MANET
Internet-Draft
Expires: August 31, 2006

K. Mase
Niigata University
C. Adjih
INRIA
Feb 27, 2006

No Overhead Autoconfiguration OLSR
draft-mase-manet-autoconf-noaolsr-01

Status of this Memo

Copyright Notice

Abstract

This document specifies one method for autoconfiguration for the Optimized Link State Routing (OLSR) protocol for stand-alone ad hoc networks. OLSR is a routing protocol for mobile ad hoc networks, designed for use in multi-hop wireless ad hoc networks ; and as such it specifies how individual nodes can construct routes to each other. To achieve this, it relies on preliminary assignment of unique IP addresses to OLSR interfaces ; hence the task of generating

addresses, checking their uniqueness and assigning them to interfaces is defined externally. This document proposes a complementary method, called "No Overhead Autoconfiguration for OLSR" (NOA-OLSR), to perform this task of ensuring uniqueness of addresses which have been generated. It also performs to ensure uniqueness of addresses which have been assigned and used when network merger occurs. This method consists of modifications in the OLSR specification.

Table of Contents

1. Introduction

A mobile ad hoc network is a collection of nodes, which collaborate to each other without depending on centralized control for enabling wireless communication among nodes. When two nodes are within direct transmission range, they communicate directly (one hop wireless communication) ; and otherwise they communicate using other nodes as intermediary nodes (multihop wireless communication), where the intermediary nodes act as routers for forwarding IP datagrams. Accordingly, routing is a key problem for mobile ad hoc networks and many routing protocols have been proposed. In IETF, in the MANET working group, two proactive routing protocols, OLSR [3] and TBRPF [4], and two reactive routing protocols, AODV [5] and DSR [6] are or will progress to experimental RFC status. The former and the latter are succeeded to OLSRv2 [7] and DYMO [8], respectively for the standard truck RFCs. However these routing protocols assume that each node has been assigned an unique IP address on each of its network interfaces. In the traditional Internet,DHCP is typically used for mobile stations to obtain their IP address. However, in stand-alone mobile ad hoc networks, such a centralized mechanism may not be available and each node needs to obtain its address in a decentralized fashion. If each node generates addresses at random from a given address space for its interfaces, uniqueness of these addresses is not guaranteed. That is, two different nodes generate and assign the same address (Dupulicate address) to their interface respectively. This is called address conflict. The chances of occurrence of address conflict is significant in IPv4 networks, where the number of bits used for host address is limited. On the other hand, this issue may be negligible in IPv6 networks, since longer bits(ex. 64 bits) can be used for host address space. However, address space should be set as small as possible even in IPv6 networks in order to maximize the effect of address compression, that is specified in OLSRv2[7]. IP address autoconfiguration is therefore an important pratical issue, regardless of IPv4 or IPv6 networks.

Many conventional methods are organized independently from routing protocols so that they can be used for any MANET regardless of the routing protocols. Unfortunately, all of these proposed methods are rather expensive as they require significant control message overhead for either avoiding or resolving address conflicts.

We propose a novel MANET-local address autoconfiguration method for MANET with OLSR, called "No Overhead Autoconfiguration for OLSR"(NOA-OLSR). Our method is an duplicate address detection without overhead based on properties of proactive link state routing protocols. NOA-OLSR is in accordauce with "a common framework for autoconfiguration of stand-alone ad hoc networks" [10].

2. Autoconfiguration Method Overview

In this section, an overview of the autoconfiguration method is given, followed by a description of the structure of the document.

The autoconfiguration algorithm detailed in this document applies to the OLSR protocol, and changes its operation. The node is assumed to implement the OLSR protocol ([3], thereafter denoted "standard OLSR"), complemented by the modifications specified here (thenceforth, "NOA-OLSR").

Under these assumptions, an OLSR node running NOA-OLSR will proceed as follows. An address is initially qenerated for its OLSR interface (manually, or using the autoconfiguration methods suggested in this document). Then, the node runs the OLSR protocol using this address, while at the same time constantly checking that it is not conflicting with the address of another node in the network (using the detection algorithm of this document). Finally, it doesn't run fully OLSR protocol initially, because it might be entering in a network where its address could be already used by another node, and it would possibly break routes of nodes which are already running. Instead, the node goes through several states, in the last of which, only, the node will ultimately run the full OLSR protocol. Similarily, in order to avoid routing table contamination, the other nodes avoid relying on this node initially, and will rely on it for routing and forwarding messages, when it has reached proper states.

