFC6130], the recipient MUST be able to deduce an originator address. The message originator address will usually be included in the message as its <msg-orig-addr> element as defined in [RFC5444]. However, an exceptional case, which does not add a <msg-orig-addr> element to a HELLO message, may be used by a router that only has a single address.

Willingness:
A numerical value between WILL_NEVER and WILL_ALWAYS (both inclusive), that represents the router's willingness to be selected as an MPR. A router has separate willingness values to be a flooding MPR and a routing MPR.

Willing symmetric 1-hop neighbor:
A symmetric 1-hop neighbor that has willingness not equal to WILL_NEVER.

Multipoint relay (MPR):
A router, X, is an MPR for a router, Y, if router Y has indicated its selection of router X as an MPR in a recent HELLO message. Router X may be a flooding MPR for Y if it is indicated to participate in the flooding process of messages received from router Y, or it may be a routing MPR for Y, if it is indicated to declare link-state information for the link from X to Y. It may also be both at the same time.

MPR selector:
A router, Y, is a flooding/routing MPR selector of router X if router Y has selected router X as a flooding/routing MPR.

MPR flooding:
The optimized MANET-wide information distribution mechanism, employed by this protocol, in which a message is relayed by only a reduced subset of the routers in the network. MPR flooding is the mechanism by which flooding reduction is achieved.

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

This specification employs the same notational conventions as in [RFC5444] and [RFC6130].

3. Applicability Statement

This document specifies OLSRv2, a proactive routing protocol intended for use in mobile ad hoc networks (MANETs) [RFC2501]. The protocol's applicability is determined by its characteristics, which are that this protocol:

- Is designed to work in networks with a dynamic topology, and in which messages may be lost, such as due to collisions over wireless media.

- Supports routers that each have one or more participating OLSRv2 interfaces, which will consist of some or all of its MANET interfaces using [RFC6130]. The set of a router's OLSRv2 interfaces, and the sets of its other MANET and non-MANET interfaces, may change over time. Each interface may have one or more network addresses (which may have prefix lengths), and these may also be dynamically changing.

- Enables hop-by-hop routing, i.e., each router can use its local information provided by this protocol to route packets.

- Continuously maintains routes to all destinations in the network, i.e., routes are instantly available and data traffic is subject to no delays due to route discovery. Consequently, no data traffic buffering is required.

- Supports routers that have non-OLSRv2 interfaces that may be local to a router or that can serve as gateways towards other networks.

- Enables the use of bidirectional additive link metrics to use shortest distance routes (i.e., routes with smallest total of link metrics). Incoming link metric values are to be determined by a process outside this specification.

- Is optimized for large and dense networks; the larger and more dense a network, the more optimization can be achieved by using MPRs, compared to the classic link state algorithm [MPR].

- Uses [RFC5444] as described in its "Intended Usage" appendix and by [RFC5498].

- Allows "external" and "internal" extensibility (adding new message types and adding information to existing messages) as enabled by [RFC5444].

- Is designed to work in a completely distributed manner, and does not depend on any central entity.

4. Protocol Overview and Functioning

The objectives of this protocol are for each router to:

- Identify all destinations in the network.

- Identify a sufficient subset of links in the network, in order that shortest routes can be calculated to all available destinations.

- Provide a Routing Set, containing these shortest routes from this router to all destinations (routable addresses and local links).

4.1. Overview

These objectives are achieved, for each router, by:

- Using [RFC6130] to identify symmetric 1-hop neighbors and symmetric 2-hop neighbors.

- Reporting its participation in OLSRv2, and its willingness to be a flooding MPR and to be a routing MPR, by extending the HELLO messages defined in [RFC6130], by the addition of an MPR_WILLING Message TLV. The router's "flooding willingness" indicates how willing it is to participate in MPR flooding. The router's "routing willingness" indicates how willing it is to be an intermediate router for routing. Note that a router is still able to be a routing source or destination, even if unwilling to perform either function.

- Extending the HELLO messages defined in [RFC6130] to allow the addition of directional link metrics to advertised links with other routers participating in OLSRv2, and to indicate which link metric type is being used by those routers. Both incoming and outgoing link metrics may be reported, the former determined by the advertising router.

- Selecting flooding MPRs and routing MPRs from among its willing symmetric 1-hop neighbors such that, for each set of MPRs, all symmetric 2-hop neighbors are reachable either directly or via at least one selected MPR, using a path of appropriate minimum total metric for at least routing MPR selection. An analysis and examples of MPR selection algorithms are given in [MPR]; a suggested algorithm, appropriately adapted for each set of MPRs, is included in Appendix B of this specification. Note that it is not necessary for routers to use the same algorithm in order to interoperate in the same MANET, but each such algorithm must have the appropriate properties, described in Section 18.

- Signaling its flooding MPR and routing MPR selections, by extending the HELLO messages defined in [RFC6130] to report this information by the addition of MPR Address Block TLV(s) associated with the appropriate network addresses.

- Extracting its flooding MPR selectors and routing MPR selectors from received HELLO messages, using the included MPR Address Block TLV(s).

- Defining a TC (Topology Control) Message Type using the message format specified in [RFC5444]. TC messages are used to periodically signal links between routing MPR selectors and itself throughout the MANET. This signaling includes suitable directional neighbor metrics (the best link metric in that direction between those routers).

- Allowing its TC messages, as well as HELLO messages, to be included in packets specified in [RFC5444], using the "manet" IP protocol or UDP port as specified in [RFC5498].

- Diffusing TC messages by using a flooding reduction mechanism, denoted "MPR flooding"; only the flooding MPRs of a router will retransmit messages received from (i.e., originated or last relayed by) that router.

Note that the indicated extensions to [RFC6130] are of forms permitted by that specification.

This specification defines:

- The requirement to use [RFC6130], its parameters, constants, HELLO messages, and Information Bases, each as extended in this specification.

- Two new Information Bases: the Topology Information Base and the Received Message Information Base.

- TC messages, which are used for MANET wide signaling (using MPR flooding) of selected topology (link state) information.

- A requirement for each router to have an originator address to be included in, or deducible from, HELLO messages and TC messages.

- The specification of new Message TLVs and Address Block TLVs that are used in HELLO messages and TC messages, including for reporting neighbor status, MPR selection, external gateways, link metrics, willingness to be an MPR, and content sequence numbers. Note that the generation of (incoming) link metric values is to be undertaken by a process outside this specification; this specification concerns only the distribution and use of those metrics.

- The generation of TC messages from the appropriate information in the Information Bases.

- The updating of the Topology Information Base according to received TC messages.

- The MPR flooding mechanism, including the inclusion of message originator address and sequence number to manage duplicate messages, using information recorded in the Received Message Information Base.

- The response to other events, such as the expiration of information in the Information Bases.

This protocol inherits the stability of a link state algorithm, and has the advantage of having routes immediately available when needed, due to its proactive nature.

This protocol only interacts with IP through routing table management, and the use of the sending IP address for IP datagrams containing messages used by this specification.

4.2. Routers and Interfaces

In order for a router to participate in a MANET using this protocol it must have at least one, and possibly more, OLSRv2 interfaces. Each OLSRv2 interface:

- Is a MANET interface, as specified in [RFC6130]. In particular it must be configured with one or more network addresses; these addresses must each be specific to this router, and must include any address that will be used as the sending address of any IP packet sent on this OLSRv2 interface.

- Has a number of interface parameters, adding to those specified in [RFC6130].

- Has an Interface Information Base, extending that specified in [RFC6130].

- Generates and processes HELLO messages according to [RFC6130], extended as specified in Section 15.

In addition to a set of OLSRv2 interfaces as described above, each router:

- May have one or more non-OLSRv2 interfaces (which may include MANET interfaces and/or non-MANET interfaces) and/or local attached networks for which this router can accept IP packets. All routable addresses for which the router is to accept IP packets must be used as an (OLSRv2 or non-OLSRv2) interface network address, or as an address of a local attached network of the router.

- Has a number of router parameters, adding to those specified in [RFC6130].

- Has a Local Information Base, extending that specified in [RFC6130], including selection of an originator address and

recording any locally attached networks.

- Has a Neighbor Information Base, extending that specified in [RFC6130] to record MPR selection and advertisement information.

- Has a Topology Information Base, recording information received in TC messages.

- Has a Received Message Information Base, recording information about received messages to ensure that each TC message is only processed once, and forwarded at most once on each OLSRv2 interface, by a router.

- Generates, receives, and processes TC messages.

4.3. Information Base Overview

Each router maintains the Information Bases described in the following sections. These are used for describing the protocol in this specification. An implementation of this protocol may maintain this information in the indicated form, or in any other organization which offers access to this information. In particular, note that it is not necessary to remove Tuples from Sets at the exact time indicated, only to behave as if the Tuples were removed at that time.

4.3.1. Local Information Base

The Local Information Base is specified in [RFC6130], and contains a router's local configuration. It is extended in this specification to also record an originator address, and to include a router's:

- Originator Set, containing addresses that were recently used as this router's originator address, and is used, together with the router's current originator address, to enable a router to recognize and discard control traffic which was originated by the router itself.

- Local Attached Network Set, containing network addresses of networks to which this router can act as a gateway, and advertises in its TC messages.

4.3.2. Interface Information Bases

The Interface Information Base for each OLSRv2 interface is as specified in [RFC6130], extended to also record, in each Link Set, link metric values (incoming and outgoing) and flooding MPR selector information.

4.3.3. Neighbor Information Base

The Neighbor Information Base is specified in [RFC6130], and is extended to also record, in the Neighbor Tuple for each neighbor:

- Its originator address.
- Neighbor metric values, these being the minimum of the link metric values in the indicated direction for all symmetric 1-hop links with that neighbor.
- Its willingness to be a flooding MPR and to be a routing MPR.
- Whether it has been selected by this router as a flooding MPR or as a routing MPR, and whether it is a routing MPR selector of this router. (Whether it is a flooding MPR selector of this neighbor is recorded in the Interface Information Base.)
- Whether it is to be advertised in TC messages sent by this router.

4.3.4. Topology Information Base

The Topology Information Base in each router contains:

- An Advertising Remote Router Set, recording each remote router from which TC messages have been received. This is used in order to determine if a received TC message contains fresh or outdated information; a received TC message is ignored in the latter case.
- A Router Topology Set, recording links between routers in the MANET, as described by received TC messages.
- A Routable Address Topology Set, recording routable addresses in the MANET (available as IP packet destinations) and from which remote router these routable addresses can be directly reached (i.e., in a single IP hop from that remote router), as reported by received TC messages.
- An Attached Network Set, recording networks to which a remote router has advertised that it may act as a gateway. These networks may be reached in one or more IP hops from that remote router.
- A Routing Set, recording routes from this router to all available destinations. The IP routing table is to be updated using this Routing Set. (A router may choose to use any or all destination network addresses in the Routing Set to update the IP routing table, this selection is outside the scope of this specification.)

The purpose of the Topology Information Base is to record information used, in addition to that in the Local Information Base, the Interface Information Bases and the Neighbor Information Base, to construct the Routing Set (which is also included in the Topology Information Base).

This specification describes the calculation of the Routing Set based on a Topology Graph constructed in two phases. First, a "backbone" graph representing the routers in the MANET, and the connectivity between them, is constructed from the Local Information Base, the Neighbor Information Base and the Router Topology Set. Second, this graph is "decorated" with additional destination network addresses using the Local Information Base, the Routable Address Topology Set and the Attached Network Set.

The Topology Graph does not need to be recorded in the Topology Information Base, it can either be constructed as required when the Routing Set is to be changed, or need not be explicitly constructed (as illustrated in Appendix C). An implementation may however construct and retain the Topology Graph if preferred.

4.3.5. Received Message Information Base

The Received Message Information Base in each router contains:

- A Received Set for each OLSRv2 interface, describing TC messages received by this router on that OLSRv2 interface.

- A Processed Set, describing TC messages processed by this router.

- A Forwarded Set, describing TC messages forwarded by this router.

The Received Message Information Base serves the MPR flooding mechanism by ensuring that received messages are forwarded at most once by a router, and also ensures that received messages are processed exactly once by a router. The Received Message Information Base may also record information about other Message Types that use the MPR flooding mechanism.

4.4. Signaling Overview

This protocol generates and processes HELLO messages according to [RFC6130]. HELLO messages transmitted on OLSRv2 interfaces are extended according to Section 15 of this specification to include an originator address, link metrics, and MPR selection information.

This specification defines a single Message Type, the TC message. TC messages are sent by their originating router proactively, at a

regular interval, on all OLSRv2 interfaces. This interval may be fixed, or may be dynamic, for example it may be backed off due to congestion or network stability. TC messages may also be sent as a response to a change in the router itself, or its advertised symmetric 1-hop neighborhood, for example on first being selected as a routing MPR.

Because TC messages are sent periodically, this protocol is tolerant of unreliable transmissions of TC messages. Message losses may occur more frequently in wireless networks due to collisions or other transmission problems. This protocol may use "jitter", randomized adjustments to message transmission times, to reduce the incidence of collisions, as specified in [RFC5148].

This protocol is tolerant of out of sequence delivery of TC messages due to in transit message reordering. Each router maintains an Advertised Neighbor Sequence Number (ANSN) that is incremented when its recorded neighbor information that is to be included in its TC messages changes. This ANSN is included in the router's TC messages. The recipient of a TC message can use this included ANSN to identify which of the information it has received is most recent, even if messages have been reordered while in transit. Only the most recent information received is used, older information received later is discarded.

TC messages may be "complete" or "incomplete". A complete TC message advertises all of the originating router's routing MPR selectors, it may also advertise other symmetric 1-hop neighbors. Complete TC messages are generated periodically (and also, optionally, in response to symmetric 1-hop neighborhood changes). Incomplete TC messages may be used to report additions to advertised information, without repeating unchanged information.

TC messages, and HELLO messages as extended by this specification, define (by inclusion, or by deduction when having a single address) an originator address for the router that created the message. A TC message reports both the originator addresses and routable addresses of its advertised neighbors, distinguishing the two using an Address Block TLV (an address may be both routable and an originator address). TC messages also report the originator's locally attached networks.

TC messages are MPR flooded throughout the MANET. A router retransmits a TC message received on an OLSRv2 interface if and only if the message did not originate at this router and has not been previously forwarded by this router, this is the first time the message has been received on this OLSRv2 interface, and the message is received from (i.e., originated from or was last relayed by) one

of this router's flooding MPR selectors.

Some TC messages may be MPR flooded over only part of the network, e.g., allowing a router to ensure that nearer routers are kept more up to date than distant routers, such as is used in Fisheye State Routing [FSR] and Fuzzy Sighted Link State routing [FSLS]. This is enabled using [RFC5497].

TC messages include outgoing neighbor metrics that will be used in the selection of routes.

4.5. Link Metrics

OLSRv1 [RFC3626] created minimum hop routes to destinations. However in many, if not most, circumstances, better routes (in terms of quality of service for end users) can be created by use of link metrics.

OLSRv2, as defined in this specification, supports metric-based routing, i.e., it allows links to each have a chosen metric. Link metrics as defined in OLSRv2 are additive, and the routes that are to be created are those with the minimum sum of the link metrics along that route.

Link metrics are defined to be directional; the link metric from one router to another may be different from that on the reverse link. The link metric is assessed at the receiver, as on a (typically) wireless link, that is the better informed as to link information. Both incoming and outgoing link information is used by OLSRv2, the distinctions in this specification must be clearly followed.

This specification also defines both incoming and outgoing neighbor metrics for each symmetric 1-hop neighbor, these being the minimum value of the link metrics in the same direction for all symmetric links with that neighbor. Note that this means that all neighbor metric values are link metric values and that specification of, for example, link metric value encoding also includes encoding of neighbor metric values.

This specification does not define the nature of the link metric. However, this specification allows, through use of the type extension of a defined Address Block TLV, for link metrics with specific meanings to be defined and either allocated by IANA or privately used. Each HELLO or TC message carrying link (or neighbor) metrics thus indicates which link metric information it is carrying, allowing routers to determine if they can interoperate. If link metrics require additional signaling to determine their values, whether in HELLO messages or otherwise, then this is permitted but is outside

the scope of this specification.

Careful consideration should be given to how to use link metrics. In particular it is advisable to not simply default to use of all links with equal metrics (i.e., hop count) for routing without careful consideration of whether that is appropriate or not.

4.6. Flooding MPRs and Routing MPR

This specification uses two sets of MPRs: flooding MPRs and routing MPRs. These are selected separately, because:

- Flooding MPRs may use metrics, routing MPRs must use metrics.
- When flooding MPRs use metrics, these are outgoing link metrics, routing MPRs use incoming neighbor metrics.
- Flooding MPRs need not be selected per OLSRv2 interface, routing MPRs must be selected per OLSRv2 interface.

4.7. Routing Set Use

The purpose of the Routing Set is to determine and record routes (local interface network address and next hop interface network address) to all possible routable addresses advertised by this protocol, as well as of all destinations that are local, i.e., within one hop, to the router (whether using routable addresses or not). Only symmetric links are used in such routes.

It is intended that the Routing Set can be used for IP packet routing, by using its contents to update the IP routing table. That update, and whether any Routing Tuples are not used in the IP routing table, is outside the scope of this specification.

The signaling in this specification has been designed so that a "backbone" Topology Graph of routers, each identified by its originator address, with at most one direct connection between any pair of routers, can be constructed (from the Neighbor Set and the Router Topology Set) using a suitable minimum path length algorithm. This Topology Graph can then have other network addresses (routable, or of symmetric 1-hop neighbors) added to it (using the Interface Information Bases, the Routable Address Topology Set and the Attached Network Set).

5. Protocol Parameters and Constants

The parameters and constants used in this specification are those defined in [RFC6130] plus those defined in this section. The

separation in [RFC6130] into interface parameters, router parameters and constants is also used in this specification.

As for the parameters in [RFC6130], parameters defined in this specification MAY be changed dynamically by a router, and need not be the same on different routers, even in the same MANET, or, for interface parameters, on different interfaces of the same router.

5.1. Protocol and Port Numbers

This protocol specifies TC messages, which are included in packets as defined by [RFC5444]. These packets MUST be sent either using the "manet" protocol number or the "manet" UDP well-known port number, as specified in [RFC5498].

TC messages and HELLO messages [RFC6130] MUST, in a given MANET, either both be using IP or both be using UDP, in order that it is possible to combine messages of both protocols into the same [RFC5444] packet for transmission.

5.2. Multicast Address

This protocol specifies TC messages, which are included in packets as defined by [RFC5444]. These packets MAY be transmitted using the Link-Local multicast address "LL-MANET-Routers", as specified in [RFC5498].

5.3. Interface Parameters

A single additional interface parameter is specified for OLSRv2 interfaces only.

5.3.1. Received Message Validity Time

The following parameter manages the validity time of recorded received message information:

RX_HOLD_TIME:
: The period after receipt of a message by the appropriate OLSRv2 interface of this router for which that information is recorded, in order that the message is recognized as having been previously received on this OLSRv2 interface.

The following constraints apply to this parameter:

- RX_HOLD_TIME > 0

- RX_HOLD_TIME SHOULD be greater than the maximum difference in time that a message may take to traverse the MANET, taking into account any message forwarding jitter as well as propagation, queuing, and processing delays.

5.4. Router Parameters

The following router parameters are specified for routers.

5.4.1. Local History Times

The following router parameter manages the time for which local information is retained:

O_HOLD_TIME:
: The time for which a recently used and replaced originator address is used to recognize the router's own messages.

The following constraint applies to this parameter:

- O_HOLD_TIME > 0

5.4.2. Link Metric Parameters

All routes found using this specification use a single link metric type that is specified by the router parameter LINK_METRIC_TYPE, which may take any value from 0 to 255, both inclusive.

5.4.3. Message Intervals

The following parameters regulate TC message transmissions by a router. TC messages are usually sent periodically, but MAY also be sent in response to changes in the router's Neighbor Set and/or Local Attached Network Set. In a highly dynamic network, and with a larger value of the parameter TC_INTERVAL and a smaller value of the parameter TC_MIN_INTERVAL, TC messages MAY be transmitted more often in response to changes than periodically. However, because a router has no knowledge of, for example, routers remote to it (i.e., beyond two hops away) joining the network, TC messages MUST NOT be sent purely responsively.

