INTERNET-DRAFT
IETF MANET Working Group
Expiration: 02 September 2001

Philippe Jacquet
Paul Muhlethaler
Amir Qayyum
Anis Laouiti
Laurent Viennot
Thomas Clausen
INRIA Rocquencourt, France
2 March 2001

Optimized Link State Routing Protocol
draft-ietf-manet-olsr-04.txt

Abstract

This document describes the Optimized Link State Routing (OLSR) protocol for mobile ad hoc networks. The protocol is an optimization of the pure link state algorithm tailored to the requirements of a mobile wireless LAN. The key concept used in the protocol is that of multipoint relays (MPRs) [1] & [2]. MPRs are selected nodes which forward broadcast messages during the flooding process. This technique substantially reduces the message overhead as compared to pure flooding mechanism where every node retransmits each message when it receives the first copy of the packet. In OLSR, information flooded in the network "through" these MPRs is also "about" the MPRs. Thus a

second optimization is achieved by minimizing the "contents" of the control messages flooded in the network. Hence, as contrary to the classic link state algorithm, only a small subset of links with the neighbor nodes are declared instead of all the links. This information is then used by the OLSR protocol for route calculation. As a consequence hereof, the routes contain only the MPRs as intermediate nodes from a Source to a Destination. OLSR provides optimal routes (in terms of number of hops). The protocol is particularly suitable for large and dense networks as the technique of MPRs works well in this context.

Contents

1. Introduction

This Optimized Link State Routing protocol inherits the concept of forwarding and relaying from HIPERLAN (a MAC layer protocol) which is standardized by ETSI [3]. The OLSR protocol is developed in the IPANEMA project (part of Euclid program) and in the PRIMA project (part of RNRT program).

This protocol is developed for mobile ad hoc networks. It operates as a table driven and proactive protocol and exchanges topology information with other nodes of the network at regular intervals. The nodes which are selected as a multipoint relay by some neighbor nodes announce this information periodically in their control messages. The protocol uses the MPRs to facilitate efficient flooding of control messages in the network. In route calculation, the MPRs are used to form the route from a given node to any destination in the network.

MPRs are selected by a node among its one hop neighbors with "symmetric", i.e. bi-directional, link. Therefore, selecting the route through MPRs automatically avoids the problems associated with data packet transfer on uni-directional links (such as the problem of not getting link-layer acknowledgments for the data packets at each hop)

The OLSR protocol is developed to work independently from other protocols. But it can be adapted to operate with a protocol (like IMEP [4]) which could provide common functionalities such as neighbor sensing, multipoint relaying, security authentication, etc.

2. Changes

Major changes from version 03 to version 04

- Finalized the generic packet/message format to include features for scope-limited (diameter-bound) flooding of messages and to handle duplicate messages.

- Editorial changes towards language consistency.

Major changes from version 02 to version 03

- Introduction of assigned port number for use with OLSR.

- The packet format now uses "message length" rather than an offset to the next message.

- Optional section describing how link-layer notifications can be utilized included.

Major changes from version 01 to version 02

- Introduction of a unified packet format for encapsulation of all messages being exchanged between nodes. This also serves to facilitate extensions in future versions of the protocol (i.e. introduction of new protocol messages) without breaking backwards compatibility.

- Removal of "Power Conservation" from this draft. Power Conservation may be considered as an extension to the basic routing capabilities, and the information is therefore moved to draft-ietf-manet-olsr-extensions-00.txt.

3. OLSR Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119 [9]. The OLSR protocol uses the following terminology, in addition to the terms defined in [5].

connection

A communication channel or medium *on the same physical interface*, over which the nodes can communicate with each other.

holding time

The lifetime associated with an entry in any table. An entry is kept in the table for a period of time, equal to its holding time. If the entry is not refreshed during this period, it is removed from the table when the holding time expires.

multipoint relay (MPR)

A node which is selected by its one-hop neighbor, node X, to "re-transmit" all the broadcast messages that it receives from X, provided that the same message is not already received, and the time to live field of the message is greater than zero.

multipoint relay selector (MPR selector, MS)

A node which has selected its one-hop neighbor, node X, as its multipoint relay, will be called a multipoint relay selector of node X.

node

A MANET router which implements this Optimized Link State Routing protocol.

symmetric link

A bi-directional *link* between two neighbor nodes, i.e. node X and node Y where both can hear each other.

4. Applicability Section

This section dictates the characteristics of the OLSR protocol as specified in the Applicability Statement draft [6].

4.1. Networking Context

OLSR is well suited to large and dense mobile networks, as the optimi- zation achieved using the MPRs works well in this context. The larger and more dense a network, the more optimization can be achieved as compared to the normal link state algorithm. OLSR uses hop-by-hop routing, i.e. each node uses its local information to route packets.

OLSR is well suited for networks, where the traffic is random and sporadic between "several" nodes rather than being almost exclusively between a small specific set of nodes. The performance of the protocol, compared to a reactive protocol, is even better if these [source, destination] pairs change with time [8]. Such changes may initiate substantial traffic (Query flooding) in case of reactive protocol, but nothing in OLSR, as the routes are maintained for each known destination all the time.