To sum up, the autoconfiguration of an OLSR node includes in three parts:

- Address qeneration
- Ongoing duplicate address detection
- Gradual entry in the OLSR network and routing table contamination avoidance

Considering the address genenation, it is actually a peripherical issue of the protocol described in this document, because it is fairly independent of it. Hence an overview of address generation is provided, along with guidelines, and pointers to relevant references.

The ongoing duplicate address detection is the main addition to the OLSR protocol, detailed in Section 4.3 is , checking for inconsistencies in the routing protocol messages to diagnose duplicate address detection, using variants of the ideas pioneered by [2]:

- The first kind of inconsistency is based on information included in OLSR messages (such as HELLO messages and TC messages): many cases of duplicate address in one MANET network result into inconsistent information being received ; topology information, for instance.

- The second kind of inconsistency is based on sequence numbers: when two nodes, which selected the same IP address, are present in a network, they would send control messages that will be inconsistent.

Finally the protocol runs based on a state for each OLSR node, the "autoconfiguration state proposed in [10]". It allows one OLSR node with a newly generated address to enter gradually in running OLSR network, by sending messages which will be used by more and more nodes. At the same time, it also prevents routing table contamination by ensuring that routes go through nodes which have been present in the network long enough for the duplicate address detection to have been performed. Specifically there are three autoconfiguration states, NO_ADDRESS_STATE, ADVERTISING_STATE and NORMAL_STATE. ADVERTISING_STATE may be further divided into LOCAL_AD_STATE and GLOBAL_AD_STATE. General definitions of autoconfiguration states are given in [10]. Definition of autoconfiguration states, specific to OLSR networks, are generated straightforwardly from these general definitions, as given in Section 4.5. Note that LOCAL_AD_STATE and GLOBAL_AD_STATE are renamed to HELLO_STATE and TOPOLOGY_STATE, respectively, reflecting types of control message used in OLSR.

The remaining of this document is organized as follows:

- Section 3 collects specific terminology used

- Section 4 provides the high-level, algorithmic, part of this document. It includes:

  * Address generation.

  * Ongoing duplicate address detection.

  * Principles behind checking sequence number consistency of messages.

  * Gradual entry in the OLSR network and routing table contamination avoidance.

3. Terminology

This section provides definition for terms that have a specific meaning to the protocol specified in this document and that are used throughout the text.

MANET-local Address: a unique-local address having scope that is limited to the MANET.

Tentative Address: an address whose uniqueness in a MANET is being verified, prior to its assignment to an interface. A tentative address is not considered assigned to an interface in the usual sense. An interface discards received packets addressed to a tentative address, but accepts routing control packets related to MANET-wide Duplicate Address Detection for the tentative address.

Address Generation: Adress generation is a procedure for a node, that has currently no address, to obtain a tentative address in the MANET -either of its own accord or with support of other nodes.

Address Conflict: When two nodes in the same MANET network share the same address, the situation is described as an "Address Conflict". The nodes involved are "conflicting nodes" and their shared address is called "conflicting address". Conflicting nodes may each send one message with the same sequence number and same message type: such messages are denoted "conflicting messages".

Autoconfiguration State for OLSR: The current autoconfiguration state of the node, one of NO_ADDRESS, HELLO, TOPOLOGY, and NORMAL, which indicates what messages it should (or should not) generate and processing it should (or should not) do (see Section 4.5).

Busy Address: An address which is being used by some node in the network (see Section 4.2).

MANET_wide Duplicate Address Detection (MANET-DAD): MANET-DAD is the action of detecting address conflict, the situation where some nodes are using the same address in the same MANET network.

MANET-DAD Rule: A MANET-DAD rule is one rule of this document, which used to detect the existence of address conflict (see Section 4.3).

Familiar Address (Node): An address is familiar for a node, if the node has seen it in an OLSR message, for a sufficiently long period of time (see 4.6 and 5.4.7). A node is familiar for another node if it has a familiar address for this other node. An address or a node which is not familiar is said "unfamiliar".

NOA-OLSR: "NOA-OLSR" is the protocol specified by this document. It is the standard OLSR protocol [3] with the additions and changes specified in this document.

Routing Table Contamination Avoidance: Routing table contamination avoidance is the idea of preserving the routing table from incorrect information due to address conflict. This is achieved by using the autoconfiguration state (see Section 4.5).

Sequence Number Consistency: All OLSR messages have a sequence number. One trademark of duplicate addresses, is sequence numbers of different messages, which could not result from a correct implementation of the OLSR protocol (such as decrease in sequence numbers, etc.). The properties of sequence numbers which would result from the normal OLSR protocol implementation are termed "Sequence number consistency" (see Section 4.4).