TC_INTERVAL:
: The maximum time between the transmission of two successive TC messages by this router. When no TC messages are sent in response to local network changes (by design, or because the local network is not changing) then TC messages MUST be sent at a regular interval TC_INTERVAL, possibly modified by jitter as specified in [RFC5148].

TC_MIN_INTERVAL:
The minimum interval between transmission of two successive TC messages by this router. (This minimum interval MAY be modified by jitter, as specified in [RFC5148].)

The following constraints apply to these parameters:

- TC_INTERVAL > 0
- 0 <= TC_MIN_INTERVAL <= TC_INTERVAL
- If TLVs with Type = INTERVAL_TIME, as defined in [RFC5497], are included in TC messages, then TC_INTERVAL MUST be representable by way of the exponent-mantissa notation described in Section 5 of [RFC5497].

5.4.4. Advertised Information Validity Times

The following parameters manage the validity time of information advertised in TC messages:

T_HOLD_TIME:
Used as the minimum value in the TLV with Type = VALIDITY_TIME included in all TC messages sent by this router. If a single value of parameter TC_HOP_LIMIT (see Section 5.4.7) is used then this will be the only value in that TLV.

A_HOLD_TIME:
The period during which TC messages are sent after they no longer have any advertised information to report, but are sent in order to accelerate outdated information removal by other routers.

The following constraints apply to these parameters:

- T_HOLD_TIME > 0
- A_HOLD_TIME >= 0
- T_HOLD_TIME >= TC_INTERVAL
- If TC messages can be lost, then both T_HOLD_TIME and A_HOLD_TIME SHOULD be significantly greater than TC_INTERVAL; a value >= 3 x TC_INTERVAL is RECOMMENDED.
- T_HOLD_TIME MUST be representable by way of the exponent-mantissa notation described in Section 5 of [RFC5497].

5.4.5. Processing and Forwarding Validity Times

The following parameters manage the processing and forwarding validity time of recorded message information:

P_HOLD_TIME:
: The period after receipt of a message that is processed by this router for which that information is recorded, in order that the message is not processed again if received again.

F_HOLD_TIME:
: The period after receipt of a message that is forwarded by this router for which that information is recorded, in order that the message is not forwarded again if received again.

The following constraints apply to these parameters:

- P_HOLD_TIME > 0
- F_HOLD_TIME > 0
- Both of these parameters SHOULD be greater than the maximum difference in time that a message may take to traverse the MANET, taking into account any message forwarding jitter as well as propagation, queuing, and processing delays.

5.4.6. Jitter

If jitter, as defined in [RFC5148], is used then the governing jitter parameters are as follows:

TP_MAXJITTER:
: Represents the value of MAXJITTER used in [RFC5148] for periodically generated TC messages sent by this router.

TT_MAXJITTER:
: Represents the value of MAXJITTER used in [RFC5148] for externally triggered TC messages sent by this router.

F_MAXJITTER:
: Represents the default value of MAXJITTER used in [RFC5148] for messages forwarded by this router. However, before using F_MAXJITTER a router MAY attempt to deduce a more appropriate value of MAXJITTER, for example based on any TLVs with Type = INTERVAL_TIME or Type = VALIDITY_TIME contained in the message to be forwarded.

For constraints on these parameters see [RFC5148].

5.4.7. Hop Limit

The parameter TC_HOP_LIMIT is the hop limit set in each TC message. TC_HOP_LIMIT MAY be a single fixed value, or MAY be different in TC messages sent by the same router. However each other router, at any hop count distance, MUST see a regular pattern of TC messages in order that meaningful values of TLVs with Type = INTERVAL_TIME and Type = VALIDITY_TIME at each hop count distance can be included as defined in [RFC5497]. Thus the pattern of TC_HOP_LIMIT MUST be defined to have this property. For example the repeating pattern (255 4 4) satisfies this property (having period TC_INTERVAL at hop counts up to 4, inclusive, and 3 x TC_INTERVAL at hop counts greater than 4), but the repeating pattern (255 255 4 4) does not satisfy this property because at hop counts greater than 4, message intervals are alternately TC_INTERVAL and 3 x TC_INTERVAL.

The following constraints apply to this parameter:

- The maximum value of TC_HOP_LIMIT >= the network diameter in hops, a value of 255 is RECOMMENDED. Note that if using a pattern of different values of TC_HOP_LIMIT as described above, then only the maximum value in the pattern is so constrained.

- All values of TC_HOP_LIMIT >= 2.

5.4.8. Willingness

Each router has two willingness parameters: WILL_FLOODING and WILL_ROUTING, each of which MUST be in the range WILL_NEVER to WILL_ALWAYS, inclusive.

WILL_FLOODING represents the router's willingness to be selected as a flooding MPR and hence to participate in MPR flooding, in particular of TC messages.

WILL_ROUTING represents the router's willingness to be selected as a routing MPR and hence to be an intermediate router on routes.

In either case, the higher the value, the greater the router's willingness to be a flooding or routing MPR, respectively. If a router has a willingness value of WILL_NEVER (the lowest possible value) it does not perform the corresponding task. A MANET using this protocol with too many routers having either willingness value equal to WILL_NEVER will not function; it MUST be ensured, by administrative or other means, that this does not happen.

Note that how many routers having either willingness value equal to WILL_NEVER is "too many" depends on the network topology - which, in

a MANET may change dynamically. Willingness is intended to enable that certain routers (e.g., routers that have generous resources, such as a permanent power supply) can be configured to assume more of the network operation, while others (e.g., routers that have lesser resources, such as are battery operated) can avoid such tasks. A general guideline would be that only if a router is not actually able to assume the task (flooding or routing) should it be configured with the corresponding willingness WILL_NEVER.

If a router has a willingness value equal to WILL_ALWAYS (the highest possible value) then it will always be selected as a flooding or routing MPR, respectively, by all symmetric 1-hop neighbors.

In a MANET in which all routers have WILL_FLOODING = WILL_ALWAYS, flooding reduction will effectively be disabled, and flooding will perform as blind flooding.

In a MANET in which all routers have WILL_ROUTING = WILL_ALWAYS, topology reduction will effectively be disabled, and all routers will advertise all of their links in TC messages.

A router that has WILL_ROUTING = WILL_NEVER will not act as an intermediate router in the MANET. Such a router can, act as a source, destination or gateway to another routing domain.

Different routers MAY have different values for WILL_FLOODING and/or WILL_ROUTING.

The following constraints apply to these parameters:

- WILL_NEVER <= WILL_FLOODING <= WILL_ALWAYS

- WILL_NEVER <= WILL_ROUTING <= WILL_ALWAYS

5.5. Parameter Change Constraints

If protocol parameters are changed dynamically, then the constraints in this section apply.

RX_HOLD_TIME

  * If RX_HOLD_TIME for an OLSRv2 interface changes, then the expiry time for all Received Tuples for that OLSRv2 interface MAY be changed.

O_HOLD_TIME

* If O_HOLD_TIME changes, then the expiry time for all Originator Tuples MAY be changed.

TC_INTERVAL

* If TC_INTERVAL increases, then the next TC message generated by this router MUST be generated according to the previous, shorter, TC_INTERVAL. Additional subsequent TC messages MAY be generated according to the previous, shorter, TC_INTERVAL.

* If TC_INTERVAL decreases, then the following TC messages from this router MUST be generated according to the current, shorter, TC_INTERVAL.

P_HOLD_TIME

* If P_HOLD_TIME changes, then the expiry time for all Processed Tuples MAY be changed.

F_HOLD_TIME

* If F_HOLD_TIME changes, then the expiry time for all Forwarded Tuples MAY be changed.

TP_MAXJITTER

* If TP_MAXJITTER changes, then the periodic TC message schedule on this router MAY be changed immediately.

TT_MAXJITTER

* If TT_MAXJITTER changes, then externally triggered TC messages on this router MAY be rescheduled.

F_MAXJITTER

* If F_MAXJITTER changes, then TC messages waiting to be forwarded with a delay based on this parameter MAY be rescheduled.

TC_HOP_LIMIT

* If TC_HOP_LIMIT changes, and the router uses multiple values after the change, then message intervals and validity times included in TC messages MUST be respected. The simplest way to do this is to start any new repeating pattern of TC_HOP_LIMIT

values with its largest value.

LINK_METRIC_TYPE

* If LINK_METRIC_TYPE changes, then all link metric information recorded by the router is invalid. The router MUST take the following actions, and all consequent actions described in Section 17 and [RFC6130].

  + For each Link Tuple in any Link Set for an OLSRv2 interface, either update L_in_metric (the value MAXIMUM_METRIC MAY be used) or remove the Link Tuple from the Link Set.

  + For each Link Tuple that is not removed, set:

    - L_out_metric := UNKNOWN_METRIC;

    - L_SYM_time := EXPIRED;

    - L_MPR_selector := false.

  + Remove all Router Topology Tuples, Routable Address Topology Tuples, Attached Network Tuples and Routing Tuples from their respective Protocol Sets in the Topology Information Base.

5.6. Constants

The following constants are specified for routers. Unlike router parameters, constants MUST NOT change, and MUST be the same on all routers.

5.6.1. Link Metric Constants

The constant minimum and maximum link metric values are defined by:

o MINIMUM_METRIC := 1

o MAXIMUM_METRIC := 16776960

The symbolic value UNKNOWN_METRIC is defined in Section 6.1.

5.6.2. Willingness Constants

The constant minimum, RECOMMENDED default, and maximum, willingness values are defined by:

- WILL_NEVER := 0
- WILL_DEFAULT := 7
- WILL_ALWAYS := 15

5.6.3. Time Constant

The constant C (time granularity) is used as specified in [RFC5497]. It MUST be the same as is used by [RFC6130], with RECOMMENDED value:

- C := 1/1024 second

Note that this parameter is used in the representation of time intervals. Time values (such as are stored in Protocol Tuples) are not so represented. A resolution of C in such values is sufficient (but not necessary) for such values.

6. Link Metric Values

A router records a link metric value for each direction of a link of which it has knowledge. These link metric values are used to create metrics for routes by the addition of link metric values.

6.1. Link Metric Representation

Link metrics are reported in messages using a compressed representation that occupies 12 bits, consisting of a 4 bit field and an 8 bit field. The compressed representation represents positive integer values with a minimum value of 1 and a maximum value that is slightly smaller than the maximum 24 bit value. Only those values that have exact representation in the compressed form are used. Route metrics are the summation of no more than 255 link metric values, and can therefore be represented using no more than 32 bits.

Link and route metrics used in the Information Bases defined in this specification refer to the uncompressed values, and arithmetic involving them does likewise, and assumes full precision in the result. (How an implementation records the values is not part of this specification, as long as it behaves as if recording uncompressed values. An implementation can, for example, use 32 bit values for all link and route metrics.)

In some cases a link metric value may be unknown. This is indicated in this specification by the symbolic value UNKNOWN_METRIC. An implementation may use any representation of UNKNOWN_METRIC as it is never included in messages, or used in any computation. (Possible representations are zero, or any value greater than the maximum

representable metric value.)

6.2. Link Metric Compressed Form

The 12-bit compressed form of a link metric uses a modified form of a representation with an 8-bit mantissa (denoted a) and a 4-bit exponent (denoted b). Note that if represented as the 12 bit value 256b+a then the ordering of those 12 bit values is identical to the ordering of the represented values.

The value so represented is (257+a)2^b - 256, where ^ denotes exponentiation. This has a minimum value (when a = 0 and b = 0) of MINIMUM_METRIC = 1 and a maximum value (when a = 255 and b = 15) of MAXIMUM_METRIC = 2^24 - 256.

An algorithm for computing a and b for the smallest representable value not less than a link metric value v such that MINIMUM_METRIC <= v <= MAXIMUM_METRIC is:

1. Find the smallest integer b such that v + 256 <= 2^(b + 9).

2. Set a := (v - 256(2^b - 1)) / (2^b) - 1, rounded up to the nearest integer.

7. Local Information Base

The Local Information Base, as defined for each router in [RFC6130], is extended by this protocol by:

- Recording the router's originator address. The originator address MUST be unique to this router. It MUST NOT be used by any other router as an originator address. It MAY be included in any network address in any I_local_iface_addr_list of this router, it MUST NOT be included in any network address in any I_local_iface_addr_list of any other router. It MAY be included in, but MUST NOT be equal to, the AL_net_addr in any Local Attached Network Tuple in this or any other router.

- The addition of an Originator Set, defined in Section 7.1, and a Local Attached Network Set, defined in Section 7.2.

All routable addresses of the router for which it is to accept IP packets as destination MUST be included in the Local Interface Set or the Local Attached Network Set.

7.1. Originator Set

A router's Originator Set records addresses that were recently used as originator addresses by this router. If a router's originator address is immutable then the Originator Set is always empty and MAY be omitted. It consists of Originator Tuples:

(O_orig_addr, O_time)

where:

O_orig_addr is a recently used originator address; note that this does not include a prefix length.

O_time specifies the time at which this Tuple expires and MUST be removed.

7.2. Local Attached Network Set

A router's Local Attached Network Set records its local non-OLSRv2 interfaces via which it can act as a gateway to other networks. The Local Attached Network Set MUST be provided to this protocol and MUST reflect any changes in the router's status. (In cases where the router's configuration is static, the Local Attached Network Set will be constant; in cases where the router has no non-OLSRv2 interfaces, the Local Attached Network Set will be empty.) The Local Attached Network Set is not modified by this protocol. This protocol will respond to (externally provided) changes to the Local Attached Network Set. It consists of Local Attached Network Tuples:

(AL_net_addr, AL_dist, AL_metric)

where:

AL_net_addr is the network address of an attached network which can be reached via this router. This SHOULD be a routable address. It is constrained as described below.

AL_dist is the number of hops to the network with network address AL_net_addr from this router.

AL_metric is the metric of the link to the attached network with address AL_net_addr from this router.

Attached networks local to this router only (i.e., not reachable except via this router) SHOULD be treated as local non-MANET interfaces, and added to the Local Interface Set, as specified in [RFC6130], rather than be added to the Local Attached Network Set.

Because an attached network is not specific to the router, and may be outside the MANET, an attached network MAY also be attached to other routers. Routing to an AL_net_addr will use maximum prefix length matching; consequently an AL_net_addr MAY include, but MUST NOT equal or be included in, any network address which is of any interface of any router (i.e., is included in any I_local_iface_addr_list) or equal any router's originator address.

It is not the responsibility of this protocol to maintain routes from this router to networks recorded in the Local Attached Network Set.

Local Attached Neighbor Tuples are removed from the Local Attached Network Set only when the router's local attached network configuration changes, i.e., they are not subject to timer-based expiration or changes due to received messages.

8. Interface Information Base

An Interface Information Base, as defined in [RFC6130], is maintained for each MANET interface. The Link Set and 2-Hop Set in the Interface Information Base for an OLSRv2 interface are modified by this protocol. In some cases it may be convenient to consider these Sets as also containing these additional elements for other MANET interfaces, taking the indicated values on creation, but never being updated.

8.1. Link Set

The Link Set is modified by adding these additional elements to each Link Tuple:

L_in_metric is the metric of the link from the OLSRv2 interface with addresses L_neighbor_iface_addr_list to this OLSRv2 interface;

L_out_metric is the metric of the link to the OLSRv2 interface with addresses L_neighbor_iface_addr_list from this OLSRv2 interface;

L_mpr_selector is a boolean flag, describing if this neighbor has selected this router as a flooding MPR, i.e., is a flooding MPR selector of this router.

L_in_metric will be specified by a process that is external to this specification. Any Link Tuple with L_status = HEARD or L_status = SYMMETRIC MUST have a specified value of L_in_metric if it is to be used by this protocol.

A Link Tuple created (but not updated) by [RFC6130] MUST set:

- L_in_metric := UNKNOWN_METRIC;
- L_out_metric := UNKNOWN_METRIC;
- L_mpr_selector := false.

8.2. 2-Hop Set

The 2-Hop Set is modified by adding these additional elements to each 2-Hop Tuple:

N2_in_metric is the neighbor metric from the router with address N2_2hop_iface_addr to the router with OLSRv2 interface addresses N2_neighbor_iface_addr_list;

N2_out_metric is the neighbor metric to the router with address N2_2hop_iface_addr from the router with OLSRv2 interface addresses N2_neighbor_iface_addr_list.

A 2-Hop Tuple created (but not updated) by [RFC6130] MUST set:

- N2_in_metric := UNKNOWN_METRIC;
- N2_out_metric := UNKNOWN_METRIC.

9. Neighbor Information Base

A Neighbor Information Base, as defined in [RFC6130], is maintained for each router. It is modified by this protocol by adding these additional elements to each Neighbor Tuple in the Neighbor Set. In some cases it may be convenient to consider these Sets as also containing these additional elements for other MANET interfaces, taking the indicated values on creation, but never being updated.

N_orig_addr is the neighbor's originator address, which may be unknown. Note that this originator address does not include a prefix length;

N_in_metric is the neighbor metric of any link from this neighbor to an OLSRv2 interface of this router, i.e., the minimum of all corresponding L_in_metric with L_status = SYMMETRIC and L_in_metric != UNKNOWN_METRIC, UNKNOWN_METRIC if there are no such Link Tuples;

N_out_metric is the neighbor metric of any link from an OLSRv2 interface of this router to this neighbor, i.e., the minimum of

all corresponding L_out_metric with L_status = SYMMETRIC and L_out_metric != UNKNOWN_METRIC, UNKNOWN_METRIC if there are no such Link Tuples;

N_will_flooding is the neighbor's willingness to be selected as a flooding MPR, in the range from WILL_NEVER to WILL_ALWAYS, both inclusive, taking the value WILL_NEVER if no OLSRv2 specific information is received from this neighbor;

N_will_routing is the neighbor's willingness to be selected as a routing MPR, in the range from WILL_NEVER to WILL_ALWAYS, both inclusive, taking the value WILL_NEVER if no OLSRv2 specific information is received from this neighbor;

N_flooding_mpr is a boolean flag, describing if this neighbor is selected as a flooding MPR by this router;

N_routing_mpr is a boolean flag, describing if this neighbor is selected as a routing MPR by this router;

N_mpr_selector is a boolean flag, describing if this neighbor has selected this router as a routing MPR, i.e., is a routing MPR selector of this router.

N_advertised is a boolean flag, describing if this router has elected to advertise a link to this neighbor in its TC messages.

A Neighbor Tuple created (but not updated) by [RFC6130] MUST set:

- N_orig_addr := unknown;
- N_in_metric := UNKNOWN_METRIC;
- N_out_metric := UNKNOWN_METRIC;
- N_will_flooding := WILL_NEVER;
- N_will_routing := WILL_NEVER;
- N_routing_mpr := false;
- N_flooding_mpr := false;
- N_mpr_selector := false;
- N_advertised := false.

The Neighbor Information Base also includes a variable, the

Advertised Neighbor Sequence Number (ANSN), whose value is included in TC messages to indicate the freshness of the information transmitted. The ANSN is incremented whenever advertised information (the originator and routable addresses included in Neighbor Tuples with N_advertised = true, and local attached networks recorded in the Local Attached Network Set in the Local Information Base) changes, including addition or removal of such information.

10. Topology Information Base

The Topology Information Base, defined for each router by this specification, stores information received in TC messages, in the Advertising Remote Router Set, the Router Topology Set, the Routable Address Topology Set and the Attached Network Set.

Additionally, a Routing Set is maintained, derived from the information recorded in the Local Information Base, the Interface Information Bases, the Neighbor Information Base and the rest of the Topology Information Base.

10.1. Advertising Remote Router Set

A router's Advertising Remote Router Set records information describing each remote router in the network that transmits TC messages, allowing outdated TC messages to be recognized and discarded. It consists of Advertising Remote Router Tuples:

(AR_orig_addr, AR_seq_number, AR_time)

where:

AR_orig_addr is the originator address of a received TC message, note that this does not include a prefix length;

AR_seq_number is the greatest ANSN in any TC message received which originated from the router with originator address AR_orig_addr (i.e., which contributed to the information contained in this Tuple);

AR_time is the time at which this Tuple expires and MUST be removed.

