INTERNET-DRAFT
IETF MANET Working Group
Expiration: 7 August 2000

Philippe Jacquet
Paul Muhlethaler
Amir Qayyum
INRIA Rocquencourt, France

7 February 2000

Optimized Link State Routing Protocol
draft-ietf-manet-olsr-01.txt

Status of this Memo

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 the selected nodes which forward the broadcast packets during the flooding process. This technique substantially reduces the packet overhead as compared to pure flooding mechanism where every node retransmits the packet when it receives the first copy of the packet. In OLSR protocol, the information flooded in the network

"through" these multipoint relays is also "about" the multipoint relays. Hence, a second optimization is achieved here by minimizing the "contents" of the control packet 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 OLSR protocol for route calculation, and therefore the routes contain only the MPRs as the intermediate nodes from Source to Destination. It results in providing the optimal routes (in terms of number of hops), and hence another optimization. The protocol is particularly suitable for the large dense networks as the technique of multipoint relays works well in this context.

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]. 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 the mobile adhoc networks. It functions as a table driven or 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 multipoint relays to do efficient flooding of its control messages in the network. In route calculation, these are the multipoint relays which are used to form the route towards any destination in the network.

Multipoint relays are selected among the one hop neighbors with "symmetric" i.e. bi-directional link. Therefore, selecting the route through multipoint relays automatically avoids the problems associated with data packet transfer on uni-directional links; like the problem of getting the acknowledgement for the data packets at each hop, which cannot be received if there is a uni-directional link in the selected route.

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

2. OLSR terminology

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

multipoint relay (MPR)

A node which is selected by its one-hop neighbor node X to "re-transmit" all the broadcast packets that it will receive from X, provided that the same packet is not already received, and the hop count field of the packet is greater than zero.

multipoint relay selector (MPR-S)

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

node

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

symmetric link

A bi-directional *link* (not connection) between two neighbor nodes, i.e. node X and node Y, both can hear each other. This bi-directional link can be a union of two oppositely-directed uni-directional connections using different interfaces.

3. Applicability Section

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

3.1. Networking Context

The protocol is best suited to the large dense networks, as the optimization done using the multipoint relays works well in this context. More the network is dense and large, more optimization is achieved as compared to the normal link state algorithm. OLSR uses the hop-by-hop routing, i.e. each node uses its most recent information to route the packet. So when a node is moving, its speed should be such that its movement could be followed in *at least* its neighborhood, to correctly route the packets to the destination.

This protocol is best suited for the networks where the traffic is random and sporadic between "several" nodes instead of being between a small specific set of nodes of the network. The comparative performance of the protocol with a reactive protocol is still better if these [source, destination] pairs change with time. These 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.

3.2. Protocol Characteristics and Mechanisms

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

  No. It uses only bi-directional links (like in 802.11), but these links may be composed of oppositely directed uni-directional "connections".

* 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. The protocol uses hop-by-hop routing.

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

Yes. Periodically, each node in the network send a message containing the addresses of its neighbors who has selected that node as a multipoint relay. This information helps other nodes to build routes to that node through these multipoint relay selectors.

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

No. As the packets are sent periodically, they need not be sent reliably. Each packet not only contains a unique Packet Sequence Number, but it also contains the sequence number for the most recent information (for example "MPR Sequence Number" in the TC packet), so un-sequenced delivery of packets will also not create any problem.

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

Yes. Each network interface is assigned a unique IP address.

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

Yes. The concept of RID [4] may be used to associate to a single RID (which can also be a unique IP address) more than one IP addresses which represent different hosts associated to the router.

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

Yes.

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

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

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

No. All the routers in the network have their own routing tables and does not depend on any specific node.

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

No. But it decreases and increases the interval (within certain limits) of sending the TC packet periodically, depending upon the rate of link changes in its neighborhood.

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

Yes. It periodically sends the information about its multipoint relay selectors, which helps other nodes to build routes to it.

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

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

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

Yes, it can provide support for sleep period operation. To enable this feature, the protocol should select its multipoint relays from only among the nodes which can (or agree to) store its packets while it is in sleep mode.

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

No, not itself. It can use other protocols (like IMEP [4]) which provide authentication and security.

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

Yes. Each interface has a unique IP address.

4. Protocol Overview

OLSR is a proactive routing protocol for mobile adhoc networks. The protocol inherits the stability of a link state algorithm and has the advantage of having the routes immediately available when needed due to its proactive nature. OLSR protocol is an optimization of the pure link state protocol, tailored for the mobile adhoc networks. First, it reduces the size of the control packets: instead of all links, it declares only a subset of links with its neighbors who are its multipoint relay selectors (see section 5 on Multipoint Relays). Secondly, it minimizes flooding of its control traffic by using only the selected nodes, called multipoint relays, to diffuse its messages. This technique significantly reduces the number of retransmissions in a flooding or broadcast procedure.