4.2. Protocol Characteristics and Mechanisms

* Does the protocol provide shortest path routes ?

  Yes.

* Does the protocol provide support for unidirectional links? (if so, how?)

  No. However, the use of uni-directional links may easily be enabled through optional extensions to the protocol.

* Does the protocol require the use of tunneling? (if so, how?)

  No.

* Does the protocol require using some form of source routing? (if so, how?)

  No.

* Does the protocol require the use of periodic messaging? (if so, how?)

  Yes. Periodically, each node in the network sends a message containing the addresses of the neighbors which have selected that node as a MPR. This information enables other nodes to build routes to that node through the MPRs.

* Does the protocol require the use of reliable or sequenced packet delivery? (if so, how?)

  No.

* Does the protocol provide support for routing through a multi-technology routing fabric? (if so, how?)

  No. However, provisions for multiple interfaces may easily be enabled through extensions to the protocol.

* Does the protocol provide support for multiple hosts per router? (if so, how?)

  Yes. The hosts are added to the MPR selector set of the node (router), which will then announce that the hosts can be reached through that node.

* Does the protocol support the IP addressing architecture? (if so, how?)

  Yes. Nodes are assigned and addressed by regular IP-addresses.

* Does the protocol require link or neighbor status sensing (if so, how?)

  Yes. The protocol requires link status sensing. This service is provided by sending/receiving periodic HELLO messages to/from one hop neighbors.

* Does the protocol depend on a central entity? (if so, how?)

  No.

* Does the protocol function reactively? (if so, how?)

  No.

* Does the protocol function proactively? (if so, how?)

  Yes. Each node periodically sends information about its MPR selectors, which enables the nodes to construct routes to these MPR selectors through the node.

* Does the protocol provide loop-free routing? (if so, how?)

  Yes. As the protocol uses a link state algorithm, routing is loop-free when in a stable state.

* Does the protocol provide for sleep period operation? (if so, how?)

  No. However, provisions for sleep-operation may easily be enabled through extensions to the protocol.

* Does the protocol provide some form of security? (if so, how?)

  No.

* Does the protocol provide support for utilizing multi-channel, link-layer technologies? (if so, how?)

  Yes. OLSR makes no assumptions on the underlying link-layer other, than that local broadcast must be available.

5. Protocol Overview

OLSR is a proactive routing protocol for mobile ad hoc networks. The 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. OLSR is an optimization over the pure link state protocol, tailored for mobile ad hoc networks.

Firstly, it reduces the size of the control messages: rather than declaring all links, a node declares only a subset of links with its neighbors, namely the links to those nodes which are its MPR selectors (see section 6 on MPRs). Secondly, OLSR minimizes flooding of control traffic by using only selected nodes, called MPRs, to diffuse its messages. This technique significantly reduces the number of retransmissions in a flooding or broadcast procedure.

OLSR MAY optimize the reactivity to topological changes by reducing the time interval for periodic control message transmission. Furthermore, as OLSR keeps the routes for all destinations in the network, the protocol is beneficial for traffic patterns where a large subset of nodes are communicating with another large subset of nodes, and where the [source,destination] pairs are changing over time. The protocol is particularly suited for large and dense networks, as the optimization done using the MPRs works well in this context. The larger and more dense a network, the more optimization can be achieved as compared to the normal link state algorithm.

OLSR is designed to work in a completely distributed manner and does thus not depend on any central entity. The protocol does NOT REQUIRE reliable transmission for control messages: each node sends control messages periodically, and can therefore sustain an occasional loss of some such messages. Such losses occur frequent in radio networks due to collisions or other transmission problems.

Also, OLSR does NOT REQUIRE sequenced delivery of messages. Each control message contains a sequence number which is incremented for each message. Thus the recipient of a control message can easily identify which information is newer - even if messages have been re-ordered while in transmission.

Furthermore, OLSR provides support for protocol extensions such as sleep mode operation, multicast-routing etc. Such extensions may be introduced as additions to the protocol without breaking backwards compatibility with earlier versions.

OLSR performs hop by hop routing, i.e. each node uses its most recent local information to route a packet. Hence for OLSR to be able to route packets, the frequency of control messages should be tuned to the speed of the mobile nodes such that their movements can be tracked by their neighborhood.

OLSR does NOT REQUIRE any changes to the format of IP packets. Thus any existing IP stack can be used as it is: the protocol only interacts with routing table management.

6. Multipoint Relays

The idea of multipoint relays is to minimize the overhead of flooding messages in the network by reducing duplicate retransmissions in the same region. Each node in the network selects a set of nodes in its neighborhood which may retransmit its messages. This set of selected neighbor nodes is called the "Multipoint Relay" (MPR) set of that node. The neighbors of node N which are *NOT* in its MPR set, receive and process broadcast messages but do not retransmit broadcast messages received from node N.