10.2. Router Topology Set

A router's Topology Set records topology information about the links between routers in the MANET. It consists of Router Topology Tuples:

(TR_from_orig_addr, TR_to_orig_addr, TR_seq_number, TR_metric,

TR_time)

where:

TR_from_orig_addr is the originator address of a router which can reach the router with originator address TR_to_orig_addr in one hop, note that this does not include a prefix length;

TR_to_orig_addr is the originator address of a router which can be reached by the router with originator address TR_to_orig_addr in one hop, note that this does not include a prefix length;

TR_seq_number is the greatest ANSN in any TC message received which originated from the router with originator address TR_from_orig_addr (i.e., which contributed to the information contained in this Tuple);

TR_metric is the neighbor metric from the router with originator address TR_from_orig_addr to the router with originator address TR_to_orig_addr;

TR_time specifies the time at which this Tuple expires and MUST be removed.

10.3. Routable Address Topology Set

A router's Routable Address Topology Set records topology information about the routable addresses within the MANET, and via which routers they may be reached. It consists of Routable Address Topology Tuples:

(TA_from_orig_addr, TA_dest_addr, TA_seq_number, TA_metric, TA_time)

where:

TA_from_orig_addr is the originator address of a router which can reach the router with routable address TA_dest_addr in one hop, note that this does not include a prefix length;

TA_dest_addr is a routable address of a router which can be reached by the router with originator address TA_from_orig_addr in one hop;

TA_seq_number is the greatest ANSN in any TC message received which originated from the router with originator address TA_from_orig_addr (i.e., which contributed to the information contained in this Tuple);

TA_metric is the neighbor metric from the router with originator address TA_from_orig_addr to the router with OLSRv2 interface address TA_dest_addr;

TA_time specifies the time at which this Tuple expires and MUST be removed.

10.4. Attached Network Set

A router's Attached Network Set records information about networks (which may be outside the MANET) attached to other routers and their routable addresses. It consists of Attached Network Tuples:

(AN_orig_addr, AN_net_addr, AN_seq_number, AN_dist, AN_metric, AN_time)

where:

AN_orig_addr is the originator address of a router which can act as gateway to the network with network address AN_net_addr, note that this does not include a prefix length;

AN_net_addr is the network address of an attached network, which may be reached via the router with originator address AN_orig_addr;

AN_seq_number is the greatest ANSN in any TC message received which originated from the router with originator address AN_orig_addr (i.e., which contributed to the information contained in this Tuple);

AN_dist is the number of hops to the network with network address AN_net_addr from the router with originator address AN_orig_addr;

AN_metric is the metric of the link from the router with originator address AN_orig_addr to the attached network with address AN_net_addr;

AN_time specifies the time at which this Tuple expires and MUST be removed.

10.5. Routing Set

A router's Routing Set records the first hop along a selected path to each destination for which any such path is known. It consists of Routing Tuples:

(R_dest_addr, R_next_iface_addr, R_local_iface_addr, R_dist,

R_metric)

where:

R_dest_addr is the network address of the destination, either the network address of an interface of a destination router, or the network address of an attached network;

R_next_iface_addr is the network address of the "next hop" on the selected path to the destination;

R_local_iface_addr is an address of the local interface over which an IP packet MUST be sent to reach the destination by the selected path.

R_dist is the number of hops on the selected path to the destination;

R_metric is the metric of the route to the destination with address R_dest_addr.

The Routing Set for a router is derived from the contents of other Protocol Sets of the router (the Link Sets, the Neighbor Set, the Router Topology Set, the Routable Address Topology Set, the Attached Network Set, and OPTIONAL use of the 2-Hop Sets). The Routing Set is updated (Routing Tuples added or removed, or the complete Routing Set recalculated) when routing paths are calculated, based on changes to these other Protocol Sets. Routing Tuples are not subject to timer-based expiration.

11. Received Message Information Base

The Received Message Information Base, defined by this specification, records information required to ensure that a message is processed at most once and is forwarded at most once per OLSRv2 interface of a router, using MPR flooding. Messages are recorded using their "signature", consisting of their type, originator address, and message sequence number.

11.1. Received Set

A router has a Received Set per OLSRv2 interface. Each Received Set records the signatures of messages which have been received over that OLSRv2 interface. Each consists of Received Tuples:

(RX_type, RX_orig_addr, RX_seq_number, RX_time)

where:

RX_type is the received Message Type;

RX_orig_addr is the originator address of the received message, note that this does not include a prefix length;

RX_seq_number is the message sequence number of the received message;

RX_time specifies the time at which this Tuple expires and MUST be removed.

11.2. Processed Set

A router has a single Processed Set which records signatures of messages which have been processed by the router. It consists of Processed Tuples:

(P_type, P_orig_addr, P_seq_number, P_time)

where:

P_type is the processed Message Type;

P_orig_addr is the originator address of the processed message, note that this does not include a prefix length;

P_seq_number is the message sequence number of the processed message;

P_time specifies the time at which this Tuple expires and MUST be removed.

11.3. Forwarded Set

A router has a single Forwarded Set which records signatures of messages which have been forwarded by the router. It consists of Forwarded Tuples:

(F_type, F_orig_addr, F_seq_number, F_time)

where:

F_type is the forwarded Message Type;

F_orig_addr is the originator address of the forwarded message, note that this does not include a prefix length;

F_seq_number is the message sequence number of the forwarded message;

F_time specifies the time at which this Tuple expires and MUST be removed.

12. Information Base Properties

This section describes some additional properties of Information Bases and their contents.

12.1. Corresponding Protocol Tuples

As part of this specification, in a number of cases there is a natural correspondence from a Protocol Tuple in one Protocol Set to a single Protocol Tuple in another Protocol Set, in the same or another Information Base. The latter Protocol Tuple is referred to as "corresponding" to the former Protocol Tuple.

Specific examples of corresponding Protocol Tuples include:

- There is a Local Interface Tuple corresponding to each Link Tuple, where the Link Tuple is in the Link Set for a MANET interface, and the Local Interface Tuple represents that MANET interface.

- There is a Neighbor Tuple corresponding to each Link Tuple which has L_HEARD_time not EXPIRED, such that N_neighbor_addr_list contains L_neighbor_iface_addr_list.

- There is a Link Tuple (in the Link Set in the same Interface Information Base) corresponding to each 2-Hop Tuple such that L_neighbor_iface_addr_list = N2_neighbor_iface_addr_list.

- There is a Neighbor Tuple corresponding to each 2-Hop Tuple, such that N_neighbor_addr_list contains N2_neighbor_iface_addr_list. (This is the Neighbor Tuple corresponding to the Link Tuple corresponding to the 2-Hop Tuple.)

- There is an Advertising Remote Router Tuple corresponding to each Router Topology Tuple such that AR_orig_addr = TR_from_orig_addr.

- There is an Advertising Remote Router Tuple corresponding to each Routable Address Topology Tuple such that AR_orig_addr = TA_from_orig_addr.

- There is an Advertising Remote Router Tuple corresponding to each Attached Network Tuple such that AR_orig_addr = AN_orig_addr.

- There is a Neighbor Tuple corresponding to each Routing Tuple such that N_neighbor_addr_list contains R_next_iface_addr.

12.2. Address Ownership

Addresses or network addresses with the following properties are considered as "fully owned" by a router when processing a received message:

- Equaling its originator address, OR;
- Equaling the O_orig_addr in an Originator Tuple, OR;
- Equaling or being a sub-range of the I_local_iface_addr_list in a Local Interface Tuple, OR;
- Equaling or being a sub-range of the IR_local_iface_addr in a Removed Interface Address Tuple, OR;
- Equaling an AL_net_addr in a Local Attached Network Tuple.

Addresses or network addresses with the following properties are considered as "partially owned" (which may include being fully owned) by a router when processing a received message:

- Overlapping (equaling or containing) its originator address, OR;
- Overlapping (equaling or containing) the O_orig_addr in an Originator Tuple, OR;
- Overlapping the I_local_iface_addr_list in a Local Interface Tuple, OR;
- Overlapping the IR_local_iface_addr in a Removed Interface Address Tuple, OR;
- Equaling or having as a sub-range an AL_net_addr in a Local Attached Network Tuple.

13. Packets and Messages

The packet and message format used by this protocol is defined in [RFC5444]. Except as otherwise noted, options defined in [RFC5444] may be freely used, in particular alternative formats defined by packet, message, Address Block and TLV flags.

This section describes the usage of the packets and messages defined in [RFC5444] by this specification, and the TLVs defined by, and used

in, this specification.

13.1. Messages

Routers using this protocol exchange information through messages. The message types used by this protocol are the HELLO message and the TC message. The HELLO message is defined by [RFC6130] and extended by this specification, see Section 15. The TC message is defined by this specification, see Section 16.

13.2. Packets

One or more messages sent by a router at the same time SHOULD be combined into a single packet, subject to any constraints on maximum packet size (such as derived from the MTU over a local single hop) that MAY be imposed. These messages may have originated at the sending router, or have originated at another router and are being forwarded by the sending router. Messages with different originating routers MAY be combined for transmission within the same packet. Messages from other protocols defined using [RFC5444], including but not limited to [RFC6130], MAY be combined for transmission within the same packet. This specification does not define or use any contents of the Packet Header.

Forwarded messages MAY be jittered as described in [RFC5148], including the observation that the forwarding jitter of all messages received in a single packet SHOULD be the same. The value of MAXJITTER used in jittering a forwarded message MAY be based on information in that message (in particular any Message TLVs with Type = INTERVAL_TIME or Type = VALIDITY_TIME) or otherwise SHOULD be with a maximum delay of F_MAXJITTER. A router MAY modify the jitter applied to a message in order to more efficiently combine messages in packets, as long as the maximum jitter is not exceeded.

13.3. TLVs

This specification defines two Message TLVs and four Address Block TLVs.

All references in this specification to TLVs that do not indicate a type extension, assume Type Extension = 0. TLVs in processed messages with a type extension which is neither zero as so assumed, nor a specifically indicated non-zero type extension, are ignored.

13.3.1. Message TLVs

The MPR_WILLING TLV is used in HELLO messages. A message MUST NOT contain more than one MPR_WILLING TLV.

Type	Value Length	Value
MPR_WILLING	1 octet	Bits 0-3 encode the parameter WILL_FLOODING; bits 4-7 encode the parameter WILL_ROUTING.

Table 1: MPR_WILLING TLV definition

The CONT_SEQ_NUM TLV is used in TC messages. A message MUST NOT contain more than one CONT_SEQ_NUM TLV.

Type	Value Length	Value
CONT_SEQ_NUM	2 octets	The ANSN contained in the Neighbor Information Base.

Table 2: CONT_SEQ_NUM TLV definition

13.3.2. Address Block TLVs

The LINK_METRIC TLV is used in HELLO messages and TC messages. It MAY use any type extension; only LINK_METRIC TLVs with type extension equal to LINK_METRIC_TYPE will be used by this specification. An address MUST NOT be associated with more than one link metric value for any given type extension, kind (link or neighbor) and direction using this TLV.

Type	Value Length	Value
LINK_METRIC	2 octets	Bits 0-3 indicates kind(s) and direction(s), bits 4-7 indicate exponent (a), bits 8-15 indicate mantissa (b)

Table 3: LINK_METRIC TLV definition

The exponent and mantissa use the representation defined in Section 6. Each bit of the types and directions sub-field, if set ('1') indicates that the link metric is of the indicated kind and direction. Any combination of these bits MAY be used.

Bit	Kind	Direction
0	Link metric	Incoming
1	Link metric	Outgoing
2	Neighbor metric	Incoming
3	Neighbor metric	Outgoing

Table 4: LINK_METRIC TLV types and directions

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.

Type	Value Length	Value
MPR	1 octet	FLOODING indicates that the corresponding address is of a neighbor selected as a flooding MPR, ROUTING indicates that the corresponding address is of a neighbor selected as a routing MPR, FLOOD_ROUTE indicates both (see Section 24.6).

Table 5: MPR TLV definition

The NBR_ADDR_TYPE TLV is used in TC messages.

Type	Value Length	Value
NBR_ADDR_TYPE	1 octet	ORIGINATOR indicates that the corresponding address (which MUST have maximum prefix length) is an originator address, ROUTABLE indicates that the corresponding network address is a routable address of an interface, ROUTABLE_ORIG indicates that the corresponding address is both (see Section 24.6).

Table 6: NBR_ADDR_TYPE TLV definition

If an address is both an originator address and a routable address, then it may be associated with either one Address Block TLV with Type := NBR_ADDR_TYPE and Value := ROUTABLE_ORIG, or with two Address Block TLVs with Type:= NBR_ADDR_TYPE, one with Value := ORIGINATOR and one with Value := ROUTABLE.

The GATEWAY TLV is used in TC messages. An address MUST NOT be associated with more than one hop count value using this TLV.

Type	Value Length	Value
GATEWAY	1 octet	Number of hops to attached network.

Table 7: GATEWAY TLV definition

All address objects included in a TC message according to this specification MUST be associated either with at least one TLV with Type := NBR_ADDR_TYPE or with a TLV with Type := GATEWAY, but not both. Any other address objects MAY be included in Address Blocks in a TC message, but are ignored by this specification.

14. Message Processing and Forwarding

This section describes the optimized flooding operation (MPR flooding) used when control messages, as instances of [RFC5444], are intended for MANET wide distribution. This flooding mechanism defines when a received message is, or is not, processed and/or

forwarded.

This flooding mechanism is used by this protocol and MAY be used by extensions to this protocol which define, and hence own, other Message Types, to manage processing and/or forwarding of these messages. This specification contains elements (P_type, RX_type, F_type) required only for such usage.

This flooding mechanism is always used for TC messages (see Section 16). Received HELLO messages (see Section 15) are, unless invalid, always processed, and never forwarded by this flooding mechanism. They thus do not need to be recorded in the Received Message Information Base.

The processing selection and forwarding mechanisms are designed to only need to parse the Message Header in order to determine whether a message is to be processed and/or forwarded, and not to have to parse the Message Body even if the message is forwarded (but not processed). An implementation MAY only parse the Message Body if necessary, or MAY always parse the Message Body and reject the message if it cannot be so parsed, or any other error is identified.

An implementation MUST discard the message silently if it is unable to parse the Message Header or (if attempted) the Message Body, or if a message (other than a HELLO message) does not include a message sequence number.

14.1. Actions when Receiving a Message

On receiving, on an OLSRv2 interface, a message of a type specified to be using this mechanism, which includes the TC messages defined in this specification, a router MUST perform the following:

1. If the router recognizes from the originator address of the message that the message is one which the receiving router itself originated (i.e., the message originator address is the originator address of this router, or is an O_orig_addr in an Originator Tuple) then the message MUST be silently discarded.

2. Otherwise:

    1. If the message is of a type which may be processed, then the message is considered for processing according to Section 14.2.

    2. If the message is of a type which may be forwarded, AND:

+ <msg-hop-limit> is present and <msg-hop-limit> > 1; AND

+ <msg-hop-count> is not present or <msg-hop-count> < 255;

then the message is considered for forwarding according to Section 14.3.

14.2. Message Considered for Processing

If a message (the "current message") is considered for processing, then the following tasks MUST be performed:

1. If the sending address (i.e., the source address of the IP datagram containing the current message) does not match (taking into account any address prefix) a network address in an L_neighbor_iface_addr_list of a Link Tuple, with L_status = SYMMETRIC, in the Link Set for the OLSRv2 interface on which the current message was received (the "receiving interface") then processing the current message is OPTIONAL. If the current message is not processed then the following steps are not carried out.

2. If a Processed Tuple exists with:

   * P_type = the Message Type of the current message; AND

   * P_orig_addr = the originator address of the current message; AND

   * P_seq_number = the message sequence number of the current message;

   then the current message MUST NOT be processed.

3. Otherwise:

   1. Create a Processed Tuple in the Processed Set with:

      + P_type := the Message Type of the current message;

      + P_orig_addr := the originator address of the current message;

      + P_seq_number := the sequence number of the current message;

      + P_time := current time + P_HOLD_TIME.

2. Process the current message according to its Message Type. For a TC message this is as defined in Section 16.3.

14.3. Message Considered for Forwarding

If a message (the "current message") is considered for forwarding, then the following tasks MUST be performed:

1. If the sending address (i.e., the source address of the IP datagram containing the current message) does not match (taking into account any address prefix) a network address in an L_neighbor_iface_addr_list of a Link Tuple, with L_status = SYMMETRIC, in the Link Set for the OLSRv2 interface on which the current message was received (the "receiving interface") then the current message MUST be silently discarded.

2. Otherwise:

    1. If a Received Tuple exists in the Received Set for the receiving interface, with:

        + RX_type = the Message Type of the current message; AND

        + RX_orig_addr = the originator address of the current message; AND

        + RX_seq_number = the sequence number of the current message;

        then the current message MUST be silently discarded.

    2. Otherwise:

        1. Create a Received Tuple in the Received Set for the receiving interface with:

            - RX_type := the Message Type of the current message;

            - RX_orig_addr := originator address of the current message;

            - RX_seq_number := sequence number of the current message;

            - RX_time := current time + RX_HOLD_TIME.

2. If a Forwarded Tuple exists with:

   - F_type = the Message Type of the current message; AND

   - F_orig_addr = the originator address of the current message; AND

   - F_seq_number = the sequence number of the current message.

   then the current message MUST be silently discarded.

3. Otherwise if the sending address matches (taking account of any address prefix) any network address in an L_neighbor_iface_addr_list of a Link Tuple in the Link Set for the receiving OLSRv2 interface that has L_status = SYMMETRIC and whose corresponding Neighbor Tuple has N_mpr_selector = true, then:

   1. Create a Forwarded Tuple in the Forwarded Set with:

      o F_type := the Message Type of the current message;

      o F_orig_addr := originator address of the current message;

      o F_seq_number := sequence number of the current message;

      o F_time := current time + F_HOLD_TIME.

   2. The Message Header of the current message is modified by:

      o Decrement <msg-hop-limit> in the Message Header by 1; AND

      o If present, increment <msg-hop-count> in the Message Header by 1.

   3. The message is transmitted over all OLSRv2 interfaces, as described in Section 13.

4. Otherwise, the current message MUST be silently discarded.

15. HELLO Messages

The HELLO Message Type is owned by [RFC6130], and thus HELLO messages are generated, transmitted, received and processed by [RFC6130]. This protocol, as permitted by [RFC6130], also uses HELLO messages, including adding to HELLO message generation, and implementing additional processing based on received HELLO messages. HELLO messages are not forwarded by [RFC6130] or by this specification.

15.1. HELLO Message Generation

HELLO messages sent over OLSRv2 interfaces are generated as defined in [RFC6130], and then modified as described in this section. HELLO messages sent on other MANET interfaces are not modified by this specification.

HELLO messages sent over OLSRv2 interfaces are extended by adding the following elements:

- A message originator address, recording this router's originator address. This MUST use a <msg-orig-addr> element, unless:
  - The message specifies only a single local interface address (i.e., contains only one address object that is associated with an Address Block TLV with Type = LOCAL_IF, and which has no prefix length, or a maximum prefix length) which will then be used as the message originator address, OR;
  - The message does not include any local interface network addresses (i.e., has no address objects associated with an Address Block TLV with Type = LOCAL_IF), as permitted by the specification in [RFC6130], when the router that generated the HELLO message has only one interface address and will use that as the sending address of the IP datagram in which the HELLO message is contained. In this case, that address will be used as the message originator address.
- A Message TLV with Type := MPR_WILLING MUST be included.
- The following cases associate Address Block TLVs with one or more addresses from a Link Tuple or a Neighbor Tuple if these are included in the HELLO message. In each case, the TLV MUST be associated with at least one address object for an address from the relevant Tuple; the TLV MAY be associated with more such addresses (including a copy of that address object, possibly not itself associated with any other indicated TLVs, in the same or a different Address Block). These additional TLVs MUST NOT be associated with any other addresses in a HELLO message that will

be processed by [RFC6130].