Apart from its normal periodic control messages, the protocol does not generate extra control traffic in response to link failures and additions. Thus it is suitable for networks with a high rate of topological changes. As OLSR protocol keeps the routes for all the destinations in the network so the protocol is beneficial for the traffic patterns where a large subset of network nodes are communicating with another large subset of nodes, and the [source,destination] pairs are also changing with time. The protocol is particularly suited to large and dense networks, as the optimization done using the multipoint relays works well in this context. More dense and large a network is, more optimization is achieved as compared to the normal link state algorithm.

The protocol is designed to work in a completely distributed manner and thus does not depend upon any central entity. The protocol does not require a reliable transmission for its control messages: each node sends its control messages periodically, and can therefore sustain a loss of some packets from time to time, which arrives very often in the radio networks due to collisions or other transmission errors and problems. The protocol also does not need a sequenced delivery of its messages, as each control message contains a sequence number of the most recent information. So the re-ordering at the receiving end can not affect the functioning of the protocol. Furthermore, it provides support for the sleep mode operation, which is quite advantageous for the battery operated small terminals.

OLSR protocol does not do the source routing. Instead it performs hop by hop routing, i.e. each node uses its most recent information to route the packet. Hence, when a node is moving, its speed should be such that its movement could be followed in at least its neighborhood, to correctly route the packets to the destination.

The protocol does not require any change in the IP format of packets and classical IP stack can be used as it is, since the protocol only impacts the routing table management.

5. Multipoint relays

The idea of multipoint relays is to minimize the flooding of broadcast 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 retransmit its packets. 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, read and process the packet but do not retransmit the broadcast message received from node N.

Each node selects its multipoint relay set among its one hop neighbors in such a manner that it covers (in terms of radio range) all the nodes that are two hops away. We define the neighborhood of any node N as the set of nodes which have a symmetric link to N. We define the two hop neighborhood of N 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 multipoint relay set of N (we can call as MPR(N)) is an arbitrary subset of the neighborhood of N which satisfies the following condition: every node in the two hop neighborhood of N must have a symmetric link toward MPR(N). The smaller the multipoint relay set is, the more optimal is the routing protocol. [2] gives an analysis and example about multipoint relay selection algorithms.

Each node has a set of its neighbors which are called the "Multipoint Relay Selectors" of the node. A node obtain this information from the periodic HELLO messages of its 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, which 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 protocol relies on the selection of multipoint relays, and calculates the routes to a destination through these nodes, i.e. MPR nodes are selected as intermediate nodes. To implement this, each node in the network periodically broadcast the information describing which neighbors have selected it as a multipoint relay. Upon receipt of this "MPR Selectors" information, each node calculates or updates the route to each known destination. So principally, the route is a sequence of hops through the multipoint relays from source to the destination.

Multipoint relays are selected among the one hop neighbors with "symmetric" i.e. bi-directional link. Therefore, selecting the route through multipoint relays automatically avoids the problems associated with data packet transfer on uni-directional links such as the problem of getting an acknowledgement for the data packets at each hop which cannot be received if there is a uni-directional link in the selected route.

6. Protocol functioning

OLSR protocol carry out different functions which are required to perform the task of routing. Here we discuss these functionalities of the protocol.

6.1. Neighbor sensing

6.1.1. HELLO message broadcast

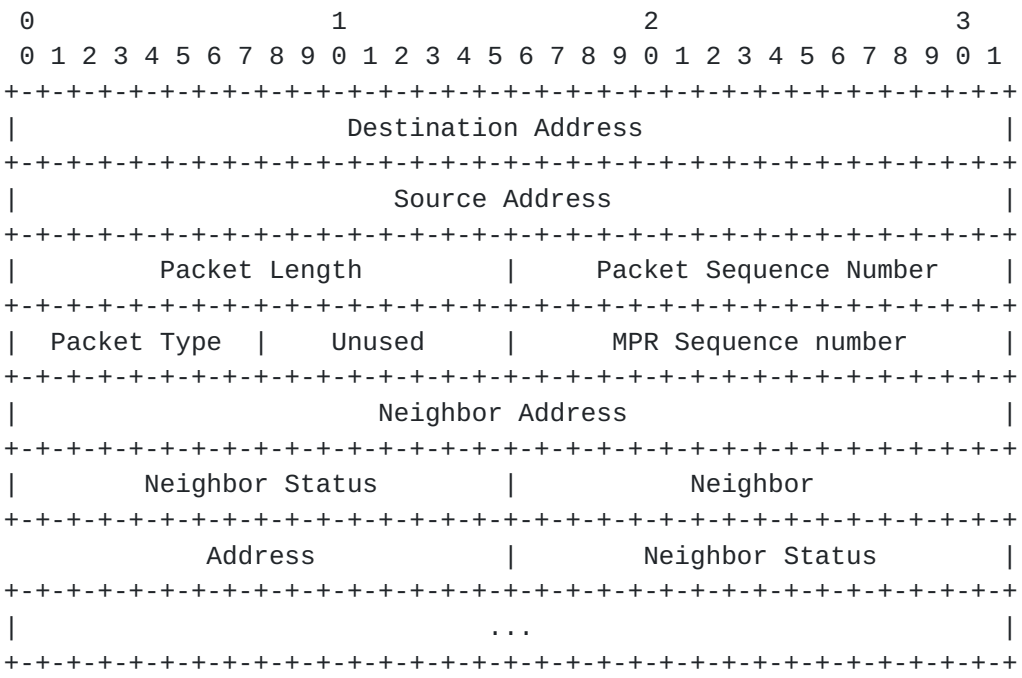
Each node must detect the neighbor nodes with which it has a direct and symmetric link. The uncertainties over radio propagation may make some links asymmetric. Consequently, all links must be checked in both directions in order to be considered valid.