Each node selects its MPR set among its one hop neighbors. This set is selected such that it covers (in terms of radio range) all nodes that are two hops away. The neighborhood of any node N can be defined as the set of nodes which have a symmetric link to N. The 2-hop neighborhood of N can be defined as the set of nodes which don't have a symmetric link to N but have a symmetric link to the neighborhood of N. The MPR set of N, denoted as MPR(N), is then an arbitrary subset of the neighborhood of N which satisfies the following condition: every node in the 2-hop neighborhood of N must have a symmetric link toward MPR(N). The smaller the MPR set is, the more optimal is the routing protocol. [2] gives an analysis and example about MPR selection algorithms.

Each node maintains information about a set of its neighbors. This is the set of neighbors, called the "Multipoint Relay Selector set" (MPR selector set), which have selected the node as a MPR. A node obtains this information from the periodic HELLO messages received from the neighbors. A broadcast message, intended to be diffused in the whole network, coming from these MPR selector neighbor nodes is assumed to be retransmitted by the node. This set can change over time (i.e. when a node selects another MPR-set) and is indicated by the selector nodes in their HELLO messages. Each node has a specific "Multipoint relay Selector Sequence Number" (MSSN) associated with this set. Whenever its MPR selector set is updated, the node also increments its MSSN.

OLSR relies on selection of MPRs, and calculates routes through these nodes. I.e. MPR nodes are selected as intermediate nodes in the path between a source and a destination. To enable this, each node in the network periodically broadcast the information describing which neighbors have selected it as a MPR. Upon receipt of this "MPR Selector" information, each node calculates or updates the route to each known destination. So principally, the route is a sequence of hops through the MPRs from source to the destination.

MPRs are selected among the one hop neighbors with "symmetric" i.e. bi-directional link. Therefore, selecting the route through MPRs automatically avoids the problems associated with data packet transfer on uni-directional links such as the problem of not getting an acknowledgment for the data packets at each hop.

7. Protocol Functioning

This section describes the details of the protocol functioning. This includes descriptions of the format and contents of the packets being exchanged by routers, the algorithms (e.g. for packet handling and routing table calculation) and suggested data structures internally in each router.

7.1. Protocol and Port Number

Packages in OLSR are communicated using UDP. Port 698 has been assigned by IANA for exclusive usage by the OLSR protocol.

7.2. Packet Format

OLSR communicates using an unified packet format for all data related to the protocol. The purpose of this is to facilitate extensibility of the protocol without breaking backwards compatibility as well as to provide an easy way of piggybacking different "types" of information into a single transmission. These packets are embedded in UDP datagrams for transmission over the network. The present draft uses IPv4 addresses. Support for IPv6 will be included in a future draft.

Each package encapsulates one or more messages. The messages share a common header format, which enables nodes to correctly accept and (if applicable) retransmit messages of an unknown type.

Messages can be flooded onto the entire network, or flooding can be limited to nodes within a diameter (in terms of number of hops) from the originator of the message. Thus, broadcasting a message to a nodes neighborhood is just a special case of flooding. When flooding any control message, duplicate retransmissions will be eliminated locally (i.e. each node maintains a duplicate table to prevent transmitting the same message twice) and minimized in the entire network through the usage of MPRs as described in section 5 and 6.

Furthermore, a node can examine the header of a message to obtain information on the distance (in terms of number of hops) to the originator of the message. This feature may be useful in situations where, e.g., the time information from a recieved control messages is stored in a node depends on the distance to the originator.