* For each Link Tuple for which L_in_metric != UNKNOWN_METRIC, and for which one or more addresses in its L_neighbor_iface_addr_list are included as address objects with an associated Address Block TLV with Type = LINK_STATUS and Value = HEARD or Value = SYMMETRIC, at least one of these addresses MUST be associated with an Address Block TLV with Type := LINK_METRIC indicating an incoming link metric with value L_in_metric.

* For each Link Tuple for which L_out_metric != UNKNOWN_METRIC, and for which one or more addresses in its L_neighbor_iface_addr_list are included as address objects with an associated Address Block TLV with Type = LINK_STATUS and Value = SYMMETRIC, at least one of these addresses MUST be associated with an Address Block TLV with Type := LINK_METRIC indicating an outgoing link metric with value L_out_metric.

* For each Neighbor Tuple for which N_symmetric = true, and for which one or more addresses in its N_neighbor_addr_list are included as address objects with an associated Address Block TLV with Type = LINK_STATUS or Type = OTHER_NEIGHB and Value = SYMMETRIC, at least one of these addresses MUST be associated with an Address Block TLV with Type := LINK_METRIC indicating an incoming neighbor metric with value N_in_metric.

* For each Neighbor Tuple for which N_symmetric = true, and for which one or more addresses in its N_neighbor_addr_list are included as address objects with an associated Address Block TLV with Type = LINK_STATUS or Type = OTHER_NEIGHB and Value = SYMMETRIC, at least one of these addresses MUST be associated with an Address Block TLV with Type := LINK_METRIC indicating an outgoing neighbor metric with value N_out_metric.

* For each Neighbor Tuple with N_flooding_mpr = true, and for which one or more network addresses in its N_neighbor_addr_list are included as address objects in the HELLO message with an associated Address Block TLV with Type = LINK_STATUS and Value = SYMMETRIC, at least one of these addresses MUST be associated with an Address Block TLV with Type := MPR and Value := FLOODING or Value := FLOOD_ROUTE.

* For each Neighbor Tuple with N_routing_mpr = true, and for which one or more network addresses in its N_neighbor_addr_list are included as address objects in the HELLO message with an associated Address Block TLV with Type = LINK_STATUS and Value = SYMMETRIC, at least one of these addresses MUST be associated

with an Address Block TLV with Type := MPR and Value := ROUTING or Value := FLOOD_ROUTE.

15.2. HELLO Message Transmission

HELLO messages are scheduled and transmitted by [RFC6130]. This protocol MAY require that an additional HELLO message is sent on each OLSRv2 interface when either of the router's sets of MPRs changes, in addition to the cases specified in [RFC6130], and subject to the constraints specified in [RFC6130] (notably on minimum HELLO message transmission intervals).

15.3. HELLO Message Processing

When received on an OLSRv2 interface, HELLO messages are made available to this protocol in two ways, both as permitted by [RFC6130]:

- Such received HELLO messages MUST be made available to this protocol on reception, which allows them to be discarded before being processed by [RFC6130], for example if the information added to the HELLO message by this specification is inconsistent.

- Such received HELLO messages MUST be made available to OLSRv2 after [RFC6130] has completed its processing thereof, unless discarded as malformed by [RFC6130], for processing by this specification.

15.3.1. HELLO Message Discarding

In addition to the reasons specified in [RFC6130] for discarding a HELLO message on reception, a HELLO message received on an OLSRv2 interface MUST be discarded before processing by [RFC6130] or this specification if it:

- Has more than one Message TLV with Type = MPR_WILLING.

- Has a message originator address, or a network address corresponding to an address object associated with an Address Block TLV with Type = LOCAL_IF, that is partially owned by this router. (Some of these cases are already excluded by [RFC6130].)

- Includes any address object associated with an Address Block TLV with Type = LINK_STATUS or Type = OTHER_NEIGHB that overlaps the message's originator address.

- Contains any address that will be processed by [RFC6130] that is associated, using the same or different address objects, with two

different values of link metric with the same kind and direction using a TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE. This also applies to different addresses that are both of the OLSRv2 interface on which the HELLO message was received.

- Contains any address object associated with an Address Block TLV with Type = MPR that is not also associated with an Address Block TLV with Type = LINK_STATUS and Value = SYMMETRIC (including using a different copy of that address object, in the same or a different Address Block).

15.3.2. HELLO Message Usage

HELLO messages are first processed as specified in [RFC6130]. That processing includes identifying (or creating) a Link Tuple and a Neighbor Tuple corresponding to the originator of the HELLO message (the "current Link Tuple" and the "current Neighbor Tuple"). After this, the following processing MUST also be performed if the HELLO message is received on an OLSRv2 interface and contains a TLV with Type = MPR_WILLING:

1. If the HELLO message has a well-defined message originator address, i.e., has an <msg-orig-addr> element, or has zero or one network addresses associated with a TLV with Type = LOCAL_IF:

    1. Remove any Neighbor Tuple, other than the current Neighbor Tuple, with N_orig_addr = message originator address, taking any consequent action (including removing one or more Link Tuples) as specified in [RFC6130].

    2. The current Link Tuple is then updated according to:

        1. Update L_in_metric and L_out_metric as described in Section 15.3.2.1;

        2. Update L_mpr_selector as described in Section 15.3.2.3.

    3. The current Neighbor Tuple is then updated according to:

        1. N_orig_addr := message originator address;

        2. Update N_in_metric and N_out_metric as described in Section 15.3.2.1;

        3. Update N_will_flooding and N_will_routing as described in Section 15.3.2.2;

4. Update N_mpr_selector as described in Section 15.3.2.3.

2. If there are any changes to the router's Information Bases, then perform the processing defined in Section 17.

15.3.2.1. Updating Metrics

For each address in a received HELLO message with an associated TLV with Type = LINK_STATUS and Value = HEARD or Value = SYMMETRIC, an incoming (to the message originator) link metric value is defined either using an associated TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE that indicates the appropriate kind (link) and direction (incoming) of metric, or as UNKNOWN_METRIC.

For each address in a received HELLO message with an associated TLV with Type = LINK_STATUS and Value = SYMMETRIC, an outgoing (from the message originator) link metric value is defined either using an associated TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE that indicates the appropriate kind (link) and direction (outgoing) of metric, or as UNKNOWN_METRIC.

For each address in a received HELLO message with an associated TLV with Type = LINK_STATUS or Type = OTHER_NEIGHB and Value = SYMMETRIC, an incoming (to the message originator) neighbor metric value is defined either using an associated TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE that indicates the appropriate kind (neighbor) and direction (incoming) of metric, or as UNKNOWN_METRIC.

For each address in a received HELLO message with an associated TLV with Type = LINK_STATUS or Type = OTHER_NEIGHB and Value = SYMMETRIC, an outgoing (from the message originator) neighbor metric value is defined either using an associated TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE that indicates the appropriate kind (neighbor) and direction (outgoing) of metric, or as UNKNOWN_METRIC.

The link metric elements L_in_metric and L_out_metric in a Link Tuple are updated according to the following:

- For any Link Tuple, L_in_metric MAY be set to any representable value, by a process outside this specification, at any time. L_in_metric MUST be so set whenever L_status becomes equal to HEARD or SYMMETRIC (if no other value is available then the value MAXIMUM_METRIC MUST be used). Setting L_in_metric MAY use information based on the receipt of a packet including a HELLO message that causes the creation or updating of the Link Tuple.

- When, as specified in [RFC6130], a Link Tuple is updated (possibly immediately after being created) due to the receipt of a HELLO

message, if L_status = SYMMETRIC, then L_out_metric is set equal to the incoming link metric for any included address of the interface on which the HELLO message was received. (Note that the rules for discarding HELLO messages in Section 15.3.1 make this value unambiguous.) If there is no such link metric then L_out_metric is set to UNKNOWN_METRIC.

The neighbor metric elements N_in_metric and N_out_metric in a Neighbor Tuple are updated according to Section 17.3.

The metric elements N2_in_metric and N2_out_metric in any 2-Hop Tuple updated as defined in [RFC6130] are updated to equal the incoming neighbor metric and outgoing neighbor metric, respectively, associated with the corresponding N2_2hop_addr. If there are no such metrics then these elements are set to UNKNOWN_METRIC.

15.3.2.2. Updating Willingness

N_will_flooding and N_will_routing in the current Neighbor Tuple are updated using the Message TLV with Type = MPR_WILLING (note that this must be present) as follows:

- N_will_flooding := bits 0-3 of the value of that TLV; AND
- N_will_routing := bits 4-7 of the value of that TLV.

(Each being in the range 0 to 15, i.e., WILL_NEVER to WILL_ALWAYS.)

15.3.2.3. Updating MPR Selector Status

L_mpr_selector is updated as follows:

1. If a router finds an address object representing any of its relevant local interface network addresses (i.e., those contained in the I_local_iface_addr_list of an OLSRv2 interface) with an associated Address Block TLV with Type = MPR and Value = FLOODING or Value = FLOOD_ROUTE in the HELLO message (indicating that the originating router has selected the receiving router as a flooding MPR) then, for the current Link Tuple:

   * L_mpr_selector := true.

2. Otherwise (i.e., if no such address object and Address Block TLV was found), if a router finds an address object representing any of its relevant local interface network addresses (i.e., those contained in the I_local_iface_addr_list of an OLSRv2 interface) with an associated Address Block TLV with Type = LINK_STATUS and Value = SYMMETRIC in the HELLO message, then for the current Link

Tuple:

* L_mpr_selector := false.

N_mpr_selector is updated as follows:

1. If a router finds an address object representing any of its relevant local interface network addresses (those contained in the I_local_iface_addr_list of an OLSRv2 interface) with an associated Address Block TLV with Type = MPR and Value = ROUTING or Value = FLOOD_ROUTE in the HELLO message (indicating that the originating router has selected the receiving router as a routing MPR) then, for the current Neighbor Tuple:

   * N_mpr_selector := true;

   * N_advertised := true.

2. Otherwise (i.e., if no such address object and Address Block TLV was found), if a router finds an address object representing any of its relevant local interface network addresses (those contained in the I_local_iface_addr_list of an OLSRv2 interface) with an associated Address Block TLV with Type = LINK_STATUS and Value = SYMMETRIC in the HELLO message, then for the current Neighbor Tuple:

   * N_mpr_selector := false;

   * The router MAY also set N_advertised := false.

16. TC Messages

This protocol defines, and hence owns, the TC Message Type (see Section 24). Thus, as specified in [RFC5444], this protocol generates and transmits all TC messages, receives all TC messages and is responsible for determining whether and how each TC message is to be processed (updating the Topology Information Base) and/or forwarded, according to this specification.

16.1. TC Message Generation

A TC message is a message as defined in [RFC5444]. A generated TC message MUST contain the following elements as defined in [RFC5444]:

o A message originator address, recording this router's originator address. This MUST use a <msg-orig-addr> element.

- <msg-seq-num> element containing the message sequence number.

- A <msg-hop-limit> element, containing TC_HOP_LIMIT. A router MAY use the same or different values of TC_HOP_LIMIT in its TC messages, see Section 5.4.7.

- A <msg-hop-count> element, containing zero, if the message contains a TLV with either Type = VALIDITY_TIME or Type = INTERVAL_TIME (as specified in [RFC5497]) indicating more than one time value according to distance. A TC message MAY contain such a <msg-hop-count> element even if it does not need to.

- A single Message TLV with Type := CONT_SEQ_NUM and Value := ANSN from the Neighbor Information Base. If the TC message is complete then this Message TLV MUST have Type Extension := COMPLETE, otherwise it MUST have Type Extension := INCOMPLETE. (Exception: a TC message MAY omit such a Message TLV if the TC message does not include any address objects with an associated Address Block TLV with Type = NBR_ADDR_TYPE or Type = GATEWAY.)

- A single Message TLV with Type := VALIDITY_TIME, as specified in [RFC5497]. If all TC messages are sent with the same hop limit then this TLV MUST have a value encoding the period T_HOLD_TIME. If TC messages are sent with different hop limits (more than one value of TC_HOP_LIMIT) then this TLV MUST specify times that vary with the number of hops appropriate to the chosen pattern of TC message hop limits, as specified in [RFC5497]; these times SHOULD be appropriate multiples of T_HOLD_TIME. The options included in [RFC5497] for representing zero and infinite times MUST NOT be used.

- If the TC message is complete, all network addresses which are the N_orig_addr of a Neighbor Tuple with N_advertised = true, MUST be represented by address objects in one or more Address Blocks. If the TC message is incomplete then any such address objects MAY be included. At least one copy of each such address object that is included MUST be associated with an Address Block TLV with Type := NBR_ADDR_TYPE, and Value := ORIGINATOR, or with Value := ROUTABLE_ORIG if that address object is also to be associated with Value = ROUTABLE.

- If the TC message is complete, all routable addresses which are in the N_neighbor_addr_list of a Neighbor Tuple with N_advertised = true MUST be represented by address objects in one or more Address Blocks. If the TC message is incomplete then any such address objects MAY be included. At least one copy of each such address object MUST be associated with an Address Block TLV with Type = NBR_ADDR_TYPE, and Value = ROUTABLE, or with Value = ROUTABLE_ORIG

if also to be associated with Value = ORIGINATOR. At least one copy of each such address object MUST be associated with an Address Block TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE indicating an outgoing neighbor metric with value equal to the corresponding N_out_metric.

- If the TC message is complete, all network addresses which are the AL_net_addr of a Local Attached Network Tuple MUST be represented by address objects in one or more Address Blocks. If the TC message is incomplete then any such address objects MAY be included. At least one copy of each such address object MUST be associated with an Address Block TLV with Type := GATEWAY, and Value := AN_dist. At least one copy of each such address object MUST be associated with an Address Block TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE indicating an outgoing neighbor metric equal to the corresponding AL_metric.

A TC message MAY contain:

- A single Message TLV with Type := INTERVAL_TIME, as specified in [RFC5497]. If all TC messages are sent with the same hop limit then this TLV MUST have a value encoding the period TC_INTERVAL. If TC messages are sent with different hop limits, then this TLV MUST specify times that vary with the number of hops appropriate to the chosen pattern of TC message hop limits, as specified in [RFC5497]; these times MUST be appropriate multiples of TC_INTERVAL. The options included in [RFC5497] for representing zero and infinite times MUST NOT be used.

16.2. TC Message Transmission

A router with one or more OLSRv2 interfaces, and with any Neighbor Tuples with N_advertised = true, or with a non-empty Local Attached Network Set MUST generate TC messages. A router which does not have such information to advertise MUST also generate "empty" TC messages for a period A_HOLD_TIME after it last generated a non-empty TC message.

Complete TC messages are generated and transmitted periodically on all OLSRv2 interfaces, with a default interval between two consecutive TC message transmissions by the same router of TC_INTERVAL.

TC messages MAY be generated in response to a change in the information which they are to advertise, indicated by a change in the ANSN in the Neighbor Information Base. In this case a router MAY send a complete TC message, and if so MAY re-start its TC message schedule. Alternatively, a router MAY send an incomplete TC message

with at least the newly advertised network addresses (i.e., not previously, but now, an N_orig_addr or in an N_neighbor_addr_list in a Neighbor Tuple with N_advertised = true, or an AL_net_addr) in its Address Blocks, with associated Address Block TLV(s). Note that a router cannot report removal of advertised content using an incomplete TC message.

When sending a TC message in response to a change of advertised network addresses, a router MUST respect a minimum interval of TC_MIN_INTERVAL between sending TC messages (complete or incomplete), and a maximum interval of TC_INTERVAL between sending complete TC messages. Thus a router MUST NOT send an incomplete TC message if within TC_MIN_INTERVAL of the next scheduled time to send a complete TC message.

The generation of TC messages, whether scheduled or triggered by a change of contents, MAY be jittered as described in [RFC5148]. The values of MAXJITTER used MUST be:

- TP_MAXJITTER for periodic TC message generation;

- TT_MAXJITTER for responsive TC message generation.

16.3. TC Message Processing

On receiving a TC message on an OLSRv2 interface, the receiving router MUST then follow the processing and forwarding procedures, defined in Section 14.

If the message is considered for processing (Section 14.2), then a router MUST first check if the message is invalid for processing by this router, as defined in Section 16.3.1. A router MAY make a similar check before considering a message for forwarding, it MUST make those aspects of the check that apply to elements in the Message Header.

If the TC message is not invalid, then the TC Message Type specific processing, described in Section 16.3.2 MUST be applied. This will update its appropriate Interface Information Bases and its Router Information Base. Following this, if there are any changes in these Information Bases, then the processing in Section 17 MUST be performed.

16.3.1. TC Message Discarding

A received TC message is invalid for processing by this router if the message:

- Has an address length specified in the Message Header that is not equal to the length of the addresses used by this router.
- Does not include a message originator address and a message sequence number.
- Does not include a hop count, and contains a multi-value TLV with Type = VALIDITY_TIME or Type = INTERVAL_TIME, as defined in [RFC5497].
- Does not have exactly one Message TLV with Type = VALIDITY_TIME.
- Has more than one Message TLV with Type = INTERVAL_TIME.
- Does not have a Message TLV with Type = CONT_SEQ_NUM and Type Extension = COMPLETE or Type Extension = INCOMPLETE, and contains at least one address object associated with an Address Block TLV with Type = NBR_ADDR_TYPE or Type = GATEWAY.
- Has more than one Message TLV with Type = CONT_SEQ_NUM and Type Extension = COMPLETE or Type Extension = INCOMPLETE.
- Has a message originator address that is partially owned by this router.
- Includes any address object with a prefix length which is not maximal (equal to the address length, in bits), associated with an Address Block TLV with Type = NBR_ADDR_TYPE and Value = ORIGINATOR or Value = ROUTABLE_ORIG.
- Includes any address object that represents a non-routable address, associated with an Address Block TLV with Type = NBR_ADDR_TYPE and Value = ROUTABLE or Value = ROUTABLE_ORIG.
- Includes any address object associated with an Address Block TLV with Type = NBR_ADDR_TYPE or Type = GATEWAY that also represents the message's originator address.
- Includes any address object (including different copies of an address object, in the same or different Address Blocks) that is associated with an Address Block TLV with Type = NBR_ADDR_TYPE or Type = GATEWAY, that is also associated with more than one outgoing neighbor metric using a TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE.
- Associates any address object (including different copies of an address object, in the same or different Address Blocks) with more than one single hop count value using one or more Address Block

TLV(s) with Type = GATEWAY.

- Associates any address object (including different copies of an address object, in the same or different Address Blocks) with Address Block TLVs with Type = NBR_ADDR_TYPE and Type = GATEWAY.

A router MAY recognize additional reasons for identifying that a message is invalid. An invalid message MUST be silently discarded, without updating the router's Information Bases.

Note that a router that acts inconsistently, for example rejecting TC messages "at random", may cause parts of the network to not be able to communicate with other parts of the network. It is RECOMMENDED that such "additional reasons for identifying that a message is invalid" be a consistent network-wide policy (e.g., as part of a security policy), implemented on all participating routers.

16.3.2. TC Message Processing Definitions

When, according to Section 14.2, a TC message is to be "processed according to its type", this means that:

- If the TC message contains a Message TLV with Type = CONT_SEQ_NUM and Type Extension = COMPLETE, then processing according to Section 16.3.3 and then according to Section 16.3.4 is carried out.

- If the TC message contains a Message TLV with Type = CONT_SEQ_NUM and Type Extension = INCOMPLETE, then only processing according to Section 16.3.3 is carried out.

For the purposes of the TC message processing in Section 16.3.3 and Section 16.3.4:

- "validity time" is calculated from a VALIDITY_TIME Message TLV in the TC message according to the specification in [RFC5497]. All information in the TC message has the same validity time.

- "received ANSN" is defined as being the value of a Message TLV with Type = CONT_SEQ_NUM.

- "associated metric value" is defined for any address in the TC message as being either the outgoing neighbor metric value indicated by a TLV with Type = LINK_METRIC and Type Extension = LINK_METRIC_TYPE that is associated with any address object in the TC message that is equal to that address, or as UNKNOWN_METRIC otherwise. (Note that the rules in Section 16.3.1 make this definition unambiguous.)

- Comparisons of sequence numbers are carried out as specified in Section 21.