To accomplish this, each node periodically broadcasts its HELLO messages, containing the information about its neighbors and their link status. These control packets are transmitted in the broadcast mode. These are received by all one-hop neighbors, but they are *not relayed* to further nodes. A HELLO message contain:

- list of addresses of the neighbors to which there exists a valid symmetric link;
- list of addresses corresponding to heard nodes, i.e., the nodes which are heard by this node (a HELLO has been received) but the link is not yet validated as symmetric

The list of neighbors in the HELLO packet can be partial, the rule being that all neighbor nodes are cited at least once within a predetermined refreshing period (HELLO_INTERVAL).

The proposed format of a HELLO packet is



6.1.1.1. Description of the fields

Destination address

For all the HELLO packets, this 4-bytes address field is always set to the broadcast address of the network, so that when this packet is transmitted, each neighbor node get this information and hence update its neighbor table.

Source address

It is set to the address of the node transmitting this HELLO packet.

Packet Length

This field contains the length of the HELLO packet in bytes, starting from the Packet Type field, i.e., length of rest of the packet.

Packet Sequence Number

While generating the HELLO packet, the node will assign a unique identification number to this packet, and will put this number in the Sequence Number field. This sequence number will be different for all the packets originated by that node.

Packet Type

The Packet Type field is set to the HELLO_PACKET value.

Unused

This unused one byte field is reserved for the future use.

MPR Sequence Number

This field indicates the sequence number with which the most recent multipoint relay set is calculated by the sender node.

[Neighbor Address, Neighbor Status]

These pairs of fields contain, one by one, all the addresses of the neighbor nodes present in the neighbor table, along with their link status.

6.1.2. HELLO message processing

The HELLO messages permit each node to learn the knowledge of its neighborhood up to two hops. A node maintains a Neighbor table in which it records the information obtained from the HELLO messages, about its one hop neighbors, the status of the link with these neighbors, and a list of two hop neighbors that these one hop neighbors give access to. The information is recorded in the Neighbor table as a neighbor entry, which may have the following format to record these entries:

		N_MPR_seq		
1.	N_addr	N_status	N_2hop_list	N_time
2.	N_addr	N_status	N_2hop_list	N_time
3.	,,	,,	,,	,,

Each entry in the table consists of N_addr, N_status, N_2hop_list, and N_time, which specifies that the node with address N_addr is a one-hop neighbor to this node, the status of the link between them is N_status, and this neighbor provides access to the two hop neighbors listed in N_2hop_list. The N_status can be ASYM_LINK, SYM_LINK or MPR_LINK. The link status as MPR_LINK implies that the link with the neighbor node N_addr is symmetric *AND* the node N_addr is selected as a multipoint relay by this node. Each neighbor entry has an associated holding time N_time, upon expiry of which it is no longer valid and hence removed.

The neighbor table also contains a sequence number N_MPR_seq which specifies that the node keeping this neighbor table has selected its most recent MPR set with the sequence number N_MPR_seq. Every time a node selects or updates its multipoint relay set, this N_MPR_seq is incremented to a higher value.

On reception of a HELLO message, the node updates the neighbor entry corresponding to the sender node address:

1. If the entry already exists:

   1.1 the holding time of the entry is refreshed to NEIGHB_HOLD_TIME
   1.2 if the node finds its own address in one of the [Neighbor, Status] pairs in the HELLO message, it updates the status of the link to the sender node as SYM_LINK if it was ASYM_LINK before.