The basic layout of any packet in OLSR will be as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Packet Length         |    Reserved for future use    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Originator Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Message Size          |  Time To Live |   Hop Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Message Type |            Message Sequence Number            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
:                            MESSAGE                            :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Originator Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Message Size          |  Time To Live |   Hop Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Message Type |            Message Sequence Number            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
:                            MESSAGE                            :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
         (etc)
```

7.2.1. Packet Header

Packet Length

The length (in bytes) of the packet

Reserved for future use

MUST be set to '0000000000000000' to be in compliance with this version of the draft.

The sender information for a packet is obtainable from the UDP header.

7.2.2. Message Header

Originator Address

This field contains the address of the node, which has originally generated this message. This field SHOULD NOT be confused with the source address from the UDP header, which is changed each time to the address of the intermediate node which is "re-transmitting" this message. The Originator Address field MUST *NEVER* be changed in the retransmissions.

Message Sequence Number

While generating the TC message, the "originator" node will assign a unique identification number to each message. This number is inserted into the Sequence Number field of the message. The sequence number is increased by 1 (one) for each message originating from the node - "wrap-arounds" are handled as described in section 10.

Message Size

This field gives the size of this message, measured from the beginning of the "Message Type" field and until the beginning of the next "Message Type" field (or - if there are no following messages - the end of the packet).

Message Type

This field indicates which type of message are to be found in the "MESSAGE" partition. Message types in the range of 0-127 are reserved for messages in this draft and in draft-ietf-manet-olsr-extensions-00.txt.

Time To Live

This field contains the maximum number of hops a message will be retransmitted. Before a message is transmitted, the Time To Live MUST be decremented by 1. When a node receives a message with a Time To Live equal to 0, the message MUST NOT be retransmitted under any circumstances.

Thus, by setting this field, the originator of a message can limit the flooding radius.

Hop Count

This field will contain the number of hops a message has attained. Before a message is (re-) transmitted, the Hop Count MUST be incremented by 1.

Initially, this is set to '0' by the originator of the message.

Reserved

This field is reserved for future usage, and MUST be set to '00000000' for compliance with this draft.

7.2.3. Packet Processing

Upon receiving a basic packet, the protocol parser examines each of the "message headers". Based on the value of the "Message Type" field, the parser can determine the faith of the message. A node may receive the same message in several packets. This can happen only if the message is retransmitted by two nodes in the receivers neighborhood, i.e. the "Time To Live" and the "Hop Count" fields in the message satisfies the following condition:

Time To Live + Hop Count > 1

Thus, to avoid re-processing of a message which was already received and processed, each node maintains a Duplicate table. In this table, the node records information about the most recently received messages where the above condition holds. For each message, satisfying the above condition, a node records a "Duplicate Tuple" (D_addr, D_seq_num, D_time), where D_addr is the originator address of the message, D_seq_num is the message sequence number of the message and D_time specifies the time at which a tuple expires and *MUST* be removed.

In a node, the set of Duplicate Tuples are denoted the "Duplicate set".

Thus, upon receiving a basic packet, a node performs the following tasks for each encapsulated message:

1. If there exists a tuple in the duplicate set, where:

   D_addr == Originator Address, AND

   D_seq_num == Message Sequence Number

   then the message has already been completely processed and MUST silently be ignored.

2. Otherwise, if the Message Type of the message is known to the node, the message MUST be processed according to the specifications of such message type.

3. Otherwise, If the Message Type of the message is not known to the node, the message MUST be processed according to the following algorithm:

   3.1 If the sender of the message is not in the MPR selector set of the node, the message MUST silently be dropped.

   3.2 If the time to live of the message is less than or equal to '0' (zero), the message MUST silently be dropped.

   3.3 Otherwise, if the sender of the message is an MPR selector of this node and if the time to live of the message is greater than '0' (zero), the message MUST be forwarded according to the following algorithm:

   3.3.1 The time to live of the message is reduced by one.

   3.3.2 The hop-count of the message is increased by one

   3.3.3 An entry in the duplicate set is recorded with:

   D_addr = originator address

   D_seq_num = Message Sequence Number

   D_time = current time + D_HOLD_TIME.

3.3.4 The message is retransmitted (Notice: The remaining fields of the message header SHOULD be left unmodified.)

Notice: known message types are *not* forwarded "blindly" by this algorithm. Forwarding (and setting the correct message header in the forwarded, known, message) is the responsibility of the algorithm specifying how the message is to be handled. This enables, e.g., a message type to be specified such that the message can be modified while in transit (e.g. to reflect the route the message has taken). Further, it enables that the optimization through the MPRs can be bypassed: if for some reason pure flooding of a message type is required (e.g. to transmit control information over unidirectional links), the algorithm specifying handling of these messages will simply rebroadcast the message, regardless of MPR selectors.

Finally, notice that a message, which is to be broadcast in the neighborhood, but not flooded into the entire network, (e.g. a HELLO-message) is simply specified by setting the time to live to '0' (zero), and that no duplicate entries are recorded for such messages.

By defining a set of message types, which MUST be recognized by all implementations of OLSR, it will be possible to extend the protocol through introduction of additional message types, while still be able to maintain compatibility with older implementations. The two REQUIRED message types for OLSR are:

- HELLO-messages, performing the task of neighbor sensing.
- TC-messages, performing the task of MPR information declaration.

Extensions may e.g. be PC-messages for enabling power conservation / sleep mode, multicast routing, gateway announcements, auto-configuration/address assignment etc.

7.3. Neighbor sensing

7.3.1. Neighbor sensing information base

7.3.1.1 Neighbor information

A node maintains information (obtained from the HELLO messages) about its one hop neighbors, the status of the link with these neighbors, a list of 2-hop neighbors that these one hop neighbors give access to, and an associated holding time.

Thus, for each neighbor, a node records a "Neighbor Tuple" (N_addr, N_status, N_time) where N_addr is the address of the neighbor, N_status designates the status of the link with that neighbor (MPR, symmetric, heard) and N_time specifies the time at which this record expires and *MUST* be removed.

Likewise, a node records a set of "2-hop tuples" (N_addr, N_2hop_addr, N_time), describing symmetric or MPR links between its neighbors and the 2-hop neighborhood. N_addr is the address of a neighbor, N_2hop_addr is the address of a 2-hop neighbor and N_time specifies the time at which a tuple expires and *MUST* be removed.

In a node, the set of Neighbor Tuples are denoted the "Neighbor Set" and the set of 2-hop tuples are denoted the "2-hop neighbor set".

7.3.1.2 MPR Selector information

A node maintains information (obtained from the HELLO messages) about the neighbors which have selected the node as a MPR.

Thus, a node records a MPR-selector tuple (MS_addr, MS_time), for each neighbor which has selected the node as MPR. MS_addr is the address of a node which has selected the node as MPR, and MS_time specifies the time at which a tuple expires and *MUST* be removed.

In a node, the set of MPR-tuples are denoted the "MPR selector set" A sequence number, MSSN, is associated with this set. Whenever a tuple is added or removed to this set, the MSSN is incremented by 1.

7.3.2. HELLO message broadcast

Each node should detect the neighbor nodes with which it has a direct and symmetric link. The uncertainties over radio propagation may make some links asymmetric. Consequently, all links MUST be checked in both directions in order to be considered valid.

To accomplish this, each node broadcasts HELLO messages, containing information about neighbors and their link status. The link status may either be "symmetric", "heard" (asymmetric) or "MPR". "Symmetric" indicates, that the link has been verified to be bi-directional, i.e. it is possible to transmit data in both

directions. "Heard" indicates that the node can hear HELLO messages from a neighbor, but it is not confirmed that this neighbor is also able to receive messages from the node. "MPR" indicates, that a node is selected by the sender as a MPR. A status of MPR further implies that the link is symmetric.

These control messages are broadcast to all one-hop neighbors, but are *not relayed* to further nodes. A HELLO-message contains:

- a list of addresses of neighbors, to which there exists a symmetric link;

- a list of addresses of neighbors, which have been "heard";

- a list of neighbors, which have been selected as MPRs.

The list of neighbors in a HELLO message can be partial (e.g. due to message size limitations, imposed by the network), the rule being that all neighbor nodes are cited at least once within a predetermined refreshing period (HELLO_INTERVAL).

To accommodate for the above constraints, as well as to accommodate for future extensions, an approach similar to the overall packet format (see section 6.1) is taken. Thus the proposed format of a HELLO message is:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Link Type   |   Reserved    |       Link Message Size       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Neighbor Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Neighbor Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                             . . .                             :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Link Type   |   Reserved    |       Link Message Size       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Neighbor Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Neighbor Address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:                                                               :
                              (etc)
```