Standard OLSR: The terms "standard OLSR protocol" refer to the OLSR protocol specified in [3]. The term "standard" is meant to differentiate with the "non-standard" OLSR protocol proposed in this document (thereafter, "NOA-OLSR"). It is not meant to express its normative status within IETF or standardization organizations.

TC Generator: A node which generates TC messages (as originator).

4. Autoconfiguration Algorithms

4.1. Overview

This section provides a high-level view of the method used for MANET-local address autoconfiguration of the node: address generation, duplicate address detection based on rules, principles for sequence number consistency, use of the autoconfiguration state.

4.2. Address Generation

When a node is in NO_ADDRESS_STATE, it can monitor the protocol message exchanges and collect information regarding the addresses in use, the "busy address list". It can then selects its own tentative address from the pool of free addresses by avoiding the busy address list. With OLSR, it is possible for each node to obtain busy address information through routing control messages received from other nodes (such information is available as part of the State Set introduced in Section 4.5).

This document doesn't specify how the addresses should be selected, apart from the fact any selected address should not be the "busy address list".

Some discussions and references about address generation (including IPv4 and IPv6 stateless address autoconfiguration) can be found, for instance, in the document [9].

4.3. MANET-wide Duplicate Address Detection(MANET-DAD)

4.3.1. Overview

MANET-DAD is performed passively, i.e., without additional control messages. Some various passive DAD techniques were proposed in [2], we propose some others. MANET-DAD is performed in either HELLO_STATE, TOPOLOGY_STATE and NORMAL_STATE. MANET-DAD in HELLO_STATE or TOPOLOGY_STATE is called "pre-service MANET-DAD" and that in NORMAL_STATE is called "in-service MANET-DAD"[10]

In this section, the detection algorithms are detailed. Protocol specifications are given in a later section.

In a MANET network with nodes running the OLSR, several different scenarios of address conflicts may occur. There are classified in three separated cases:

Neighbor Address Conflict: in this case, two neighbor nodes (in range of each other) have selected the same address.

Two-hop Address Conflict: in this case, two nodes which have selected the same address are two-hop neighbors. That is, there is another node in the network which is the neighbor of those both nodes.

Multihop Address Conflict: in this case, the two nodes in address conflict are separated by two nodes or more.

The three cases of address conflict are different in that they can be detected by different methods: for instance the multihop address conflict can be detected by the use of TC message information, while the first two cases need not.

Also, an additional case is added: it's a specific multihop address conflict case, where the address conflict results in deficiencies in the MPR selection.

4.3.2. Notation

In the Section 4.3, the following conventions are used to describe the duplicate address conflict cases for the algorithms:

- Capital letters are used to denote different nodes: such as "A", "B", "C", etc...
- Numbers are used to represent different addresses, such as "1", "2", "3", etc...
- The following notation is employed to represent the node "A" which has the address "1": "A{1}". In the event of an address conflict, two nodes may be using the same address, such as "A{1}" and "B{1}" for instance.
- Each MANET-DAD rule is associated to a figure which graphically represents the topology. An example is given on Figure 1: one node "A" with address "1". In the figures which will follow, the nodes which should apply the MANET-DAD rule, are highlighted by the mark "**", like "A" is, on the sample figure.

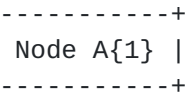

Figure 1

4.3.3. MANET-DAD Rules for Neighbor Address Conflict

In the case of "neighbor address conflict", two conflicting nodes are neighbors (see Figure 2). This case is special since many different non-OLSR methods could be used to detect the conflict: because the neighbor nodes would receive messages from each other directly, as they would, for instance, if they were connected on a Ethernet network. Thus, most of methods designed for (non-MANET) IP networks, such as IPv4 autoconfiguration detection methods or IPv6 ones, could be used.

Still, due to topology changes such methods could fail, or could not be available in a node. Hence a rule to detect conflicts at the OLSR protocol level in this case is proposed. At mininum, the two OLSR nodes should at least periodically generate HELLO messages, hence the following rule is used:

4.3.3.1. Rule R1

Rule: R1 (see Figure 2)

Context: A node A is either in HELLO_STATE, TOPOLOGY_STATE or NORMAL_STATE. An HELLO message is received by a node A{1}.

Check: Is the address {1}, the address of the originator node ?

Action: If it is the case, this node(node A) is in conflict and must enter NO_ADDRESS_STATE.