2. Otherwise a new entry is recorded in the Neighbor table with:

   2.1 N_addr is set to the sender node address
   2.2 N_status is set to the value of ASYM_LINK (asymmetric link)
   2.3 N_2hop_list is filled with the list of addresses contained in the HELLO message in the [Neighbor, Status] pairs for which the Status is *NOT* asymmetric and which are not already present in the Neighbor table (i.e., they are not one-hop neighbors). If a node finds its own address in the [Neighbor, Status] pair, it does not register itself in the N_2hop_list, and it changes the link status to the sender node from ASYM_LINK to SYM_LINK.
   2.4 N_time is set to the value of NEIGHB_HOLD_TIME

With information obtained from the HELLO messages, each node also construct its MPR Selector table, in which it puts the addresses of its one hop neighbor nodes which has selected it as a multipoint relay. The MPR Selector table may have the following format to record the entries:

	MSSN		
1.	MS_addr	MS_seq_num	MS_time
2.	MS_addr	MS_seq_num	MS_time
3.	,,	,,	,,

Each entry in the table consists of MS_addr, MS_seq_num and MS_time, which specifies that the node with address MS_addr has selected this node as its multipoint relay with the MPR sequence number equal to MS_seq_num. Each entry has an associated holding time MS_time, upon expiry of which it is no longer valid and hence removed.

A sequence number MSSN is associated to this table which specifies that the multipoint relay selector set of the node keeping this MPR Selector table is most recently modified with the sequence number MSSN. The node modifies its MPR Selector set according to the information it receives in the HELLO messages, and increment this sequence number on each modification.

On the reception of a HELLO message, if a node finds its own address in the [Neighbor, Status] pair with the Status field equal to "MPR", then it updates the entry corresponding to the sender node's address in the MPR Selector table:

1. If the entry exists already:

   1.1 if the MPR Sequence Number field of the HELLO message is greater than or equal to the MS_seq_num field of that entry, then the MS_time is refreshed to NEIGHB_HOLD_TIME.
   1.2 if the MPR Sequence Number field of the HELLO message is greater than the MS_seq_num field of that entry, the MS_seq_num field is updated to the value of MPR Sequence Number field of the HELLO message.

2. Otherwise, a new entry is recorded in the MPR Selector table, with:

   2.1 MS_addr is set to the address of sender of the HELLO message
   2.2 MS_seq_num is set to the MPR Sequence Number field of the HELLO message
   2.3 MS_time is set to the value of NEIGHB_HOLD_TIME

6.2. Multipoint relay selection

Each node of the network selects independently its own set of multipoint relays. Multipoint relays are used to flood the control messages of that node. The MPR set is calculated in a manner to contain a subset of one hop neighbors which covers all the two hop neighbors. By this we mean that the union of the neighbor sets of all MPRs contains the entire two hop neighbor set. In order to build the list of the two hop nodes from a given node, it suffices to track the list of symmetric link nodes found in the HELLO packets received by this node (this two-hop neighbor information is recorded in the neighbor table as N_2hop_list). Multipoint relays of a given node are declared in the subsequent HELLOs transmitted by this node, so that the information reaches the multipoint relays themselves. These selected multipoint relays are indicated in the HELLO messages with the link status as "MPR". The multipoint relay set is re-calculated when:

- a change in the neighborhood is detected when either a symmetric link with a neighbor is failed, or a new neighbor with a symmetric link is added; or
- a change in the two-hop neighbor set with symmetric link is detected.

The MPR set need not be optimal, however it should be small enough to achieve the benefit of the multipoint relays. Multipoint relays is an optimization of the pure flooding mechanism; it is not essential that the multipoint relay set be minimal or optimal. But it is essential that it covers the two hop nodes. By default, the multipoint relay set can coincide with the whole neighbor set. This will be the case at network initialization. Each node will manage a dedicated sequence number in order to track the changes in its multipoint relay set. This sequence number will also appear, along with the MPR list, in the HELLO messages.

We propose here a heuristic for the selection of multipoint relays [2]. We use the following terminology in describing this algorithm:

MPR(x): Multipoint relay set of node x which is running this algorithm
N(x): One hop neighbor set of node x (containing only symmetric neighbors)
N2(x): Two hop neighbor set of node x (containing only symmetric neighbors of nodes in N(x)). The two hop neighbor set N2(x) of node x does not contain any one hop neighbor of node x

D(x,y): Degree of one hop neighbor node y (where y is a member of N(x)), is defined as the number of symmetric one hop neighbors of node y EXCLUDING the node x and all the symmetric one hop neighbors of node x which exit also in N(y), i.e.,

$$D(x,y) = N(y) - x - N(x)$$

The proposed heuristic is as follows:

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

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

5. To optimize, remove each node in MPR(x), one at a time, and check if MPR(x) still covers all nodes in N2(x)

After selecting the multipoint relays from 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.