This is sent as the data-portion of the general packet format described in 6.1, with the "Message Type" set to HELLO_MESSAGE and the TTL field set to 0.

7.3.2.1. Description of the fields

MPR Sequence Number

This field indicates the sequence number corresponding to the most recent MPR set, calculated by the sender node.

Link Message Size

The size of the link message, measured from the beginning of the "Link Type" field and until the next "Link Type" field (or - if there are no more link types - the end of the message).

Link Type

This field specifies the type of link the sending node has to the following list of neighbors. As a minimum, the following three link types are REQUIRED by OLSR:

- ASYM_LINK - indicating that the links between the sender and the neighbors in the following list are asymmetric (i.e. the neighbor is "heard").

- SYM_LINK - indicating that the links between the sender and the neighbors in the following list are symmetric.

- MPR_LINK - indicating, that the nodes in the following list have been selected by the sender as MPR. (Notice: this implies, that the links from the sender of the HELLO and to the nodes in the list are symmetric).

It is possible to provide additional information by specifying additional link-types, e.g. LOST_LINK - indicating that the link between the sender and the neighbors in the following list has been lost. Upon processing a HELLO message, a node silently ignores link-types, which are unknown.

Reserved

This field is reserved for future usage, and MUST be set to 0000000000000000000000000 for compliance with this draft.

Neighbor Address

The address of a neighbor.

7.3.3. HELLO message processing

Upon receiving a HELLO message, the node SHOULD update the neighbor information corresponding to the sender node address (a node may - e.g. for security reasons - wish to restrict updating the neighbor-table, i.e. ignoring HELLO messages from some nodes).

In this section, the term "Originator Address" will be used for the address of the node which sent the HELLO-message.

1. If there exists a neighbor tuple with N_addr = Originator Address:

   1.1 if for that tuple N_status == ASYM_LINK:

      1.1.1 if the node finds its own address among the addresses listed in the HELLO message (with Link Type ASYM_LINK, SYM_LINK or MPR_LINK), it updates the N_status of the tuple to SYM_LINK and sets N_time = current time + NEIGHB_HOLDING_TIME.

      1.1.2 otherwise, if the node does not find its own address among the addresses listed in the HELLO message, it sets N_time = current time + NEIGHB_HOLDING_TIME.

   1.2 otherwise, if for that tuple:

      N_status == SYM_LINK OR
      N_status == MPR_LINK

      then:

      1.2.1 if the node finds its own address among the addresses listed in the HELLO message (with Link Type ASYM_LINK, SYM_LINK or MPR_LINK), it sets N_time = current time NEIGHB_HOLDING_TIME.

2. Otherwise, a new neighbor tuple is created with:

   N_addr = Originator Address

   N_status with the value of SYM_LINK if the node finds its own address (with Link Type ASYM_LINK, SYM_LINK or MPR_LINK) among the addresses listed in the HELLO message, and to the value of ASYM_LINK otherwise

   N_time = current time + NEIGHB_HOLD_TIME

The 2-hop neighbor set is updated as follows: for each 2-hop neighbor address listed in the HELLO message with Link Type SYM_LINK or MPR_LINK:

1. if a 2-hop tuple exists with:

   N_addr == Originator Address AND
   N_2hop_address == the address of the 2-hop neighbor,

   then the N_time of that tuple is set to:

   N_time = current time + NEIGHB_HOLD_TIME

2. otherwise a new 2-hop tuple is created with:

   N_addr = Originator Address,

   N_2hop_address = the address of the 2-hop neighbor,

   N_time = current time + NEIGHB_HOLD_TIME.