16.3.3. Initial TC Message Processing

The TC message is processed as follows:

1. The Advertising Remote Router Set is updated according to Section 16.3.3.1. If the TC message is indicated as discarded in that processing then the following steps are not carried out.

2. The Router Topology Set is updated according to Section 16.3.3.2.

3. The Routable Address Topology Set is updated according to Section 16.3.3.3.

4. The Attached Network Set is updated according to Section 16.3.3.4.

16.3.3.1. Populating the Advertising Remote Router Set

The router MUST update its Advertising Remote Router Set as follows:

1. If there is an Advertising Remote Router Tuple with:

   * AR_orig_addr = message originator address; AND

   * AR_seq_number > received ANSN,

   then the TC message MUST be discarded.

2. Otherwise:

   1. If there is no Advertising Remote Router Tuple such that:

      + AR_orig_addr = message originator address;

      then create an Advertising Remote Router Tuple with:

      + AR_orig_addr := message originator address.

   2. This Advertising Remote Router Tuple (existing or new) is then modified as follows:

      + AR_seq_number := received ANSN;

      + AR_time := current time + validity time.

16.3.3.2. Populating the Router Topology Set

The router MUST update its Router Topology Set as follows:

1. For each address (henceforth advertised address) corresponding to one or more address objects with an associated Address Block TLV with Type = NBR_ADDR_TYPE and Value = ORIGINATOR or Value = ROUTABLE_ORIG, and that is not partially owned by this router, perform the following processing:

    1. If the associated metric is UNKNOWN_METRIC then remove any Router Topology Tuple such that:

        + TR_from_orig_addr = message originator address; AND

        + TR_to_orig_addr = advertised address,

    2. Otherwise if there is no Router Topology Tuple such that:

        + TR_from_orig_addr = message originator address; AND

        + TR_to_orig_addr = advertised address,

        then create a new Router Topology Tuple with:

        + TR_from_orig_addr := message originator address;

        + TR_to_orig_addr := advertised address.

    3. This Router Topology Tuple (existing or new) is then modified as follows:

        + TR_seq_number := received ANSN;

        + TR_metric := associated link metric;

        + TR_time := current time + validity time.

16.3.3.3. Populating the Routable Address Topology Set

The router MUST update its Routable Address Topology Set as follows:

1. For each network address (henceforth advertised address) corresponding to one or more address objects with an associated Address Block TLV with Type = NBR_ADDR_TYPE and Value = ROUTABLE or Value = ROUTABLE_ORIG, and that is not partially owned by this router, perform the following processing:

1. If the associated metric is UNKNOWN_METRIC then remove any Routable Address Topology Tuple such that:

   + TA_from_orig_addr = message originator address; AND

   + TA_dest_addr = advertised address.

2. Otherwise if there is no Routable Address Topology Tuple such that:

   + TA_from_orig_addr = message originator address; AND

   + TA_dest_addr = advertised address,

   then create a new Routable Address Topology Tuple with:

   + TA_from_orig_addr := message originator address;

   + TA_dest_addr := advertised address.

3. This Routable Address Topology Tuple (existing or new) is then modified as follows:

   + TA_seq_number := received ANSN;

   + TA_metric := associated link metric;

   + TA_time := current time + validity time.

16.3.3.4. Populating the Attached Network Set

The router MUST update its Attached Network Set as follows:

1. For each network address (henceforth attached address) corresponding to one or more address objects with an associated Address Block TLV with Type = GATEWAY, and that is not fully owned by this router, perform the following processing:

   1. If the associated metric is UNKNOWN_METRIC then remove any Attached Network Tuple such that:

      + AN_net_addr = attached address; AND

      + AN_orig_addr = message originator address.

   2. Otherwise if there is no Attached Network Tuple such that:

+ AN_net_addr = attached address; AND

+ AN_orig_addr = message originator address,

then create a new Attached Network Tuple with:

+ AN_net_addr := attached address;

+ AN_orig_addr := message originator address.

3. This Attached Network Tuple (existing or new) is then modified as follows:

+ AN_seq_number := received ANSN;

+ AN_dist := the Value of the associated GATEWAY TLV;

+ AN_metric := associated link metric;

+ AN_time := current time + validity time.

16.3.4. Completing TC Message Processing

The TC message is processed as follows:

1. The Router Topology Set is updated according to Section 16.3.4.1.

2. The Routable Address Topology Set is updated according to Section 16.3.4.2.

3. The Attached Network Set is updated according to Section 16.3.4.3.

16.3.4.1. Purging the Router Topology Set

The Router Topology Set MUST be updated as follows:

1. Any Router Topology Tuples with:

* TR_from_orig_addr = message originator address; AND

* TR_seq_number < received ANSN,

MUST be removed.

16.3.4.2. Purging the Routable Address Topology Set

The Routable Address Topology Set MUST be updated as follows:

1. Any Routable Address Topology Tuples with:

   * TA_from_orig_addr = message originator address; AND

   * TA_seq_number < received ANSN,

   MUST be removed.

16.3.4.3. Purging the Attached Network Set

The Attached Network Set MUST be updated as follows:

1. Any Attached Network Tuples with:

   * AN_orig_addr = message originator address; AND

   * AN_seq_number < received ANSN,

   MUST be removed.

17. Information Base Changes

The changes described in the following sections MUST be carried out when any Information Base changes as indicated.

17.1. Originator Address Changes

If the router changes its originator address, then:

1. If there is no Originator Tuple with:

   * O_orig_addr = old originator address

   then create an Originator Tuple with:

   * O_orig_addr := old originator address

   The Originator Tuple (existing or new) with:

   * O_orig_addr = new originator address

   is then modified as follows:

* O_time := current time + O_HOLD_TIME

17.2. Link State Changes

The consistency of a Link Tuple MUST be maintained according to the following rules, in addition to those in [RFC6130]:

- If L_status = HEARD or L_status = SYMMETRIC, then L_in_metric MUST be set (by a process outside this specification).

- If L_status != SYMMETRIC, then set L_mpr_selector := false.

- If L_out_metric = UNKNOWN_METRIC, then L_status MUST NOT equal SYMMETRIC; set L_SYM_time := EXPIRED if this would otherwise be the case.

17.3. Neighbor State Changes

The consistency of a Neighbor Tuple MUST be maintained according to the following rules, in addition to those in [RFC6130]:

1. If N_symmetric = true, then N_in_metric MUST equal the minimum value of all L_in_metric of corresponding Link Tuples with L_status = SYMMETRIC and L_in_metric != UNKNOWN_METRIC. If there are no such Link Tuples then N_in_metric MUST equal UNKNOWN_METRIC.

2. If N_symmetric = true, then N_out_metric MUST equal the minimum value of all L_out_metric of corresponding Link Tuples with L_status = SYMMETRIC and L_out_metric != UNKNOWN_METRIC. If there are no such Link Tuples then N_out_metric MUST equal UNKNOWN_METRIC.

3. If N_symmetric = false, then N_flooding_mpr, N_routing_mpr, N_mpr_selector and N_advertised MUST all be equal to false.

4. If N_mpr_selector = true, then N_advertised MUST be equal to true.

5. If N_symmetric = true, N_out_metric != UNKNOWN_METRIC and N_mpr_selector = false, then a router MAY select N_advertised = true or N_advertised = false. The more neighbors that are advertised, the larger TC messages become, but the more redundancy is available for routing. A router SHOULD consider the nature of its network in making such a decision, and SHOULD avoid unnecessary changes in advertising status, which may result both in additional TC messages having to be sent by its neighbors, and in unnecessary changes to routing, which will have

similar effects to other forms of topology changes in the MANET.

17.4. Advertised Neighbor Changes

The router MUST increment the ANSN in the Neighbor Information Base whenever:

1. Any Neighbor Tuple changes its N_advertised value, or any Neighbor Tuple with N_advertised = true is removed.

2. Any Neighbor Tuple with N_advertised = true changes its N_orig_addr, or has any routable address is added to or removed from N_neighbor_addr_list.

3. Any Neighbor Tuple with N_advertised = true has N_out_metric changed.

4. There is any change to the Local Attached Network Set.

17.5. Advertising Remote Router Tuple Expires

The Router Topology Set, the Routable Address Topology Set and the Attached Network Set MUST be changed when an Advertising Remote Router Tuple expires (AR_time is reached). The following changes are required before the Advertising Remote Router Tuple is removed:

1. All Router Topology Tuples with:

   * TR_from_orig_addr = AR_orig_addr of the Advertising Remote Router Tuple

   are removed.

2. All Routable Address Topology Tuples with:

   * TA_from_orig_addr = AR_orig_addr of the Advertising Remote Router Tuple

   are removed.

3. All Attached Network Tuples with:

   * AN_orig_addr = AR_orig_addr of the Advertising Remote Router Tuple

   are removed.

17.6. Neighborhood Changes and MPR Updates

The sets of symmetric 1-hop neighbors selected as flooding MPRs and routing MPRs MUST satisfy the conditions defined in Section 18. To ensure this:

1. The set of flooding MPRs of a router MUST be recalculated if:

   * A Link Tuple is added with L_status = SYMMETRIC and L_out_metric != UNKNOWN_METRIC, OR;

   * A Link Tuple with L_status = SYMMETRIC and L_out_metric != UNKNOWN_METRIC is removed, OR;

   * A Link Tuple with L_status = SYMMETRIC and L_out_metric != UNKNOWN_METRIC changes to having L_status = HEARD, L_status = LOST or L_out_metric = UNKNOWN_METRIC, OR;

   * A Link Tuple with L_status = HEARD or L_status = LOST changes to having L_status = SYMMETRIC and L_out_metric != UNKNOWN_METRIC, OR;

   * The flooding MPR selection process uses metric values (see Section 18.4) and the L_out_metric of any Link Tuple with L_status = SYMMETRIC changes, OR;

   * The N_will_flooding of a Neighbor Tuple with N_symmetric = true and N_out_metric != UNKNOWN_METRIC changes from WILL_NEVER to any other value, OR;

   * The N_will_flooding of a Neighbor Tuple with N_flooding_mpr = true changes to WILL_NEVER from any other value, OR;

   * The N_will_flooding of a Neighbor Tuple with N_symmetric = true, N_out_metric != UNKNOWN_METRIC, and N_flooding_mpr = false changes to WILL_ALWAYS from any other value, OR;

   * A 2-Hop Tuple with N2_out_metric != UNKNOWN_METRIC is added or removed, OR;

   * The flooding MPR selection process uses metric values (see Section 18.4) and the N2_out_metric of any 2-Hop Tuple changes.

2. Otherwise, the set of flooding MPRs of a router MAY be recalculated if the N_will_flooding of a Neighbor Tuple with N_symmetric = true changes in any other way; it SHOULD be recalculated if N_flooding_mpr = false and this is an increase in

N_will_flooding or if N_flooding_mpr = true and this is a decrease in N_will_flooding.

3. The set of routing MPRs of a router MUST be recalculated if:

   * A Neighbor Tuple is added with N_symmetric = true and N_in_metric != UNKNOWN_METRIC, OR;

   * A Neighbor Tuple with N_symmetric = true and N_in_metric != UNKNOWN_METRIC is removed, OR;

   * A Neighbor Tuple with N_symmetric = true and N_in_metric != UNKNOWN_METRIC changes to having N_symmetric = false, OR;

   * A Neighbor Tuple with N_symmetric = false changes to having N_symmetric = true and N_in_metric != UNKNOWN_METRIC, OR;

   * The N_in_metric of any Neighbor Tuple with N_symmetric = true changes, OR;

   * The N_will_routing of a Neighbor Tuple with N_symmetric = true and N_in_metric != UNKNOWN_METRIC changes from WILL_NEVER to any other value, OR;

   * The N_will_routing of a Neighbor Tuple with N_routing_mpr = true changes to WILL_NEVER from any other value, OR;

   * The N_will_routing of a Neighbor Tuple with N_symmetric = true, N_in_metric != UNKNOWN_METRIC and N_routing_mpr = false changes to WILL_ALWAYS from any other value, OR;

   * A 2-Hop Tuple with N2_in_metric != UNKNOWN_METRIC is added or removed, OR;

   * The N2_in_metric of any 2-Hop Tuple changes.

4. Otherwise, the set of routing MPRs of a router MAY be recalculated if the N_will_routing of a Neighbor Tuple with N_symmetric = true changes in any other way; it SHOULD be recalculated if N_routing_mpr = false and this is an increase in N_will_routing or if N_routing_mpr = true and this is a decrease in N_will_routing.

If either set of MPRs of a router is recalculated, this MUST be as described in Section 18.

17.7. Routing Set Updates

The Routing Set MUST be updated, as described in Section 19, when changes in the Local Information Base, the Neighborhood Information Base or the Topology Information Base indicate a change (including of any potentially used outgoing neighbor metric values) of the known symmetric links and/or attached networks in the MANET, hence changing the Topology Graph. It is sufficient to consider only changes which affect at least one of:

- The Local Interface Set for an OLSRv2 interface, if the change removes any network address in an I_local_iface_addr_list. In this case, unless the OLSRv2 interface is removed, it may not be necessary to do more than replace such network addresses, if used, by an alternative network address from the same I_local_iface_addr_list.

- The Local Attached Set, if the change removes any AL_net_addr which is also an AN_net_addr. In this case it may not be necessary to do more than add Routing Tuples with R_dest_addr equal to that AN_net_addr.

- The Link Set of any OLSRv2 interface, considering only Link Tuples which have, or just had, L_status = SYMMETRIC and L_out_metric != UNKNOWN_METRIC (including removal of such Link Tuples).

- The Neighbor Set of the router, considering only Neighbor Tuples that have, or just had, N_symmetric = true and N_out_metric != UNKNOWN_METRIC, and do not have N_orig_addr = unknown.

- The 2-Hop Set of any OLSRv2 interface, if used in the creation of the Routing Set, and if the change affects any 2-Hop Tuples with N2_out_metric != UNKNOWN_METRIC.

- The Router Topology Set of the router.

- The Routable Address Topology Set of the router.

- The Attached Network Set of the router.

18. Selecting MPRs

Each router MUST select, from among its willing symmetric 1-hop neighbors, two subsets of these routers, as flooding and routing MPRs. This selection is recorded in the router's Neighbor Set, and reported in the router's HELLO messages. Routers MAY select their MPRs by any process that satisfies the conditions which follow, which may, but need not, use the organization of the data described.

Routers can freely interoperate whether they use the same or different MPR selection algorithms.

Only flooding MPRs forward control messages flooded through the MANET, thus effecting a flooding reduction, an optimization of the flooding mechanism, known as MPR flooding. Routing MPRs are used to effect a topology reduction in the MANET. (If no such reduction is required then a router can select all of its relevant neighbors as routing MPRs.) Consequently, while it is not essential that these two sets of MPRs are minimal, keeping the numbers of MPRs small ensures that the overhead of this protocol is kept to a minimum.

18.1. Overview

MPRs are selected according to the following steps, defined in the following sections:

- A data structure known as a Neighbor Graph is defined.
- The properties of an MPR Set derived from a Neighbor Graph are defined. Any algorithm that creates an MPR Set that satisfies these properties is a valid MPR selection algorithm. An example algorithm that creates such an MPR Set is given in Appendix B.
- How to create a Neighbor Graph for each interface based on the corresponding Interface Information Base is defined, and how to combine the resulting MPR Sets to determine the router's flooding MPRs and record those in the router's Neighbor Set.
- How to create a single Neighbor Graph based on all Interface Information Bases and the Neighbor Information Base is defined, and how to record the resulting MPR Set as the router's routing MPRs in the router's Neighbor Set.
- A specification as to when MPRs MUST be calculated is given.

When a router selects its MPRs it MAY consider any characteristics of its neighbors that it is aware of. In particular it SHOULD consider the willingness of the neighbor, as recorded by the corresponding N_will_flooding or N_will_routing value, as appropriate, preferring neighbors with higher values. (Note that willingness values equal to WILL_NEVER and WILL_ALWAYS are always considered, as described.) However, a router MAY consider other characteristics to have a greater significance.

Each router MAY select its flooding and routing MPRs independently from each other, or coordinate its selections. A router MAY make its MPR selections independently of the MPR selection by other routers,

or it MAY, for example, give preference to routers that either are, or are not, already selected as MPRs by other routers.

18.2. Neighbor Graph

A Neighbor Graph is a structure defined here as consisting of sets N1 and N2 and some associated metric and willingness values. Elements of set N1 represent willing symmetric 1-hop neighbors, and elements of set N2 represent addresses of a symmetric 2-hop neighbor.

A Neighbor Graph has the following properties:

- It contains two disjoint sets N1 and N2.
- For each element x in N1 there is an associated willingness value W(x) such that WILL_NEVER < W(x) <= WILL_ALWAYS.
- For each element x in N1 there is an associated metric d1(x) > 0.
- For some elements y in N2 there is an associated metric d1(y) > 0. (Other elements y in N2 have undefined d1(y), this may be considered to be infinite.)
- For each element x in N1 there is a subset N2(x) of elements of N2; this subset may be empty. For each x in N1 and each y in N2(x) there is an associated metric d2(x,y) > 0. (For other x in N1 and y in N2, d2(x,y) is undefined, and may be considered infinite.)
- N2 is equal to the union of all the N2(x) for all x in N1, i.e. for each y in N2 there is at least one x in N1 such that y is in N2(x).

It is convenient to also define:

- For each y in N2 the set N1(y) that contains x in N1 if and only if y is in N2(x). From the final property above, N1(y) is not empty.
- For each x in N1 and y in N2, if d2(x,y) is defined then d(x,y) := d1(x)+d2(x,y), otherwise d(x,y) is not defined. (Thus d(x,y) is defined if y is in N2(x), or equivalently if x is in N1(y).)
- For any subset S of N1, and for each y in N2, the metric d(y,S) is the minimum value of d1(y), if defined, and of all d(x,y) for x in N1(y) and in S. If there are no such metrics to take the minimum value of, then d(y,S) is undefined (may be considered to be infinite). From the final property above, d(y,N1) is defined for

all y.

18.3. MPR Properties

Given a Neighbor Graph as defined in Section 18.2, an MPR Set for that Neighbor Graph is a subset M of the set N1 that satisfies the following properties:

- If x in N1 has W(x) = WILL_ALWAYS then x is in M.
- For any y in N2 that does not have a defined d1(y), there is at least one element in M that is also in N1(y). This is equivalent to the requirement that d(y,M) is defined.
- For any y in N2, d(y,M) = d(y,N1).

These two properties correspond first to that the MPR Set consists 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.

Note that if M is an MPR Set, then so is any subset of N1 that contains M, and also that N1 is always an MPR Set. An MPR Set may be empty, but cannot be empty if N2 contains any elements y that do not have a defined d1(y).

18.4. Flooding MPRs

Whenever flooding MPRs are to be calculated, an implementation MUST determine and record a set of flooding MPRs that is equivalent to one calculated as described in this section.

The calculation of flooding MPRs need not use link metrics, or equivalently may use link metrics with a fixed value, here taken to be 1. However, links with unknown metric (L_out_metric = UNKNOWN_METRIC) MUST NOT be used even if link metrics are otherwise not used.

Routers MAY make individual decisions as to whether to use link metrics for the calculation of flooding MPRs. A router MUST use the same approach to the choice of whether to use link metrics for all links, i.e., in the cases indicated by A or B, the same choice MUST be made in each case.

For each OLSRv2 interface (the "current interface") define a Neighbor Graph as defined in Section 18.2 according to the following:

- Define a reachable Link Tuple to be a Link Tuple in the Link Set for the current interface with L_status = SYMMETRIC and L_out_metric != UNKNOWN_METRIC.

- Define an allowed Link Tuple to be a reachable Link Tuple whose corresponding Neighbor Tuple has N_will_flooding > WILL_NEVER.

- Define an allowed 2-Hop Tuple to be a 2-Hop Tuple in the 2-Hop Set for the current interface for which N2_out_metric != UNKNOWN_METRIC and there is an allowed Link Tuple with L_neighbor_iface_addr_list = N2_neighbor_iface_addr_list.