6.3. Multipoint relay information declaration

6.3.1. TC Packet Broadcast

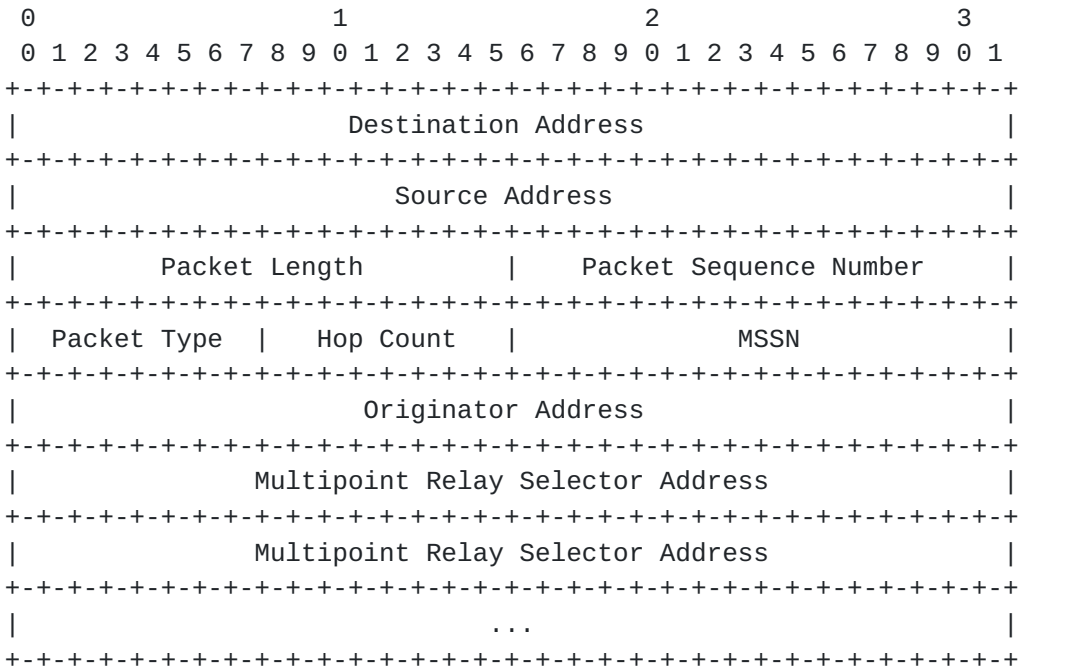
In order to build the topology information database needed for routing packets, each relay node broadcasts specific service packets called Topology Control (TC) packets. TC packets are forwarded like usual broadcast packets to all nodes in the network and take advantage of multipoint relays. Multipoint relays enable a better scalability of topology information [1].

A TC message is sent by a node in the network at regular intervals to declare its MPR Selector set, i.e., the message contains the list of neighbors who have selected the sender node as a multipoint relay. The sequence number (MSSN) associated to this multipoint

relay selector set is also sent with the list. The list of addresses can be partial in each TC packet due to maximum size limitation, but parsing must be complete within a certain refreshing period (TC_INTERVAL). The information diffused in the network by these TC packets 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 multipoint relay, does NOT generate the TC packet.

The interval between the transmission of two TC packets depends upon whether the MPR Selector set is changed or not, since the last TC packet transmitted. If no change occur in the MPR Selector set, the TC packet is sent after its normal interval (TC_INTERVAL). When a change occur in the MPR Selector set, the next TC packet is sent after a pre-specified minimum interval (TC_MIN_INTERVAL), starting from the time the last TC packet was sent. If this much time has already elapsed, the next TC packet is transmitted immediately. After sending a TC packet with that minimum interval, the default value for the interval again becomes the normal interval (TC_INTERVAL) for sending TC packets, until the MPR Selector set is changed again.

The proposed format of a TC packet is



6.3.1.1. Description of the fields

Destination address

For all the TC packets, this 4-bytes address field is always set to the broadcast address of the network, so that when this packet is diffused in the network, every node get this information and hence update its topology table.

Source address

It is set to the address of the node *transmitting* this TC packet. This field should not be confused with the Originator Address of the TC packet. Whenever a node "re-transmit" the TC packet, this field is updated to that transmitter node's address.

Packet Length

This field contains the length of the TC packet in bytes, starting from the Packet Type field, i.e., the length of rest of the packet.

Packet Sequence Number

While generating the TC packet, the "originator" node will assign a unique identification number to this packet, and will put this number in the Sequence Number field. This sequence number will be different for all the packets originated by that node, which will help to recognize the duplicate reception of the packets.

Packet Type

The Packet Type field is set to the TC_PACKET value.

Hop Count

This field will contain the maximum number of hops a TC packet can attain. Every time a TC packet is re-transmitted, this field is decremented by 1. When this field reaches zero, the TC packet is no more re-transmitted and is discarded.