Based on the information obtained from the HELLO messages, each node construct its MPR selector set.

Thus, upon receiving a HELLO message, if a node finds its own address in the address list with a link type of "MPR", it MUST update the MPR selector set to contain updated information about the sender of the HELLO message:

1. If a MPR selector tuple exists with:

   MS_addr == Originator Address

   then the expiration time of that tuple is set to:

   MS_time = current time + NEIGHB_HOLD_TIME.

2. Otherwise, a new MPR selector tuple is created with:

   MS_addr = Originator Address

   MS_time = current time + NEIGHB_HOLD_TIME

   2.1 MSSN is incremented by one to indicate that the MPR selector table has been changed.

If link layer information describing connectivity to neighboring nodes is available (i.e. loss of connectivity such as through absence of an acknowledgment), this MAY be used in addition to the information from the HELLO-messages to maintain the neighbor table and the MPR selector table as described in section 7.7.

7.4. Multipoint relay selection

Each node in the network selects independently its own set of MPRs. MPRs are used to flood control messages from that node into the network while reducing the number of retransmissions that will occur in a region. Thus, the concept of MPRs is an optimization of a pure flooding mechanism.

The MPR set must be calculated by a node in a way such that it, through the neighbors in the MPR-set, can reach all 2-hop neighbors. This means that the union of the neighbor sets the MPR nodes contains the entire 2-hop neighbor set. While it is not essential that the MPR set is minimal, it is essential that all 2-hop neighbors can be reached through the selected MPR nodes. The smaller a MPR-set, however, the more optimizations are achieved.

By default, the MPR set can coincide with the entire neighbor set. This will be the case at network initialization.

The following specifies a proposed heuristic for selection of MPRs [2]. The following terminology will be used in describing this algorithm:

N: The net of neighbors with which there exists a symmetric link.

N2: The set of 2-hop neighbors. This set does not contain any one hop neighbors.

D(y): Degree of one hop neighbor node y (where y is a member of N), is defined as the number of symmetric one hop neighbors of node y, EXCLUDING the node performing the computation and all its direct neighbors.

The proposed heuristic is as follows:

1. Start with an empty MPR set
2. Calculate D(y), where y is a member of N, for all nodes in N.
3. Select as MPRs those nodes in N which provide the "only path" to some nodes in N2
4. While there exist nodes in N2 which are not covered by MPR:

   4.1 For each node in N, calculate the number of nodes in N2 which are not yet covered by MPR and are reachable through this one hop neighbor;
   4.2 Select as a MPR that node of N which reaches the maximum number of uncovered nodes in N2. In case of a tie, select that node as MPR whose D(y) is greater.

5. As an optimization, process each node y in MPR. If MPR\{y} still covers all nodes in N2, Y SHOULD be removed from the MPR set.

After selecting the MPRs among the neighbors, the link status of the corresponding one hop neighbors is changed from SYM_LINK to MPR_LINK in the neighbor table. MPR_Seq_Num value in the Neighbor table is also incremented by one.
The MPR set is re-calculated when:

- a change in the neighborhood is detected, i.e. either a symmetric link with a neighbor is failed, or a new neighbor with a symmetric link is added; or
- a change is detected in the 2-hop neighborhood such that a symmetric link is either detected or broken between a 2-hop neighbor and a neighbor.

7.5. Multipoint relay information declaration

7.5.1 Topology information base

Each node in the network maintains topological information about the network. This information is acquired from TC-messages and used for routing table calculations.

Thus, for each destination in the network, a "Topology Tuple" (T_dest, T_last, T_seq, T_time) is recorded. T_dest is the address of a node, which may be reached in one hop from the node with the address T_last. T_seq is a sequence number, and T_time specifies the time at which this tuple expires and *MUST* be removed.

In a node, the set of Topology Tuples are denoted the "Topology Set".

7.5.2. TC Message Broadcast

In order to build the topology information base needed, each node, which has been selected as MPR, broadcasts Topology Control (TC) messages. TC messages are flooded to all nodes in the network and take advantage of MPRs. MPRs enable a better scalability in the distribution of topology information [1].

A TC message is sent by a node in the network to declare its MPR Selector set. I.e., the TC message contains the list of neighbors which have selected the sender node as a MPR. The sequence number (MSSN) associated with this MPR selector set is also sent with the list. The list of addresses can be partial in each TC message (e.g. due to message size limitations, imposed by the network), but parsing of all TC messages describing a nodes MPR selector set MUST be complete within a certain refreshing period (TC_INTERVAL). The information diffused in the network by these TC messages will help each node to calculate its routing table. A node which has an empty MPR selector set, i.e. nobody has selected it as a MPR, MUST NOT generate any TC message.