- Define an element of N1 for each allowed Link Tuple. This then defines the corresponding Link Tuple for each element of N1 and the corresponding Neighbor Tuple for each element of N1, being the Neighbor Tuple corresponding to that Link Tuple.

- For each element x in N1, define W(x) := N_will_flooding of the corresponding Neighbor Tuple.

- For each element x in N1, define d1(x) as either:

  A. L_out_metric of the corresponding Link Tuple, OR;

  B. 1.

- Define an element of N2 for each network address that is the N2_2hop_addr of one or more allowed 2-Hop Tuples. This then defines the corresponding address for each element of N2.

- For each element y in N2, if the corresponding address is in the N_neighbor_addr_list of a Neighbor Tuple that corresponds to one or more reachable Link Tuples, then define d1(y) as either:

  A. the minimum value of the L_out_metric of those Link Tuples, OR;

  B. 1.

  Otherwise d1(y) is not defined. In the latter case, where d1(y) := 1, all such y in N2 may instead be removed from N2.

- For each element x in N1, define N2(x) as the set of elements y in N2 whose corresponding address is the N2_2hop_addr of an allowed 2-Hop Tuple that has N2_neighbor_iface_addr_list = L_neighbor_iface_addr_list of the Link Tuple corresponding to x. For all such x and y, define d2(x,y) as either:

A. N2_out_metric of that 2-Hop Tuple, OR;

B. 1.

It is up to an implementation to decide how to label each element of N1 or N2. For example an element of N1 may be labeled with one or more addresses from the corresponding L_neighbor_iface_addr_list, or with a pointer or reference to the corresponding Link Tuple.

Using these Neighbor Graphs, flooding MPRs are selected and recorded by:

- For each OLSRv2 interface, determine an MPR Set as specified in Section 18.3.
- A Neighbor Tuple represents a flooding MPR and has N_flooding_mpr := true (otherwise N_flooding_mpr := false) if and only if that Neighbor Tuple corresponds to an element in an MPR Set created for any interface as described above. That is, the overall set of flooding MPRs is the union of the sets of flooding MPRs for all OLSRv2 interfaces.

A router MAY select its flooding MPRs for each OLSRv2 interface independently, or it MAY coordinate its MPR selections across its OLSRv2 interfaces, as long as the required condition is satisfied for each OLSRv2 interface. One such coordinated approach is to process the OLSRv2 interfaces sequentially, and for each OLSRv2 interface start with flooding MPRs selected (and not removable) if the neighbor has been already selected as an MPR for an OLSRv2 interface that has already been processed. The algorithm specified in Appendix B can be used in this way.

18.5. Routing MPRs

Whenever routing MPRs are to be calculated, an implementation MUST determine and record a set of routing MPRs that is equivalent to one calculated as described in this section.

Define a single Neighbor Graph as defined in Section 18.2 according to the following:

- Define a reachable Neighbor Tuple to be a Neighbor Tuple with N_symmetric = true and N_in_metric != UNKNOWN_METRIC.
- Define an allowed Neighbor Tuple to be a reachable Neighbor Tuple with N_will_routing > WILL_NEVER.

- Define an allowed 2-Hop Tuple to be a 2-Hop Tuple in the 2-Hop Set for any OLSRv2 interface with N2_in_metric != UNKNOWN_METRIC and for which there is an allowed Neighbor Tuple with N_neighbor_addr_list containing N2_neighbor_iface_addr_list.

- Define an element of N1 for each allowed Neighbor Tuple. This then defines the corresponding Neighbor Tuple for each element of N1.

- For each element x in N1, define W(x) := N_will_routing of the corresponding Neighbor Tuple.

- For each element x in N1, define d1(x) := N_in_metric of the corresponding Neighbor Tuple.

- Define an element of N2 for each network address that is the N2_2hop_addr of one or more allowed 2-Hop Tuples. This then defines the corresponding address for each element of N2.

- For each element y in N2, if the corresponding address is in the N_neighbor_addr_list of a reachable Neighbor Tuple, then define d1(y) to be the N_in_metric of that Neighbor Tuple, otherwise d1(y) is not defined.

- For each element x in N1, define N2(x) as the set of elements y in N2 whose corresponding address is the N2_2hop_addr of an allowed 2-Hop Tuple that has N2_neighbor_iface_addr_list contained in N_neighbor_addr_list of the Neighbor Tuple corresponding to x. For all such x and y, define d2(x,y) := N2_out_metric of that 2-Hop Tuple.

It is up to an implementation to decide how to label each element of N1 or N2. For example an element of N1 may be labeled with one or more addresses from the corresponding N_neighbor_addr_list, or with a pointer or reference to the corresponding Neighbor Tuple.

Using these Neighbor Graphs, routing MPRs are selected and recorded according to the following:

- Determine an MPR Set as specified in Section 18.3.

- A Neighbor Tuple represents a routing MPR and has N_routing_mpr := true (otherwise N_routing_mpr := false) if and only if that Neighbor Tuple corresponds to an element in the MPR Set created as described above.

18.6. Calculating MPRs

A router MUST recalculate each of its sets of MPRs whenever the currently selected set of MPRs does not still satisfy the required conditions. It MAY recalculate its MPRs if the current set of MPRs is still valid, but could be more efficient. Sufficient conditions to recalculate a router's sets of MPRs are given in Section 17.6.

19. Routing Set Calculation

The Routing Set of a router is populated with Routing Tuples that represent paths from that router to all destinations in the network. These paths are calculated based on the Network Topology Graph, which is constructed from information in the Information Bases, obtained via HELLO and TC message exchange.

Changes to the Routing Set do not require any messages to be transmitted. The state of the Routing Set SHOULD, however, be reflected in the IP routing table by adding and removing entries from that routing table as appropriate. Only appropriate Routing Tuples (in particular only those that represent local links or paths to routable addresses) need be reflected in the IP routing table.

19.1. Network Topology Graph

The Network Topology Graph is formed from information from the router's Local Interface Set, Link Sets for OLSRv2 interfaces, Neighbor Set, Router Topology Set, Routable Address Topology Set and Attached Network Set. The Network Topology Graph MAY also use information from the router's 2-Hop Sets for OLSRv2 interfaces. The Network Topology Graph forms the router's topological view of the network in form of a directed graph. Each edge in that graph has a metric value. The Network Topology Graph has a "backbone" (within which minimum total metric routes will be constructed) containing the following edges:

- Edges X -> Y for all possible Y, and one X per Y, such that:
  - Y is the N_orig_addr of a Neighbor Tuple; AND
  - N_orig_addr is not unknown;
  - X is in the I_local_iface_addr_list of a Local Interface Tuple; AND
  - There is a Link Tuple with L_status = SYMMETRIC and L_out_metric != UNKNOWN_METRIC such that this Neighbor Tuple and this Local Interface Tuple correspond to it. A network

address from L_neighbor_iface_addr_list will be denoted R in this case.

It SHOULD be preferred, where possible, to select R = S and X from the Local Interface Tuple corresponding to the Link Tuple from which R was selected. The metric for such an edge is the corresponding N_out_metric.

- All edges W -> U such that:
  - W is the TR_from_orig_addr of a Router Topology Tuple; AND
  - U is the TR_to_orig_addr of the same Router Topology Tuple.

  The metric of such an edge is the corresponding TR_metric.

The Network Topology Graph is further "decorated" with the following edges. If a network address S, V, Z or T equals a network address Y or W, then the edge terminating in the network address S, V, Z or T MUST NOT be used in any path.

- Edges X -> S for all possible S, and one X per S, such that:
  - S is in the N_neighbor_addr_list of a Neighbor Tuple; AND
  - X is in the I_local_iface_addr_list of a Local Interface Tuple; AND
  - There is a Link Tuple with L_status = SYMMETRIC and L_out_metric != UNKNOWN_METRIC such that this Neighbor Tuple and this Local Interface Tuple correspond to it. A network address from L_neighbor_iface_addr_list will be denoted R in this case.

  It SHOULD be preferred, where possible, to select R = S and X from the Local Interface Tuple corresponding to the Link Tuple from which R was selected. The metric for such an edge is the corresponding N_out_metric.

- All edges W -> V such that:
  - W is the TA_from_orig_addr of a Routable Address Topology Tuple; AND
  - V is the TA_dest_addr of the same Routable Address Topology Tuple.

  The metric for such an edge is the corresponding TA_metric.

- All edges W -> T such that:
  - W is the AN_orig_addr of an Attached Network Tuple; AND
  - T is the AN_net_addr of the same Attached Network Tuple.

  The metric for such an edge is the corresponding AN_metric.

- (OPTIONAL) All edges Y -> Z such that:
  - Z is a routable address and is the N2_2hop_addr of a 2-Hop Tuple with N2_out_metric != UNKNOWN_METRIC; AND
  - Y is the N_orig_addr of the corresponding Neighbor Tuple; AND
  - This Neighbor Tuple has N_will_routing not equal to WILL_NEVER.

  A path terminating with such an edge MUST NOT be used in preference to any other path. The metric for such an edge is the corresponding N2_out_metric.

Any part of the Topology Graph which is not connected to a local network address X is not used. Only one selection X SHOULD be made from each I_local_iface_addr_list, and only one selection R SHOULD be made from any L_neighbor_iface_addr_list. All edges have a hop count of 1, except edges W -> T that have a hop count of the corresponding value of AN_dist.

19.2. Populating the Routing Set

The Routing Set MUST contain the shortest paths for all destinations from all local OLSRv2 interfaces using the Network Topology Graph. This calculation MAY use any algorithm, including any means of choosing between paths of equal total metric. (In the case of two paths of equal total metric but differing hop counts, the path with the lower hop count SHOULD be used.)

Using the notation of Section 19.1, initially "backbone" paths using only edges X -> Y and W -> U need be constructed (using a minimum distance algorithm). Then paths using only a final edge of the other types may be added. These MUST NOT replace backbone paths with the same destination (and paths terminating in an edge Y -> Z SHOULD NOT replace paths with any other form of terminating edge).

Each path will correspond to a Routing Tuple. These will be of two types. The first type will represent single edge paths, of type X -> S or X -> Y, by:

- R_local_iface_addr := X;
- R_next_iface_addr := R;
- R_dest_addr := S or Y;
- R_dist := 1;
- R_metric := edge metric,

where R is as defined in Section 19.1 for these types of edges.

The second type will represent a multiple edge path, which will always have first edge of type X -> Y, and will have final edge of type W -> U, W -> V, W -> T or Y -> Z. The Routing Tuple will be:

- R_local_iface_addr := X;
- R_next_iface_addr := Y;
- R_dest_addr := U, V, T or Z;
- R_dist := the total hop count of all edges in the path;
- R_metric := the total metric of all edges in the path.

Finally, Routing Tuples of the second type whose R_dest_addr is not routable MAY be discarded.

An example algorithm for calculating the Routing Set of a router is given in Appendix C.

20. Proposed Values for Parameters

This protocol uses all parameters defined in [RFC6130] and additional parameters and defined in this specification. All but one (RX_HOLD_TIME) of these additional parameters are router parameters as defined in [RFC6130]. The proposed values of the additional parameters defined in the following sections are appropriate to the case where all parameters (including those defined in [RFC6130]) have a single value. Proposed values for parameters defined in [RFC6130] are given in that specification.

The following proposed values are based on experience with [RFC3626] deployments (such as documented in [McCabe]), and are considered typical. They can be changed to accommodate different deployment requirements - for example, a network with capacity-limited network interfaces would be expected to use greater values for message

intervals, whereas a highly mobile network would be expected to use lower values for message intervals. When determining these values, the constraints specified in Section 5 MUST be respected.

Note that routers in a MANET need not all use the same set of parameters, and those parameters that are indicated as interface parameters need not be the same on all OLSRv2 interfaces of a single router.

20.1. Local History Time Parameters

- O_HOLD_TIME := 30 seconds

20.2. Message Interval Parameters

- TC_INTERVAL := 5 seconds
- TC_MIN_INTERVAL := TC_INTERVAL/4

20.3. Advertised Information Validity Time Parameters

- T_HOLD_TIME := 3 x TC_INTERVAL
- A_HOLD_TIME := T_HOLD_TIME

20.4. Received Message Validity Time Parameters

- RX_HOLD_TIME := 30 seconds
- P_HOLD_TIME := 30 seconds
- F_HOLD_TIME := 30 seconds

20.5. Jitter Time Parameters

- TP_MAXJITTER := HP_MAXJITTER
- TT_MAXJITTER := HT_MAXJITTER
- F_MAXJITTER := TT_MAXJITTER

20.6. Hop Limit Parameter

- TC_HOP_LIMIT := 255

20.7. Willingness Parameter

- WILL_FLOODING := WILL_DEFAULT

- WILL_ROUTING := WILL_DEFAULT

21. Sequence Numbers

Sequence numbers are used in this specification for the purpose of discarding "old" information, i.e., messages received out of order. However with a limited number of bits for representing sequence numbers, wrap-around (that the sequence number is incremented from the maximum possible value to zero) will occur. To prevent this from interfering with the operation of this protocol, the following MUST be observed when determining the ordering of sequence numbers.

The term MAXVALUE designates in the following one more than the largest possible value for a sequence number. For a 16 bit sequence number (as are those defined in this specification) MAXVALUE is 65536.

The sequence number S1 is said to be "greater than" the sequence number S2 if:

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

- S2 > S1 AND S2 - S1 > MAXVALUE/2

When sequence numbers S1 and S2 differ by MAXVALUE/2 their ordering cannot be determined. In this case, which should not occur, either ordering may be assumed.

Thus when comparing two messages, it is possible - even in the presence of wrap-around - to determine which message contains the most recent information.

22. Extensions

An extension to this protocol will need to interact with this specification, and possibly also with [RFC6130]. This protocol is designed to permit such interactions, in particular:

- Through accessing, and possibly extending, the information in the Information Bases. All updates to the elements specified in this specification are subject to the normative constraints specified in [RFC6130] and Appendix A. Note that the processing specified in this document ensures that these constraints are satisfied.

- Through accessing an outgoing message prior to it being transmitted over any OLSRv2 interface, and to add information to it as specified in [RFC5444]. This MAY include Message TLVs and/or network addresses with associated Address Block TLVs. (Network addresses without new associated TLVs SHOULD NOT be added to messages.) This may, for example, be to allow a security protocol, as suggested in Section 23, to add a TLV containing a cryptographic signature to the message.

- Through accessing an incoming message, and potentially discarding it prior to processing by this protocol. This may, for example, allow a security protocol, as suggested in Section 23, to perform verification of message signatures and prevent processing and/or forwarding of unverifiable messages by this protocol.

- Through accessing an incoming message after it has been completely processed by this protocol. In particular, this may allow a protocol which has added information, by way of inclusion of appropriate TLVs, or of network addresses associated with new TLVs, access to such information after appropriate updates have been recorded in the Information Bases in this protocol.

- Through requesting that a message be generated at a specific time. In that case, message generation MUST still respect the constraints in [RFC6130] and Section 5.4.3.

23. Security Considerations

As a proactive routing protocol, this protocol is a potential target for various attacks. This section presents the envisioned security architecture for OLSRv2, and gives guidelines on how to provide integrity, confidentiality, and integration into external routing domains.

23.1. Security Architecture

OLSRv2 integrates into the architecture specified in Appendix A of [RFC5444], in [RFC5498], and in Section 16 of [RFC6130], the latter by using and extending its messages and Information Bases.

As part of this architecture, OLSRv2, and [RFC6130], recognize that there may be external reasons for rejecting messages that would be considered "badly formed" or "insecure", e.g., if a digital signature included in a message by an external mechanism cannot be verified. This architecture allows options as to whether and how to implement security features, reflecting the situation that MANET routing protocol deployment domains have varying security requirements, ranging from "practically unbreakable" to "virtually none". This

approach allows MANET routing protocol specifications to remain generic, with extensions to them, and/or extensions to the multiplexing and demultiplexing process described in Appendix A of [RFC5444], providing security mechanisms appropriate to a given deployment domain.

The following sections provide guidelines how to provide integrity, confidentiality, and integration with external routing domains in such extensions.

23.2. Integrity

Each router injects topological information into the network by transmitting HELLO messages and, for some routers, also TC messages. If some routers for some reason, malice or malfunction, inject invalid control traffic, network integrity may be compromised. Therefore, message, or packet, authentication is recommended.

Different such situations may occur, for example:

1. A router generates TC messages, advertising links to non-neighbor routers;

2. A router generates TC messages, pretending to be another router;

3. A router generates HELLO messages, advertising non-neighbor routers;

4. A router generates HELLO messages, pretending to be another router;

5. A router forwards altered control messages;

6. A router does not forward control messages;

7. A router does not select multipoint relays correctly;

8. A router forwards broadcast control messages unaltered, but does not forward unicast data traffic;

9. A router "replays" previously recorded control traffic from another router.

Authentication of the originator router for control messages (for situations 2, 4 and 5), and of the individual links announced in the control messages (for situations 1 and 3), may be used as a countermeasure. However to prevent routers from repeating old (and correctly authenticated) information (situation 9) temporal

information (a timestamp) is required, allowing a router to positively identify such replayed messages.

In general, digital signatures and other required security information may be transmitted within the HELLO and TC messages, or within a packet header, using the TLV mechanism. Either option permits "secured" and "unsecured" routers to coexist in the same network, if desired.

An important consideration is that all control messages (HELLO messages and TC messages) are transmitted to all routers in the 1-hop neighborhood and some control messages (TC messages) are flooded to all routers in the network. This is done in a packet that is transmitted to all routers in the 1-hop neighborhood, the current set of which may not be known. Thus, a control message, or packet, used by this protocol is always contained in a transmission destined for multiple destinations, and it is important that the authentication mechanism employed permits any receiving router to validate the authenticity of a message or packet.

[RFC6622bis] specifies a common exchange format for cryptographic information in the form of Packet TLVs, Message TLVs, and Address Block TLVs, as specified in [RFC5444]. These may be used (and shared) among MANET routing protocol security extensions. In particular, [RFC6622bis] specifies the format of TLVs for containing Integrity Check Values (ICVs), i.e., signatures, for providing integrity, as well as TLVs for containing temporal information for preventing replay attacks. [RFC6622bis] specifies registries for using different ciphers and formats of temporal information. Failure to verify an ICV included can be used as a reason to reject an incoming message or packet as "invalid", according to Section 12.1 of [RFC6130], and according to Section 16.3.1 of this specification, when using the multiplexing and demultiplexing process described in Appendix A of [RFC5444].

Any security extension that uses [RFC6622bis] must specify how to insert IVCs in generated messages, how to verify incoming messages, and to reject "insecure" messages (i.e., messages without an ICV or with an ICV that cannot be verified) when operating in a secure mode. Different MANET deployments may, as a result of their purpose for which they are used and the possibility and nature of their configuration, require different ICV algorithms and timestamps, and thus a security extension may use any of the different options provided in [RFC6622bis].

23.3. Confidentiality

OLSRv2 periodically MPR floods topological information to all routers in the network. Hence, if used in an unprotected network, in particular an unprotected wireless network, the network topology is revealed to anyone who successfully listens to the control messages. This information may serve an attacker to acquire details about the topology, and therefore to initiate more effective attacks against routers in the routing domain e.g., by spoofing addresses of routers in the network and attracting traffic for these addresses. Note that this is independent of the data traffic, and purely protects the control traffic, i.e., information about the network topology.

In situations where the confidentiality of the network topology is of importance, regular cryptographic techniques, such as use of OLSRv2 multicast control packets encrypted using IPsec (e.g., with a shared secret key), can be applied to ensure that control traffic can be read and interpreted by only those authorized to do so. Alternatively, a security extension may specify a mechanism to provide confidentiality for control messages and/or packets. However, unless the information about the network topology itself is confidential, integrity of control messages (as specified in Section 23.2) is sufficient to admit only trusted routers (i.e. routers with valid credentials) to the network.

23.4. Interaction with External Routing Domains

This protocol provides a basic mechanism for injecting external routing information into this protocol's routing domain. Routing information can also be extracted from this protocol's Information Bases, in particular the Routing Set, and injected into an external routing domain, if the routing protocol governing that routing domain permits this.