MPR Selector Sequence Number (MSSN)

A sequence number is associated with the multipoint relay

selector set and every time a node detects a change in its multipoint relay selector set, it increments this sequence number. This number is sent in this MSSN field of the TC packet to keep track of the most recent information. When a node receives a TC packet, it can decide on the basis of this MPR Sequence Number, whether the information about the multipoint relay selectors of the originator node is more recent than that it already has, or not.

Originator Address

This field contains the address of the node which has originally generated this TC packet to declare its multipoint relay selector's information. This field should not be confused with the Source Address field, which is changed each time to the address of the intermediate node which is "re-transmitting" this TC packet, while the Originator Address field in never changed in the retransmissions.

Multipoint Relay Selector Address (MPR-S)

This field contains the address of the node which has selected the Originator node (of the TC packet) as a multipoint relay. All the node addresses of the multipoint relay selectors of the Originator node are put in the TC packet, one after another. If the maximum allowed packet size (of IP protocol) is attained and there are still some multipoint relay selector addresses which remain in the MPR Selector set, then more TC packets will be generated, until all addresses in the multipoint relay selector set are transmitted.

6.3.2. TC Packet Processing

In OLSR protocol, TC packets are sent in broadcast and are retransmitted by the selected nodes to diffuse the packet in the whole network. In this process, a node may receive more than once the same TC packet. To avoid the re-processing of the TC packet which was already received and processed, each node maintains a Duplicate table in which it records the information about the most recently received TC packets. The information is recorded in the Duplicate table as a Duplicate entry. The table may have the following format to record these entries:

1. D_addr D_seq_num D_time
2. D_addr D_seq_num D_time
3. ,, ,, ,,

Each entry in the table consists of D_addr, D_seq_num and D_time, which specifies that a TC packet was received from the node with address D_addr, having the packet sequence number as D-seq_num. Each Duplicate entry has an associated holding time D_time, upon expiry of which it is no longer valid and hence removed.

On reception of a TC packet, a node checks in its Duplicate table if it has already received the same packet or not. If it finds a corresponding entry, the packet is discarded. Otherwise, a new entry is recorded in the Duplicate table for this newly received TC packet, and the packet is then processed further. When a node receives a TC packet from its neighbor node with an asymmetric (or uni-directional) link, it does not register the packet in the Duplicate table neither it processes the packet.

Each node of the network maintains a topology table, in which it records the information about the topology of the network obtained from the TC packets. Based on this information, the routing table is calculated. A node records information about the multipoint relays of other nodes in the network in its topology table as a topology entry, which may have the following format:

1. T_dest T_last T_seq T_time
2. T_dest T_last T_seq T_time
3. ,, ,, ,, ,,

Each entry in the table consists of T_dest, T_last, T_seq, and T_time which specifies that the node T_dest has selected the node T_last as a multipoint relay and that the node T_last has announced this information of its multipoint relay selector set with the sequence number T_seq. Therefore, the node T_dest can be reached in the last hop through the node T_last. Each topology entry has an associated holding time T_time, upon expiry of which it is no longer valid and hence removed.

The entries in the topology table are recorded with the topology information that is exchanged among the network nodes through TC packets. Upon receipt of a TC packet, the following procedure is executed to record the information in the topology table:

1. If there exist some entry in the topology table whose T_last corresponds to the originator address of the TC packet and whose T_seq is greater in value than the MSSN in the received packet, then no further processing of this TC packet is done and it is silently discarded (case: packet received out of order).

2. If there exist some entry in the topology table whose T_last corresponds to the originator address of the TC packet and whose T_seq is lesser in value than the MSSN in the received packet, then that topology entry is removed.

3. For each of the MPR Selector address received in the TC packet:

   3.1 If there exist some entry in the topology table whose T_dest corresponds to the MPR Selector address and the T_last corresponds to the originator address of the TC packet, then the holding time T_time of that topology entry is refreshed to TOP_HOLD_TIME.

   3.2 Otherwise, a new topology entry is recorded in the topology table whereas:

   - T_dest is set to the MPR Selector address,
   - T_last is set to the originator address of the TC packet,
   - T_seq is set to the value of MSSN received in the TC packet,
   - T_time is set to the value of TOP_HOLD_TIME.

6.4. Routing table calculation

Each node maintains a routing table which allows it to route the packets for the other destinations in the network. The nodes which receive the TC packet parse and store some of the connected pairs of form [node, source] where "nodes" are identified with the addresses found in the TC packet list. The routing table is built from this database by tracking the connected pairs in a descending order. To find a path from a given origin to a remote node R, one has to find a connected pair (R,X), then a connected pair (X,Y), and so forth until one finds a node Y in the neighbor set of the origin. In order to restrict to optimal paths, the relay nodes will consider only the connected pairs on the minimal path. This selection can be done dynamically and with minimal storage facilities. The sequence numbers are used to detect connected pairs which have been invalidated by further topology changes. The information contained in the topology database, which has not been refreshed is discarded.