A node MAY transmit additional TC-messages to increase its reactiveness to link failures. I.e. when a change to the MPR selector set is detected and this change can be attributed to a link failure, a TC-message SHOULD be transmitted after a shorter interval than TC_INTERVAL.

The proposed format of a TC message is

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|              MSSN             |           Reserved            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Multipoint Relay Selector Address               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Multipoint Relay Selector Address               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

This is sent as the data-portion of the general message format described in 6.1, with the "Message Type" set to TC_MESSAGE. The time to live SHOULD be set to 255 (maximum value) to diffuse the message into the entire network.

7.5.2.1. Description of the fields

MPR Selector Sequence Number (MSSN)

A sequence number is associated with the MPR selector set. Every time a node detects a change in its MPR selector set, it increments this sequence number. This number is sent in this MSSN field of the TC message to keep track of the most recent information. When a node receives a TC message, it can decide on the basis of this MPR Sequence Number, whether or not the received information about the MPR selectors of the originator node is more recent than what it already has.

Multipoint Relay Selector Address (MPR-S)

This field contains the address of a node, which has selected the Originator node (of the TC message) as a MPR. All addresses of the MPR selectors of the Originator node are put in the TC message. If the maximum allowed message size (as imposed by the network) is reached while there are still MPR selector addresses which which have not been inserted into the TC-message, more TC messages will be generated until the entire MPR selector set has been sent.

Reserved

This field is reserved for future usage, and MUST be set to '0000000000000000' for compliance with this draft.

7.5.3. TC Message Processing

TC messages are broadcasted and retransmitted by the MPRs in order to diffuse the messages in the entire network.

In this section, the term "originator" is used to designate the node from which the message originally originated, while the term "sender" is used to designate the node from which the message was received (i.e. the "last hop" of the message).

The tuples in the topology set are recorded with the topology information that is exchanged through TC messages, following the following algorithm:

1. If the sender (NB: not originator) of this message is not in the neighbor set of this node, the message is discarded.

2. A tuple is inserted into the duplicate table to prevent being processed again with:

   D_addr = originator address

   D_seq_num = Message Sequence

   D_time = current time + D_HOLD_TIME.

3. If there exist some tuple in the topology set where:

   T_last == originator address AND
   T_seq > MSSN,

   then no further processing of this TC message is performed and the message is silently discarded (case: message received out of order).

4. All tuples in the topology set where:

   T_last == originator address AND
   T_seq < MSSN

   are removed from the topology set.

5. For each of the MPR selector address received in the TC message:

   5.1 If there exist some tuple in the topology set where:

      T_dest == MPR selector address, AND
      T_last == originator address,

      then the holding time of that tuple is set to:

      T_time = current time + TOP_HOLD_TIME.

5.2 Otherwise, a new tuple is recorded in the topology set where:

T_dest = MPR selector address,

T_last = originator address,

T_seq = MSSN,

T_time = current time + TOP_HOLD_TIME.

7.6. Routing table calculation

Each node maintains a routing table which allows it to route the messages for the other destinations in the network. The routing table is based on the information contained in the neighbor set and the topology set. Therefore, if any of these tables is changed, the routing table is re-calculated to update the route information about each destination in the network. The route entries are recorded in the routing table in the following format:

```
1.  R_dest    R_next    R_dist
2.  R_dest    R_next    R_dist
3.    ,,        ,,        ,,
```

Each entry in the table consists of R_dest, R_next and R_dist, which specifies that the node identified by R_dest is estimated to be R_dist hops away from the local node, and that the one hop neighbor node with address R_next is the next hop node in the route to R_dest. Entries are recorded in the table for each destination in the network for which the route is known. All the destinations for which the route is broken or partially known are not entered in the table.

This routing table is updated when a change is detected in the neighbor set, or the topology set. The update of this routing information does not generate or trigger any messages to be transmitted, neither in the network, nor in the one-hop neighborhood.

To construct the routing table of node X, a shortest path algorithm is run on the directed graph containing the arcs X -> Y where Y is any one hop neighbor of X (with Link Type SYM_LINK or MPR_LINK) and the arcs U -> V where there exists an entry in the topology set with V as T_dest and U as T_last.

The following procedure is given as an example to calculate (or re-calculate) the routing table :

1. All the entries from the routing table are removed.

2. The new routing entries are added starting with the one hop neighbors (h=1) as the destination nodes. For each neighbor entry in the neighbor table, whose Link Type is SYM_LINK or MPR_LINK, a new routing entry is recorded in the routing table where R_dest and R_next are both set to the address of the neighbor and R_dist is set to 1.

3. The new route entries for the destination nodes h+1 hops away are recorded in the routing table. The following procedure is executed for each value of h, starting with h=1 and incrementing it by 1 each time. The execution will stop if no new entry is recorded in an iteration.

   3.1 For each topology entry in the topology table, if its T_dest does not correspond to R_dest of any route entry in the routing table AND its T_last corresponds to R_dest of a route entry whose R_dist is equal to h, then a new route entry is recorded in the routing table where :

      - R_dest is set to T_dest;
      - R_next is set to R_next of the route entry whose R_dest is equal to T_last; and
      - R_dist is set to h+1.