When operating routers connecting a routing domain using this protocol to an external routing domain, care MUST be taken not to allow potentially insecure and untrustworthy information to be injected from this routing domain to an external routing domain. Care MUST also be taken to validate the correctness of information prior to it being injected, so as to avoid polluting routing tables with invalid information.

A recommended way of extending connectivity from an external routing domain to this routing domain, which is routed using this protocol, is to assign an IP prefix (under the authority of the routers/ gateways connecting this routing domain with the external routing domain) exclusively to this routing domain, and to configure the gateways to advertise routes for that IP prefix into the external

routing domain.

23.5. Mandatory Security Mechanisms

A conformant implementation of OLSRv2 MUST, at a minimum, implement the security mechanisms specified in [OLSRv2-integrity], providing integrity and replay protection of OLSRv2 control messages, including of HELLO messages specified by [RFC6130] that are used by OLSRv2, by including an Integrity Check Value (ICV) TLV and a timestamp TLV. The ICV TLV uses a SHA-256 based HMAC and a single shared secret key. The timestamp TLV is based on POSIX time, assuming router time synchronization.

The baseline use case, for which this security mechanism provides adequate integrity protection without rekeying, is for short-lived (for example up to a couple of months) OLSRv2 deployments.

Any deployment of OLSRv2 SHOULD use an appropriate security mechanism, possibly different from the one described in [OLSRv2-integrity]. For example, for longer-term OLSRv2 deployments, alternative security mechanisms (e.g., including re-keying) SHOULD be considered.

23.6. Key Management

This specification does not mandate automatic key management (AKM) as part of the security architecture for OLSRv2. While certain use cases for OLSRv2 may lend themselves to AKM, the baseline assumption is that many use cases do not, for the reasons detailed below.

Bootstrapping a key is hard in a radio network, where it is - in general - not possible to determine from where a received signal was transmitted, or if two transmissions come from the same or from different sources.

A widespread use of radio networks, mobile phone networks, works under the assumptions that (i) secret information is embedded in mobile phones at manufacture, and (ii) a centralized database of this is accessible during the network lifetime.

However, as a primary use case of a MANET is to provide connectivity absent centralized entities, and with minimal management, a solution such as that used in a mobile phone network is not feasible. In many instances, a cryptographic authority may not be present in the MANET at all, since such a cryptographic authority would be too vulnerable. Due to the potentially dynamic topology of a MANET, a cryptographic authority may also become unreachable (to some or all of the MANET routers) without prior warning.

[BCP107] provides guidelines for cryptographic key management. Specifically, Section 2.1 describes requirements for when AKM is required, and Section 2.2 describes conditions under which manual key management is acceptable.

Section 2.1 of [BCP107] stipulates that "Automated key management MUST be used if any of [a set of given] conditions hold". These conditions are listed below, for each of which arguments are provided as to their applicability for the baseline use case of OLSRv2.

A party will have to manage n^2 static keys, where n may become large.

> The baseline use case of OLSRv2 uses only one key in the whole MANET.

Any stream cipher (such as RC4 [TK], AES-CTR [NIST], or AES-CCM [WHF]) is used.

> A stream cipher is not envisioned to be used for generating ICVs for OLSRv2 control messages.

An initialization vector (IV) might be reused, especially an implicit IV. Note that random or pseudo-random explicit IVs are not a problem unless the probability of repetition is high.

> An IV is not envisioned to be used for generating ICVs for OLSRv2 control messages.

Large amounts of data might need to be encrypted in a short time, causing frequent change of the short-term session key.

> Integrity Check Values (ICVs) are required only for OLSRv2 control messages, which are low-volume messages.

Long-term session keys are used by more than two parties. Multicast is a necessary exception, but multicast key management standards are emerging in order to avoid this in the future. Sharing long-term session keys should generally be discouraged.

> OLSRv2 control messages are all sent using link local multicast.

The likely operational environment is one where personnel (or device) turnover is frequent, causing frequent change of the short-term session key.

This is not an intended deployment of OLSRv2. For longer-term OLSRv2 deployments, alternative security mechanisms (e.g., including re-keying) SHOULD be considered.

Section 2.2 of [BCP107] stipulates that "Manual key management may be a reasonable approach in any of [a given set of] situations". These situation are listed below, for each of which arguments are provided as to their applicability for the baseline use case of OLSRv2.

The environment has very limited available bandwidth or very high round-trip times. Public key systems tend to require long messages and lots of computation; symmetric key alternatives, such as Kerberos, often require several round trips and interaction with third parties.

Bandwidth is limited, both by being a radio environment, and by the need for any signaling to consume a minimal proportion thereof. A system such as Kerberos would therefore be inappropriate - and as previously noted, there may not be the required infrastructure (cryptographic authority) present - or, if present, reachable - in the MANET.

The information being protected has low value.

This depends on the OLSRv2 use case, but the information being protected is OLSRv2 control traffic, which is of moderate value.

The total volume of traffic over the entire lifetime of the long-term session key will be very low.

Integrity Check Values (ICVs) are required only for OLSRv2 control messages, which are low-volume messages.

The scale of each deployment is very limited

A typical use case for OLSRv2 may involve only tens of devices - with even the largest use cases for OLSRv2 being small by Internet standards.

24. IANA Considerations

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

24.1. Expert Review: Evaluation Guidelines

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

24.2. Message Types

This specification defines one Message Type, to be allocated from the 0-223 range of the "Message Types" namespace defined in [RFC5444], as specified in Table 8.

Type	Description
TBD1	TC : Topology Control (MANET-wide signaling)

Table 8: Message Type assignment

24.3. Message-Type-Specific TLV Type Registries

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

Type	Description	Allocation Policy
128-223	Unassigned	Expert Review

Table 9: TC Message-Type-specific Message TLV Types

IANA is requested to create a registry for Message-Type-specific Address Block TLVs for TC 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
128-223	Unassigned	Expert Review

Table 10: TC Message-Type-specific Address Block TLV Types

24.4. Message TLV Types

This specification defines two Message TLV Types, which must be allocated from the "Message TLV Types" namespace defined in [RFC5444]. IANA is requested to make allocations in the 0-127 range for these types. This will create two new Type Extension registries with assignments as specified in Table 11 and Table 12. Specifications of these TLVs are in Section 13.3.1. Each of these TLVs MUST NOT be included more than once in a Message TLV Block.

Name	Type	Type Extension	Description	Allocation Policy
MPR_WILLING	TBD2	0	Bits 0-3 specify the originating router's willingness to act as a flooding MPR; bits 4-7 specify the originating router's willingness to act as a routing MPR.	
MPR_WILLING	TBD2	1-255	Unassigned.	Expert Review

Table 11: Message TLV Type assignment: MPR_WILLING

Name	Type	Type Extension	Description	Allocation Policy
CONT_SEQ_NUM	TBD3	0	COMPLETE: Specifies a content sequence number for this complete message.	
CONT_SEQ_NUM	TBD3	1	INCOMPLETE: Specifies a content sequence number for this incomplete message.	
CONT_SEQ_NUM	TBD3	2-255	Unassigned.	Expert Review

Table 12: Message TLV Type assignment: CONT_SEQ_NUM

Type extensions indicated as Expert Review SHOULD be allocated as described in [RFC5444], based on Expert Review as defined in [RFC5226].

24.5. Address Block TLV Types

This specification defines four Address Block TLV Types, which must be allocated from the "Address Block TLV Types" namespace defined in [RFC5444]. IANA are requested to make allocations in the 8-127 range for these types. This will create four new Type Extension registries with assignments as specified in Table 13, Table 14, Table 15 and Table 16. Specifications of these TLVs are in Section 13.3.2.

Name	Type	Type Extension	Description	Allocation Policy
LINK_METRIC	TBD4	0	Link metric meaning assigned by administrative action.	
LINK_METRIC	TBD4	1-223	Unassigned.	Expert Review
LINK_METRIC	TBD4	224-255	Unassigned.	Experimental Use

Table 13: Address Block TLV Type assignment: LINK_METRIC

All LINK_METRIC TLVs, whatever their type extension, MUST use their value field to encode the kind and value (in the interval MINIMUM_METRIC, to MAXIMUM_METRIC, inclusive) of a link metric as specified in Section 6 and Section 13.3.2. An assignment of a LINK_METRIC TLV type extension MUST specify the physical meaning, and mapping of that physical meaning to the representable values in the indicated interval, of the link metric.

Name	Type	Type Extension	Description	Allocation Policy
MPR	TBD5	0	Specifies that a given network address is of a router selected as a flooding MPR (FLOODING = 1), that a given network address is of a router selected as a routing MPR (ROUTING = 2), or both (FLOOD_ROUTE = 3).	
MPR	TBD5	1-255	Unassigned.	Expert Review

Table 14: Address Block TLV Type assignment: MPR

Name	Type	Type Extension	Description	Allocation Policy
NBR_ADDR_TYPE	TBD6	0	Specifies that a given network address is of a neighbor reached via the originating router, if it is an originator address (ORIGINATOR = 1), is a routable address (ROUTABLE = 2), or if it is both (ROUTABLE_ORIG = 3).	
NBR_ADDR_TYPE	TBD6	1-255	Unassigned.	Expert Review

Table 15: Address Block TLV Type assignment: NBR_ADDR_TYPE

Name	Type	Type extension	Description	Allocation Policy
GATEWAY	TBD7	0	Specifies that a given network address is reached via a gateway on the originating router, with value equal to the number of hops.	
GATEWAY	TBD7	1-255		Expert Review

Table 16: Address Block TLV Type assignment: GATEWAY

Type extensions indicated as Expert Review SHOULD be allocated as described in [RFC5444], based on Expert Review as defined in [RFC5226].

24.6. NBR_ADDR_TYPE and MPR Values

Note: This section does not require any IANA action, as the required information is included in the descriptions of the MPR and NBR_ADDR_TYPE Address Block TLVs allocated in Section 24.5. This information is recorded here for clarity, and for use elsewhere in this specification.

The Values which the MPR Address Block TLV can use are the following:

- FLOODING := 1;
- ROUTING := 2;
- FLOOD_ROUTE := 3.

The Values which the NBR_ADDR_TYPE Address Block TLV can use are the following:

- ORIGINATOR := 1;
- ROUTABLE := 2;
- ROUTABLE_ORIG := 3.

25. Contributors

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

- Cedric Adjih, INRIA, France, <Cedric.Adjih@inria.fr>
- Emmanuel Baccelli, INRIA , France, <Emmanuel.Baccelli@inria.fr>
- Thomas Heide Clausen, LIX, France, <T.Clausen@computer.org>
- Justin Dean, NRL, USA, <jdean@itd.nrl.navy.mil>
- Christopher Dearlove, BAE Systems, UK, <chris.dearlove@baesystems.com>
- Ulrich Herberg, Fujitsu Laboratories of America, USA, <ulrich@herberg.name>
- Satoh Hiroki, Hitachi SDL, Japan, <hiroki.satoh.yj@hitachi.com>
- Philippe Jacquet, Alcatel Lucent Bell Labs, France, <philippe.jacquet@alcatel-lucent.fr>

- Monden Kazuya, Hitachi SDL, Japan, <kazuya.monden.vw@hitachi.com>
- Kenichi Mase, Niigata University, Japan, <mase@ie.niigata-u.ac.jp>
- Ryuji Wakikawa, Toyota, Japan, <ryuji@sfc.wide.ad.jp>

26. Acknowledgments

The authors would like to acknowledge the team behind OLSRv1, as listed in RFC3626, including Anis Laouiti (INT), Pascale Minet (INRIA), Paul Muhlethaler (INRIA), Amir Qayyum (M.A. Jinnah University), and Laurent Viennot (INRIA) for their contributions.

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): Khaldoun Al Agha (LRI), Teco Boot (Infinity Networks), Ross Callon (Juniper), Song-Yean Cho (Samsung), Alan Cullen (BAE Systems), Louise Lamont (CRC), Li Li (CRC), Joseph Macker (NRL), Richard Ogier (SRI), Charles E. Perkins (Futurewei), Henning Rogge (Frauenhofer FKIE), and the entire IETF MANET Working Group.

Finally, the authors would like to express their gratitude to the Area Directors for providing valuable review comments during the IESG evaluation, in particular (listed alphabetically) Benoit Claise, Adrian Farrel, Stephen Farrell, Barry Leiba, Pete Resnick, and Martin Stiemerling.

27. References

27.1. Normative References

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

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

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 5226, BCP 26, May 2008.

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

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

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

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

[OLSRv2-integrity] Herberg, U., Dearlove, C., and T. Clausen, "Integrity Protection for Control Messages in NHDP and OLSRv2", Internet Draft draft-herberg-manet-nhdp-olsrv2-sec-02, March 2013.

27.2. 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.

[RFC6622bis] Herberg, U., Clausen, T., and C. Dearlove, "Integrity Check Value and Timestamp TLV Definitions for Mobile Ad Hoc Networks (MANETs)", Internet Draft draft-herberg-manet-rfc6622-bis-02, March 2013.

[HIPERLAN] ETSI, "ETSI STC-RES10 Committee. Radio equipment and systems: HIPERLAN type 1, functional specifications ETS 300-652", June 1996.

[HIPERLAN2] Jacquet, P., Minet, P., Muhlethaler, P., and N. Rivierre, "Increasing reliability in cable free radio LANs: Low level forwarding in HIPERLAN.", 1996.

[MPR] Qayyum, A., Viennot, L., and A. Laouiti, "Multipoint relaying: An efficient technique for flooding in mobile wireless networks.", 2001.

[FSR] Pei, G., Gerla, M., and T. Chen, "Fisheye state

routing in mobile ad hoc networks", 2000.

[FSLS] Santivanez, C., Ramanathan, R., and I. Stavrakakis, "Making link-state routing scale for ad hoc networks", 2000.

[McCabe] McCabe, A., Dearlove, C., Fredin, M., and L. Axelsson, "Scalability Modelling of Ad Hoc Routing Protocols - A Comparison of OLSR and DSR", 2004.

[BCP107] Bellovin, S. and R. Housley, "Guidelines for Cryptographic Key Management", BCP 107, RFC 4107, June 2005.

Appendix A. Constraints

Updates to the Local Information Base, the Neighborhood Information Base or the Topology Information Base MUST ensure that all constraints specified in this appendix are maintained, as well as those specified in [RFC6130]. This is the case for the processing, specified in this document. Any protocol extension or outside process, which updates the Neighborhood Information Base or the Topology Information Base, MUST also ensure that these constraints are maintained.

In each Originator Tuple:

- O_orig_addr MUST NOT equal any other O_orig_addr.
- O_orig_addr MUST NOT equal this router's originator address.

In each Local Attached Network Tuple:

- AL_net_addr MUST NOT equal any other AL_net_addr.
- AL_net_addr MUST NOT equal or be a sub-range of any network address in the I_local_iface_addr_list of any Local Interface Tuple.
- AL_net_addr MUST NOT equal this router's originator address, or equal the O_orig_addr in any Originator Tuple.
- AL_dist MUST NOT be less than zero.

In each Link Tuple:

- L_neighbor_iface_addr_list MUST NOT contain any network address that AL_net_addr of any Local Attached Network Tuple equals or is a sub-range of.

- if L_in_metric != UNKNOWN_METRIC then L_in_metric MUST be representable in the defined compressed form.

- if L_out_metric != UNKNOWN_METRIC then L_out_metric MUST be representable in the defined compressed form.

- If L_mpr_selector = true, then L_status = SYMMETRIC.

In each Neighbor Tuple:

- N_orig_addr MUST NOT be changed to unknown.

- N_orig_addr MUST NOT equal this router's originator address, or equal O_orig_addr in any Originator Tuple.

- N_orig_addr MUST NOT equal the AL_net_addr in any Local Attached Network Tuple.

- If N_orig_addr != unknown, then N_orig_addr MUST NOT equal the N_orig_addr in any other Neighbor Tuple.

- N_neighbor_addr_list MUST NOT contain any network address which includes this router's originator address, the O_orig_addr in any Originator Tuple, or equal or have as a sub-range the AL_net_addr in any Local Attached Network Tuple.

- If N_orig_addr = unknown, then N_will_flooding = WILL_NEVER, N_will_routing = WILL_NEVER, N_flooding_mpr, N_routing_mpr = false, N_mpr_selector = false, and N_advertised = false.

- N_in_metric MUST equal the minimum value of the L_in_metric values of all corresponding Link Tuples with L_status = SYMMETRIC and L_in_metric != UNKNOWN_METRIC, if any, otherwise N_in_metric = UNKNOWN_METRIC.

- N_out_metric MUST equal the minimum value of the L_out_metric values of all corresponding Link Tuples with L_status = SYMMETRIC and L_out_metric != UNKNOWN_METRIC, if any, otherwise N_out_metric = UNKNOWN_METRIC.

- N_will_flooding and N_will_routing MUST be in the range from WILL_NEVER to WILL_ALWAYS, inclusive.

- If N_flooding_mpr = true, then N_symmetric MUST be true, N_out_metric MUST NOT equal UNKNOWN_METRIC and N_will_flooding MUST NOT equal WILL_NEVER.

- If N_routing_mpr = true, then N_symmetric MUST be true, N_in_metric MUST NOT equal UNKNOWN_METRIC and N_will_routing MUST NOT equal WILL_NEVER.

- If N_symmetric = true and N_flooding_mpr = false, then N_will_flooding MUST NOT equal WILL_ALWAYS.

- If N_symmetric = true and N_routing_mpr = false, then N_will_routing MUST NOT equal WILL_ALWAYS.

- If N_mpr_selector = true, then N_advertised MUST be true.

- If N_advertised = true, then N_symmetric MUST be true and N_out_metric MUST NOT equal UNKNOWN_METRIC.

In each Lost Neighbor Tuple:

- NL_neighbor_addr MUST NOT include this router's originator address, the O_orig_addr in any Originator Tuple, or equal or have as a sub-range the AL_net_addr in any Local Attached Network Tuple.

In each 2-Hop Tuple:

- N2_2hop_addr MUST NOT equal this router's originator address, equal the O_orig_addr in any Originator Tuple, or equal or have as a sub-range the AL_net_addr in any Local Attached Network Tuple.

- if N2_in_metric != UNKNOWN_METRIC then N2_in_metric MUST be representable in the defined compressed form.

- if N2_out_metric != UNKNOWN_METRIC then N2_out_metric MUST be representable in the defined compressed form.

In each Advertising Remote Router Tuple:

- AR_orig_addr MUST NOT be in any network address in the I_local_iface_addr_list in any Local Interface Tuple or be in the IR_local_iface_addr in any Removed Interface Address Tuple.

- AR_orig_addr MUST NOT equal this router's originator address or equal the O_orig_addr in any Originator Tuple.

- AR_orig_addr MUST NOT be in the AL_net_addr in any Local Attached Network Tuple.
- AR_orig_addr MUST NOT equal the AR_orig_addr in any other Advertising Remote Router Tuple.

In each Router Topology Tuple:

- There MUST be an Advertising Remote Router Tuple with AR_orig_addr = TR_from_orig_addr.
- TR_to_orig_addr MUST NOT be in any network address in the I_local_iface_addr_list in any Local Interface Tuple or be in the IR_local_iface_addr in any Removed Interface Address Tuple.
- TR_to_orig_addr MUST NOT equal this router's originator address or equal the O_orig_addr in any Originator Tuple.
- TR_to_orig_addr MUST NOT be in the AL_net_addr in any Local Attached Network Tuple.
- The ordered pair (TR_from_orig_addr, TR_to_orig_addr) MUST NOT equal the corresponding pair for any other Router Topology Tuple.
- TR_seq_number MUST NOT be greater than AR_seq_number in the Advertising Remote Router Tuple with AR_orig_addr = TR_from_orig_addr.
- TR_metric MUST be representable in the defined compressed form.

In each Routable Address Topology Tuple:

- There MUST be an Advertising Remote Router Tuple with AR_orig_addr = TA_from_orig_addr.
- TA_dest_addr MUST be routable.
- TA_dest_addr MUST NOT overlap any network address in the I_local_iface_addr_list in any Local Interface Tuple or overlap the IR_local_iface_addr in any Removed Interface Address Tuple.
- TA_dest_addr MUST NOT include this router's originator address or include the O_orig_addr in any Originator Tuple.
- TA_dest_addr MUST NOT equal or have as a sub-range the AL_net_addr in any Local Attached Network Tuple.