The routing table is based on the information contained in the neighbor table and the topology table. 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 information is updated when

- a change in the neighborhood is detected concerning a symmetric link;
- a route to any destination is expired (because the corresponding topology entry is expired) or
- a better (e.g. shorter) route is found for a destination.

Therefore, the routing table is re-calculated locally each time the neighbor table or the topology table or both are changed. The update of this routing table does not generate or trigger any packets to be transmitted, neither in the network, nor in the one-hop neighborhood.

The following procedure is executed to calculate (or re-calculate) the routing table :

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

2. The new entries are recorded in the table starting with the one hop neighbors (h=1) as the destination nodes. For each neighbor entry in the neighbor table, whose link status is not asymmetric, a new route 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. Then 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.

4. After calculating the routing table, the topology table entries which are not used in calculating the routes may be removed, if there is a need to save memory space. Otherwise, these entries may provide multiple routes.

7. Packet forwarding

7.1. Data packet forwarding

Data packets are relayed on a hop by hop basis. In the source router and in any intermediate router, the next hop router is identified by the entry of the destination in the host routing table.

Whenever a data packet is received to route to a destination and its TTL field (in IP header) is greater than zero, the node must look at the final destination field in the packet. If the route is known, i.e. an entry is found in the routing table in which R_dest corresponds to the final destination, then the packet is transmitted to the next hop node. While forwarding a unicast packet, the originator address, and the final destination address of the packets are not changed. The packet traverses the intermediate source and destination pair, hop by hop, until it reaches its final destination.

7.2. Topology Control (TC) packet forwarding

TC packets are relayed by the multipoint relays via the following rule:

A node retransmits a TC packet only when it receives its first copy from a node which is its multipoint relay selector.

When a TC packet is received and its hop count is greater than zero, then it is retransmitted by the multipoint relays of the sender node. Before retransmitting, the hop count is decremented by one.

8. Power Conservation or Sleep mode operation

Power conservation mode is very desirable for the low capacity, battery operated small terminals. With the constraint on the power consumption, nodes may wish to conserve their battery power by going into "sleep mode". The sleep mode may simply be a pause in the operation of a node, or it may be some intermittent sleep and wake periods of a node to economically use its battery resources.

8.1. Sleep mode initiation

When a node plans to go in sleep mode, it has to stop sending its periodic control messages:

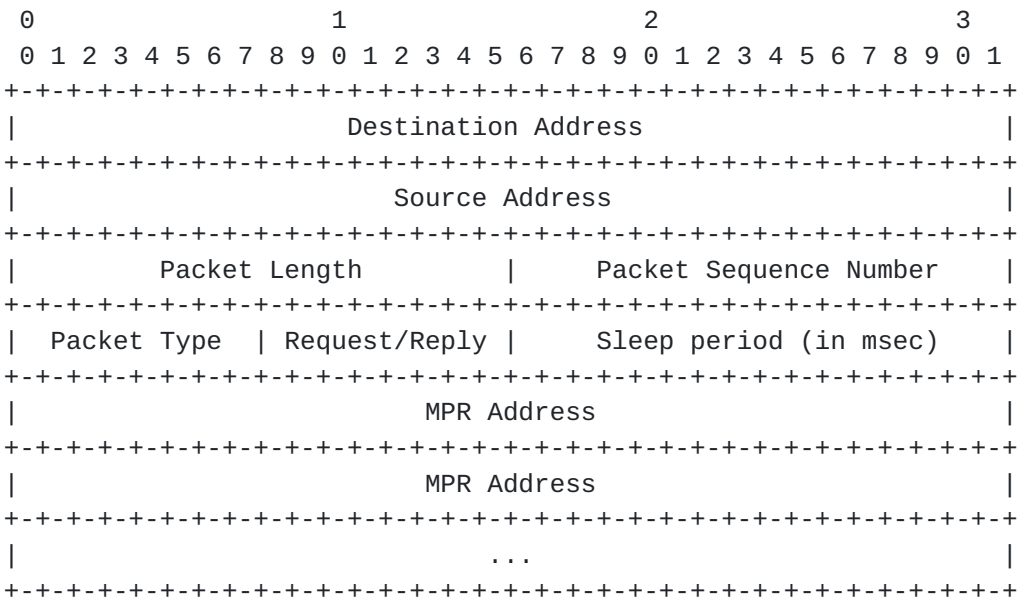
- HELLO messages: so that it is no more selected as a multipoint relay by its neighbor nodes, and
- TC messages: so that it is no more used as an intermediate node while calculating a route.