7.7 Link layer notification

OLSR is designed not to impose or expect any specific information from the link layer. However, if information from the link-layer is available, a node MAY use this as described in this section.

If link layer information describing connectivity to neighboring nodes is available (i.e. loss of connectivity such as through absence of a link layer acknowledgment), this information is used in addition to the information from the HELLO-messages to maintain the neighbor set and the MPR selector set.

Subsequently, detection of a link failure through a link-layer notification may trigger additional TC-messages to increase the protocols reactiveness to link failures. I.e. when a change to the MPR selector set is detected and this change can be attributed to a link failure, a TC-message SHOULD be transmitted.

Thus, upon receiving a link-layer notification that the link between a node and any neighbor is broken, the following actions are taken:

1. if the link is broken to either a symmetric or asymmetric neighbor, the tuple for that neighbor is removed from the neighbor set,

2. if the link is broken to a neighbor, which is selected as MPR, the tuple for that neighbor is removed from the neighbor set and the MPR set is recalculated,

3. if the link is broken to a neighbor, which has selected this node as MPR, the MPR selector set is updated and a TC message SHOULD be generated.

8. Packet forwarding

8.1. Data packet forwarding

OLSR itself does not perform packet forwarding. Rather, it maintains the routing table in the underlying operating system, which is assumed to be forwarding packets as specified in RFC1812.

8.2. Control message forwarding

Control messages, destined for flooding into the entire network, SHOULD be relayed by the MPR via the following rule:

> A node retransmits a message only when it is received from one of its MPR selector AND it is not before registered in the duplicate table AND the time to live is greater than zero.

Before retransmitting, the hop count is incremented by one and the time to live is decremented by one.

9. Proposed values for the constants

This section list the values for the constants used in the description of the protocol.

```
HELLO_INTERVAL   = 2 seconds
TC_INTERVAL      = 5 seconds

NEIGHB_HOLD_TIME = 3 x HELLO_INTERVAL
TOP_HOLD_TIME    = 3 x TC_INTERVAL
D_TIME           = 30 seconds

HELLO_MESSAGE    = 1
TC_MESSAGE       = 2

ASYM_LINK        = 1
SYM_LINK         = 2
MPR_LINK         = 3
```

10. Sequence Numbers

Sequence numbers are used in OLSR with 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-arounds (that the sequence number is incremented from the maximum possible value to zero) will occur. To prevent this from interfering with the operation of the protocol, the following MUST be observed.

The term MAXVALUE designates in the following the largest possible value for a sequence number.

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

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

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

11. References

1. P. Jacquet, P. Minet, P. Muhlethaler, N. Rivierre. Increasing reliability in cable free radio LANs: Low level forwarding in HIPERLAN. Wireless Personal Communications, 1996

2. A. Qayyum, L. Viennot, A. Laouiti. Multipoint relaying: An efficient technique for flooding in mobile wireless networks. INRIA research report RR-3898, 2000

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

4. Corson et al. Internet MANET Encapsulation Protocol. Internet draft, draft-ietf-manet-imep-spec-01.txt, Work in progress.

5. Perkins, C.E., Mobile Ad Hoc Networking Terminology, Internet draft, draft-ietf-manet-term-00.txt, work in progress.

6. Corson, S., MANET Routing Protocol Applicability Statement, Internet draft, draft-ietf-manet-appl-00.txt, Work in progress.

7. S. Bradner. Key words for use in RFCs to Indicate Requirement Levels. Request for Comments (Best Current Practice) 2119, Internet Engineering Task Force, March 1997.

8. Philippe Jacquet and Laurent Viennot, Overhead in Mobile Ad-hoc Network Protocols, INRIA research report RR-3965, 2000

12. Authors' Addresses

Amir Qayyum
Project HIPERCOM
INRIA Rocquencourt
BP 105
78153 Le Chesnay Cedex, France
Phone: +33 1 3963 5273
Email: Amir.Qayyum@inria.fr

Philippe Jacquet
Project HIPERCOM
INRIA Rocquencourt
BP 105
78153 Le Chesnay Cedex, France
Phone: +33 1 3963 5263
Email: Philippe.Jacquet@inria.fr

Anis Laouiti
Project HIPERCOM
INRIA Rocquencourt
BP 105
78153 Le Chesnay Cedex, France
Phone: +33 1 3963 508832
Email: Anis.Laouiti@inria.fr

Laurent Viennot
Project HIPERCOM
INRIA Rocquencourt
BP 105
78153 Le Chesnay Cedex, France
Phone: +33 1 3963 5225
Email: Laurent.Viennot@inria.fr

Paul Muhlethaler
Project HIPERCOM
INRIA Rocquencourt
BP 105
78153 Le Chesnay Cedex, France
Phone: +33 1 3963 5278
Email: Thomas.Clausen@inria.fr

Thomas Clausen
Project HIPERCOM
INRIA Rocquencourt
BP 105
78153 Le Chesnay Cedex, France
Phone: +33 1 3963 5133

Email: Thomas.Clausen@inria.fr