- The ordered pair (TA_from_orig_addr, TA_dest_addr) MUST NOT equal the corresponding pair for any other Attached Network Tuple.

- TA_seq_number MUST NOT be greater than AR_seq_number in the Advertising Remote Router Tuple with AR_orig_addr = TA_from_orig_addr.

- TA_metric MUST be representable in the defined compressed form.

In each Attached Network Tuple:

- There MUST be an Advertising Remote Router Tuple with AR_orig_addr = AN_orig_addr.

- AN_net_addr MUST NOT equal or be a sub-range of any network address in the I_local_iface_addr_list in any Local Interface Tuple or be a sub-range of the IR_local_iface_addr in any Removed Interface Address Tuple.

- AN_net_addr MUST NOT equal this router's originator address or equal the O_orig_addr in any Originator Tuple.

- AN_net_addr MUST NOT equal the AL_net_addr in any Local Attached Network Tuple.

- The ordered pair (AN_orig_addr, AN_net_addr) MUST NOT equal the corresponding pair for any other Attached Network Tuple.

- AN_seq_number MUST NOT be greater than AR_seq_number in the Advertising Remote Router Tuple with AR_orig_addr = AN_orig_addr.

- AN_dist MUST NOT be less than zero.

- AN_metric MUST be representable in the defined compressed form.

Appendix B. Example Algorithm for Calculating MPRs

The following specifies an algorithm which MAY be used to select an MPR Set given a Neighbor Graph, as defined in Section 18.2 and Section 18.3.

This algorithm selects an MPR Set M that is a subset of the set N1 that is part of the Neighbor Graph. This algorithm assumes that a subset I of N1 is pre-selected as MPRs, i.e., that M will contain I.

B.1. Additional Notation

The following additional notation, in addition to that in Section 18.2 will be used by this algorithm:

N:
A subset of N2, consisting of those elements y in N2 such that either d1(y) is not defined, or there is at least one x in N1 such that d(x,y) is defined and d(x,y) < d1(y).

D(x):
For an element x in N1, the number of elements y in N for which d(x,y) is defined and has minimal value among the d(z,y) for all z in N1.

R(x,M):
For an element x in N1, the number of elements y in N for which d(x,y) is defined, has minimal value among the d(z,y) for all z in N1, and no such minimal values have z in M. (Note that, denoting the empty set by 0, D(x) = R(x,0).)

B.2. MPR Selection Algorithm

To create the MPR Set M, starting with M := I:

1. Add all elements x in N1 that have W(x) = WILL_ALWAYS to M.

2. For each element y in N for which there is only one element x in N1 such that d2(x,y) is defined, add that element x to M.

3. While there exists any element x in N1 with R(x,M) > 0:

    1. Select an element x in N1 with R(x,M) > 0 in the following order of priority, and then add to M:

        + greatest W(x), THEN;

        + greatest R(x,M), THEN;

        + greatest D(x), THEN;

        + any choice, which MAY be based on other criteria (for example a router MAY choose to prefer a neighbor as an MPR if that neighbor has already selected the router as an MPR of the same type, MAY prefer a neighbor based on information freshness, or MAY prefer a neighbor based on length of time previously selected as an MPR) or MAY be random.

4. OPTIONAL: consider each element x in M, but not in I, in turn and if x can be removed from M while still leaving it satisfying the definition of an MPR Set, then remove that element x from M. Elements MAY be considered in any order, e.g., in order of increasing W(x).

Appendix C. Example Algorithm for Calculating the Routing Set

The following procedure is given as an example for calculating the Routing Set using a variation of Dijkstra's algorithm. First all Routing Tuples are removed, and then, using the selections and definitions in Appendix C.1, the procedures in the following sections (each considered a "stage" of the processing) are applied in turn.

C.1. Local Interfaces and Neighbors

The following selections and definitions are made:

1. For each Local Interface Tuple, select a network address from its I_local_iface_addr_list, this is defined as the selected address for this Local Interface Tuple.

2. For each Link Tuple, the selected address of its corresponding Local Interface Tuple is defined as the selected local address for this Local Interface Tuple.

3. For each Neighbor Tuple with N_symmetric = true and N_out_metric != UNKNOWN_METRIC, select a Link Tuple with L_status = SYMMETRIC for which this is the corresponding Neighbor Tuple and has L_out_metric = N_out_metric. This is defined as the selected Link Tuple for this Neighbor Tuple.

4. For each network address (N_orig_addr or in N_neighbor_addr_list, the "neighbor address") from a Neighbor Tuple with N_symmetric = true and N_out_metric != UNKNOWN_METRIC, select a Link Tuple (the "selected Link Tuple") from those for which this is the corresponding Neighbor Tuple, have L_status = SYMMETRIC, and have L_out_metric = N_out_metric, by:

    1. If there is such a Link Tuple whose L_neighbor_iface_addr_list contains the neighbor address, select that Link Tuple.

    2. Otherwise select the selected Link Tuple for this Neighbor Tuple.

    Then for this neighbor address:

3. The selected local address is defined as the selected local address for the selected Link Tuple.

4. The selected link address is defined as an address from the L_neighbor_iface_addr_list of the selected Link Tuple, if possible equal to this neighbor address.

5. Routing Tuple preference is decided by preference for minimum R_dist, then for minimum R_dist, and then for preference for corresponding Neighbor Tuples in this order:

   * For greater N_will_routing.

   * For N_mpr_selector = true over N_mpr_selector = false.

   Note that preferred Routing Tuples SHOULD be used. Routing Tuples with minimum R_metric MUST be used, this is specified outside the definition of preference. An implementation MAY modify this definition of preference (including for minimum R_dist) without otherwise affecting this algorithm.

C.2. Add Neighbor Routers

The following procedure is executed once.

1. For each Neighbor Tuple with N_symmetric = true and N_out_metric != UNKNOWN_METRIC, add a Routing Tuple with:

   * R_dest_addr := N_orig_addr;

   * R_next_iface_addr := selected link address for N_orig_addr;

   * R_local_iface_addr := selected local address for N_orig_addr;

   * R_metric := N_out_metric;

   * R_dist := 1.

C.3. Add Remote Routers

The following procedure is executed once.

1. Add a label that may be "used" or "unused" to each Routing Tuple, with all initial values equal to unused. (Note that this label is only required during this algorithm.)

2. If there are no unused Routing Tuples then this stage is complete, otherwise repeat the following until that is the case.

1. Find the unused Routing Tuple with minimum R_metric (if more than one, pick any) and denote it the "current Routing Tuple".

2. Mark the current Routing Tuple as used.

3. For each Router Topology Tuple, with:

   + TR_from_orig_addr = R_dest_addr of the current Routing Tuple.

   2. Define:

      - new_metric := R_metric of the current Routing Tuple + TR_metric;

      - new_dist := R_dist of the current Routing Tuple + 1.

   3. If there is no Routing Tuple with R_dest_addr = TR_to_orig_addr, then create an unused Routing Tuple with:

      - R_dest_addr := TR_to_orig_addr;

      - R_next_iface_addr := R_next_iface_addr of the current Routing Tuple;

      - R_local_iface_addr := R_local_iface_addr of the current Routing Tuple;

      - R_metric := new_metric;

      - R_dist := new_dist.

   4. Otherwise, if there is an unused Routing Tuple with R_dest_addr = TR_to_orig_addr, and either new_metric < R_metric or (new_metric = R_metric and the updated Routing Tuple would be preferred) then update this Routing Tuple to have:

      - R_next_iface_addr := R_next_iface_addr of the current Routing Tuple;

      - R_local_iface_addr := R_local_iface_addr of the current Routing Tuple;

      - R_metric := new_metric;

- R_dist := new_dist.

C.4. Add Neighbor Addresses

The following procedure is executed once.

1. For each Neighbor Tuple with N_symmetric = true and N_out_metric != UNKNOWN_METRIC:

    1. For each network address (the "neighbor address") in N_neighbor_addr_list, if the neighbor address is not equal to the R_dest_addr of any Routing Tuple, then add a new Routing Tuple, with:

        + R_dest_addr := neighbor address;

        + R_next_iface_addr := selected link address for the neighbor address;

        + R_local_iface_addr := selected local address for the neighbor address;

        + R_metric := N_out_metric;

        + R_dist := 1.

C.5. Add Remote Routable Addresses

The following procedure is executed once.

1. For each Routable Address Topology Tuple, if:

    * TA_dest_addr is not equal to the R_dest_addr of any Routing Tuple added in an earlier stage; AND

    * TA_from_orig_addr is equal to the R_dest_addr of a Routing Tuple (the "previous Routing Tuple"),

    then add a new Routing Tuple, with:

    * R_dest_addr := TA_dest_addr;

    * R_next_iface_addr := R_next_iface_addr of the previous Routing Tuple;

    * R_local_iface_addr := R_local_iface_addr of the previous Routing Tuple;

* R_metric := R_metric of the previous Routing Tuple + TA_metric.

* R_dist := R_dist of the previous Routing Tuple + 1.

There may be more than one Routing Tuple that may be added for an R_dest_addr in this stage. If so, then, for each such R_dest_addr, a Routing Tuple with minimum R_metric MUST be selected, otherwise a Routing Tuple which is preferred SHOULD be added.

C.6. Add Attached Networks

The following procedure is executed once.

1. For each Attached Network Tuple, if:

   * AN_net_addr is not equal to the R_dest_addr of any Routing Tuple added in an earlier stage; AND

   * AN_orig_addr is equal to the R_dest_addr of a Routing Tuple (the "previous Routing Tuple),

   then add a new Routing Tuple, with:

   * R_dest_addr := AN_net_addr;

   * R_next_iface_addr := R_next_iface_addr of the previous Routing Tuple;

   * R_local_iface_addr := R_local_iface_addr of the previous Routing Tuple;

   * R_metric := R_metric of the previous Routing Tuple + AN_metric;

   * R_dist := R_dist of the previous Routing Tuple + AN_dist.

   There may be more than one Routing Tuple that may be added for an R_dest_addr in this stage. If so, then, for each such R_dest_addr, a Routing Tuple with minimum R_metric MUST be selected, otherwise a Routing Tuple which is preferred SHOULD be added.

C.7. Add 2-Hop Neighbors

The following procedure is executed once.

1. For each 2-Hop Tuple with N2_out_metric != UNKNOWN_METRIC, if:

   * N2_2hop_addr is a routable address; AND

   * N2_2hop_addr is not equal to the R_dest_addr of any Routing Tuple added in an earlier stage; AND

   * the Routing Tuple with R_dest_addr = N_orig_addr of the corresponding Neighbor Tuple (the "previous Routing Tuple") has R_dist = 1,

   then add a new Routing Tuple, with:

   * R_dest_addr := N2_2hop_addr;

   * R_next_iface_addr := R_next_iface_addr of the previous Routing Tuple;

   * R_local_iface_addr := R_local_iface_addr of the previous Routing Tuple;

   * R_metric := R_metric of the previous Routing Tuple + N_out_metric of the corresponding Neighbor Tuple;

   * R_dist := 2.

   There may be more than one Routing Tuple that may be added for an R_dest_addr in this stage. If so, then, for each such R_dest_addr, a Routing Tuple with minimum R_metric MUST be selected, otherwise a Routing Tuple which is preferred SHOULD be added.

Appendix D. TC Message Example

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

The TC message's four bit Message Flags (MF) field has value 15

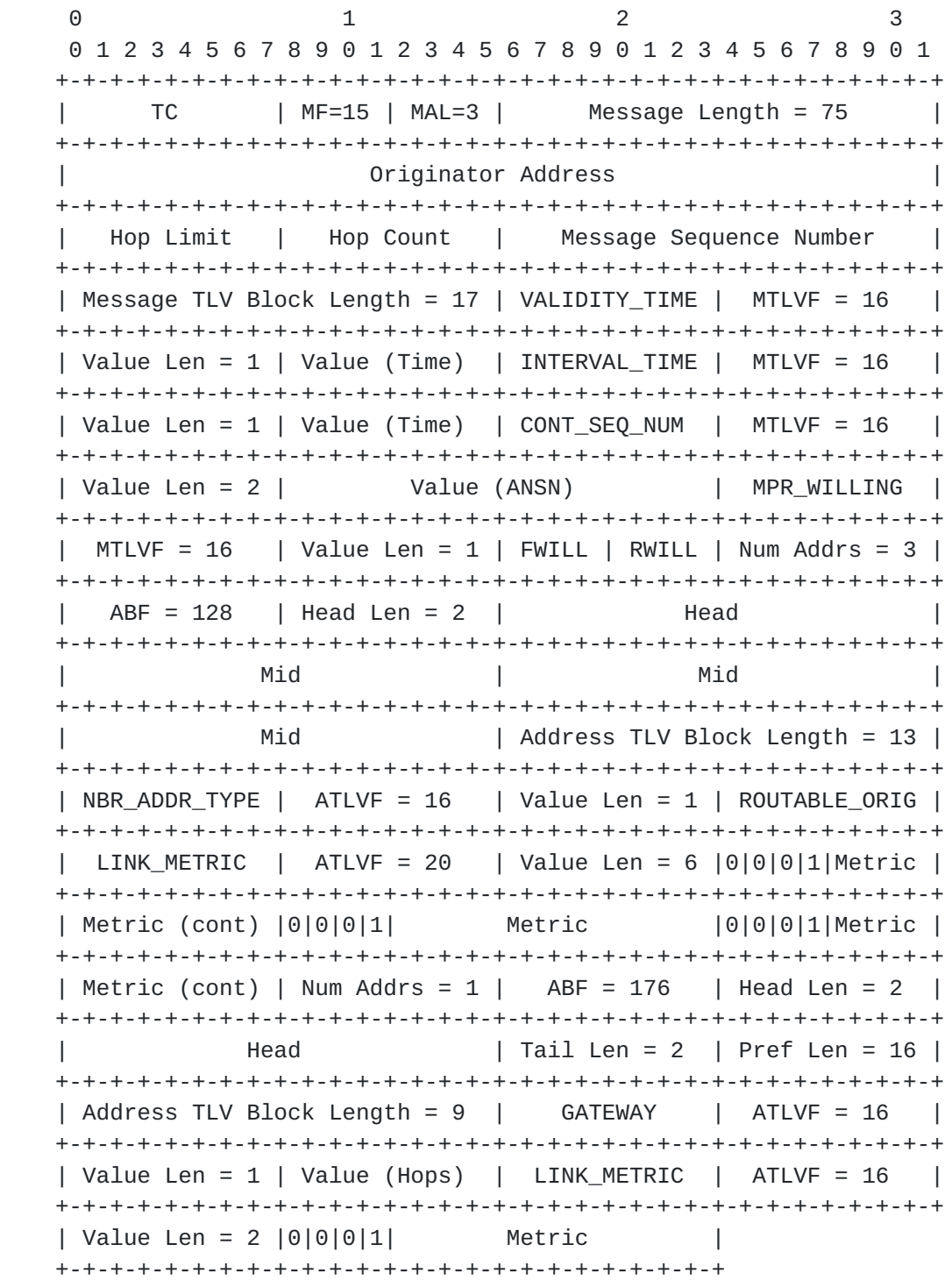
indicating that the message header contains originator address, hop limit, hop count, and message sequence number fields. Its four bit Message Address Length (MAL) field has value 3, indicating addresses in the message have a length of four octets, here being IPv4 addresses. The overall message length is 75 octets.

The message has a Message TLV Block with content length 17 octets containing four TLVs. The first two TLVs are validity and interval times for the message. The third TLV is the content sequence number TLV used to carry the 2 octet ANSN, and (with default type extension zero, i.e., COMPLETE) indicating that the TC message is complete. The fourth TLV contains forwarding and routing willingness values for the originating router (FWILL and RWILL, respectively). Each TLV uses a TLV with Flags octet (MTLVF) value 16, indicating that it has a Value, but no type extension or start and stop indexes. The first two TLVs have a Value Length of 1 octet, the last has a Value Length of 2 octets.

The message has two Address Blocks. (This is not necessary, the information could be conveyed using a single Address Block, the use of two Address Blocks, which is also allowed, is illustrative only.) The first Address Block contains 3 addresses, with Flags octet (ATLVF) value 128, hence with a Head section (with length 2 octets), but no Tail section, and hence with Mid sections with length two octets. The following TLV Block (content length 13 octets) contains two TLVs. The first TLV is a NBR_ADDR_TYPE TLV with Flags octet (ATLVF) value 16, indicating a single Value but no indexes. Thus all these addresses are associated with the Value (with Value Length 1 octet) ROUTABLE_ORIG, i.e., they are originator addresses of advertised neighbors that are also routable addresses. The second TLV is a LINK_STATUS TLV with Flags octet (ATLVF) value 20, indicating a Value for each address, i.e., as the total Value Length is 6 octets, each address is associated with a Value with length two octets. These Value fields are each shown as having four bits indicating that they are outgoing neighbor metric values, and as having twelve bits that represent the metric value (the first four bits being the exponent, the remaining twelve bits the mantissa).

The second Address Block contains 1 address, with Flags octet (ATLVF) 176, indicating that there is a Head section (with length 2 octets), that the Tail section (with length 2 octets) consists of zero valued octets (not included), and that there is a single prefix length, which is 16. The network address is thus Head.0.0/16. The following TLV Block (content length 8 octets) includes two TLVs. The first has a Flags octet (ATLVF) of 16, again indicating that no indexes are needed, but that a Value (with Value Length 1 octet) is present, indicating the address distance as a number of hops. The second TLV is another LINK_METRIC TLV, as in the first Address TLV Block except

with a Flags octet (ATLVF) value 16, indicating that a single Value is present.



Appendix E. Flow and Congestion Control

Due to its proactive nature, this protocol has a natural control over the flow of its control traffic. Routers transmit control messages at predetermined rates specified and bounded by message intervals.

This protocol employs [RFC6130] for local signaling, embedding MPR selection advertisement through a simple Address Block TLV, and router willingness advertisement (if any) as a single Message TLV. Local signaling, therefore, shares the characteristics and constraints of [RFC6130].

Furthermore, the use of MPRs can greatly reduce the signaling overhead from link state information dissemination in two ways, attaining both flooding reduction and topology reduction. First, using MPR flooding, the cost of distributing link state information throughout the network is reduced, as compared to when using blind flooding, since only MPRs need to forward link state declaration messages. Second, the amount of link state information for a router to declare is reduced to need only contain that router's MPR selectors. This reduces the size of a link state declaration as compared to declaring full link state information. In particular some routers may not need to declare any such information. In dense networks, the reduction of control traffic can be of several orders of magnitude compared to routing protocols using blind flooding [MPR]. This feature naturally provides more bandwidth for useful data traffic and pushes further the frontier of congestion.

Since the control traffic is continuous and periodic, it keeps the quality of the links used in routing more stable. However, using some options, some control messages (HELLO messages or TC messages) may be intentionally sent in advance of their deadline in order to increase the responsiveness of the protocol to topology changes. This may cause a small, temporary, and local increase of control traffic, however this is at all times bounded by the use of minimum message intervals.

A router that recognizes that the network is suffering from congestion can increase its message interval parameters. If this is done by most or all routers in the network, then the overall control traffic in the network will be reduced. Routers will have to take care in using this capability not to increase message interval parameters such that they cannot cope with network topology changes. Note that routers can make such decisions independently, it is not necessary for all routers to be using the same parameter values, nor is it necessary that all routers decide to change their intervals at the same time.

Authors' Addresses

Thomas Heide Clausen
LIX, Ecole Polytechnique

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

Christopher Dearlove
BAE Systems ATC

Phone: +44 1245 242194
EMail: chris.dearlove@baesystems.com
URI: http://www.baesystems.com/

Philippe Jacquet
Alcatel-Lucent Bell Labs

Phone: +33 6 7337 1880
EMail: philippe.jacquet@alcatel-lucent.com

Ulrich Herberg
Fujitsu Laboratories of America
1240 E. Arques Ave.
Sunnyvale, CA, 94085
USA

EMail: ulrich@herberg.name
URI: http://www.herberg.name/