Then it looks its MPR Selector set. If this set is not empty, it means that some of its neighbors are using it as a multipoint relay, and secondly, other nodes of the network may calculate the route to some destinations using this node as an intermediate node. In this case, the node can not go into sleep mode immediately because it is assumed to function as a multipoint relay of some node. The node must wait until its MPR Selector set becomes empty. As the node is sending no more HELLOs, so it will not be selected as a multipoint relay further more, and hence the entries in the MPR Selector set will be expired.

After terminating the transmission of its periodic messages, the node has to negotiate with its multipoint relays to keep its data packets while it is in sleep mode. The node has to keep only those MPRs in its MPR set which agree to keep its packets.

To initiate the negotiation for the power conservation mode, the node has to transmit a power conservation (PC) message. This PC message is broadcast to one hop neighbors only with packet type equal to PC_PACKET. It contains the list of the MPRs of the sender node, along with the intended duration of the sleep period (in milliseconds). The Request/Reply field is set to 1 by the sender node who is requesting to keep its packets while it is in sleep mode.

The proposed format of a PC packet is



When a node receives a PC message with Request/Reply field equal to 1, then:

1. if its MPR set contains the sender node's address, this address must be removed from the MPR set and the receiver must re-calculate its MPR set;
2. if the receiver is listed as an MPR address in the PC message, it will decide if it can keep the packets for the sender node during the sleep period mentioned in the PC message:

    2.1 if it does not agree to keep sender node's packets, then no further processing of PC message is done;
    2.2 otherwise, if it agrees to keep the packets of the sender node for the intended sleep mode duration, then:

2.2.1 it will add the intended sleep period time to the holding time of the entry in the MPR Selector table corresponding to the sender node's address

2.2.2 it will reply to the sender node with a PC message in unicast, with the Request/Reply field set to 2. The sleep period field will be equal to that in the received PC message and there will be no list of MPR Addresses.

When the node who intends to go in sleep mode receives a reply of its PC message from one or more of its MPRs, it should keep only these addresses (of the MPRs) in its neighbor table and remove all others. It can now go in sleep mode.

If the node does not receive a reply after one HELLO_INTERVAL, it can re-send its PC message and wait for the reply. If still no reply is received, the node can not go in sleep mode, OR, it can go in sleep mode with a risk to loose its own packets. A "sleeping" node does not affect the routing of packets which are not destined to it. In OLSR, packets are routed hop-by-hop. So the neighbor nodes of the sleeping node will not send the packets to it, and will route the packet towards its destination according to their own most recent information.

8.2. Wake up procedure

8.2.1. Wake up to resume activities after sleep mode

The sleeping node must wake up before the sleep period (mentioned in its PC message) expires. It should start its normal operation, i.e. sending periodic HELLO and TC messages. At the same time, it should look in its neighbor table the addresses of its MPR nodes who agreed to keep its packets. Then it should request those neighbors to send these kept packets, by sending a PC message to all of them, one after another. This PC message will have the Request/Reply field equal to zero and the sleep period equal to zero. A node who receives a PC message containing Request/Reply field equal to zero the sleep period equal to zero, should send all the packets, which it has kept for the sender node.

This method of requesting its kept packets to its MPRs one by one, when a node wakes up, may avoid the high packet flow towards a node who wakes up.

If a node A is keeping packets for any node B, and the intended sleep period arrives at its expiration, node A should see if the route to node B is known, i.e. if node B is alive or not. If the route is known, all the packets are send to the node B, otherwise, node A will discard all packets of node B.

8.2.2. Wake up in the intermittent wake-and-sleep periods

The sleeping node must wake up before the sleep period (mentioned in its PC message) expires. As the node intends to re-go into sleep mode after a small wake up period, it does not resume sending HELLO and TC packets. To collect its packets from its MPR neighbors, it will send a PC message with the Destination address set to the broadcast address, the Request/Reply field set to zero and the Sleep period field set to zero. There will be no list of MPR addresses attached to the PC packet. A node who receives this PC message will send all the packets it has kept for the sender node of the PC message.

To go again into sleep mode after processing (and replying to, if necessary) the packets it receives, the node has to re-negotiate with its MPRs as mentioned in section 8.1. (Future versions of the draft may explain if sending an intermittent wake-and-sleep pattern in the first negotiation could avoid the repetitive negotiations).

To end the intermittent wake-and-sleep operation, the node should follow the procedure of section 8.2.1 when it wakes up.

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
TC_MIN_INTERVAL  = 2 seconds

NEIGHB_HOLD_TIME = 6 seconds
TOP_HOLD_TIME    = 15 seconds

HELLO_PACKET     = 1
TC_PACKET        = 2
PC_PACKET        = 3

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

10. 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, 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.

11. 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

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