Rationale: A node doesn't receive its own HELLO messages (they are not forwarded), hence the occurence of such an event means that a node with the same address has sent an HELLO.

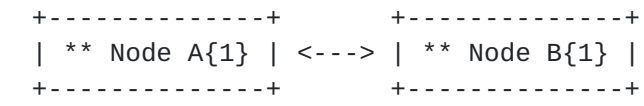

Figure 2

As mentioned, this rule can be completed by other duplicate address detection mechanisms, not specified in this document, as they are beyond its scope.

4.3.4. MANET-DAD Rules for Two-hop Address Conflict

In this case, the two conflicting nodes are two-hop neighbors, that is: they are not neighbor, but they have a common neighbor (see

Figure 3). The rule proposed here relies on the fact that a common neighbor exists, and will receive the HELLO from both nodes. The detection proceeds in three steps: the common neighbor detects the conflict using those HELLOs, then it advertises the conflict in some message(s) (rule R2), and finally, the conflicting nodes change their address upon receiving this conflict advertisement (rule R3).

4.3.4.1. Rule R2

Rule: R2 (see Figure 3)

Context: A node B is either HELLO_STATE, TOPOLOGY_STATE or NORMAL_STATE. In node B{2}: an HELLO message from address {1} was received previously, and another HELLO from address {1} is just received by B{2}.

Check: Are the sequence numbers of the HELLOs inconsistent (as defined in Section 4.4)?

Action: If it is the case, there are two or more neighbors using the same address {1}. B{2} will advertise that the address {1} is conflicting in its HELLO messages.

Rationale: If two neighbors of one node have conflicting addresses, the HELLO sequence numbers will be inconsistent.

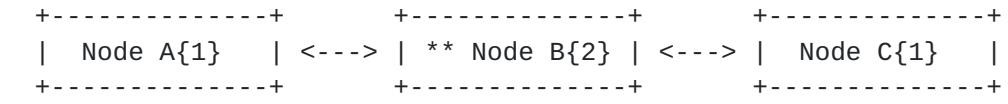

Figure 3

4.3.4.2. Rule R3

Rule: R3 (see Figure 4)

Context: A node A is either HELLO_STATE, TOPOLOGY_STATE or NORMAL_STATE. In node A{1} (and node C{1}): a neighbor B{2} is advertising that conflict exists with the address {1}.

Check: -

Action: If it is the case, A{1} is a conflicting node and must enter NO_ADDRESS_STATE.

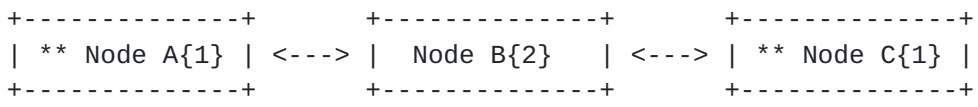

Figure 4

4.3.5. MANET-DAD Rules for Multihop Address Conflict

In this section, DAD rules are proposed to handle the case where the distance between conflicting nodes is three hops or more. In this case, in general, it cannot be assumed that a single node has enough information to detect the conflict using exclusively the HELLO messages. Hence, the logical choice is here to use information inside TC messages. However MANET-DAD is complicated by the optimizations of the OLSR routing protocol: first, not all nodes originate TC messages ; second, TC messages might include only a subset of neighbors ; third, OLSR messages may be split and as a consequence, an individual TC message from one node might not include all the topology information that the node should periodically refresh. Finally, the MPR selection algorithm can be affected by duplicate addresses, and prevent proper operation of the MPR flooding mechanism, hence prevent proper propagation of the TCs used by MANET-DAD.

The MANET-DAD rules that are specified in the case of multihop address conflict are classified depending on the status of the conflicting nodes with respect to TC generation: a node which generates TC messages (when it is a multipoint relay of some node) is called a TC generator. Three cases are possible and are handled:

- Both conflicting nodes are TC generators.
- One of the conflicting nodes is a TC generator, and the other is not.
- None of the conflicting nodes is TC generator.

In each of the three cases, the MANET-DAD rules allow detection both on the conflicting nodes (which would then change address) and on intermediary nodes (which would then avoid routing table contamination). Finally some MANET-DAD rules are used for preventing the following case:

- Conflicting nodes are impeding MPR selection.

The following four sections handle individually each case.

4.3.5.1. MANET-DAD Rules for Multihop Address Conflict with two TC generators

In this case, the two nodes in conflict are both TC generators. Then each of them would ultimately receive one TC with its own originator address, but which it did not generate (for it was generated by the other node). The intermediate nodes would also detect conflict by noticing discrepancy in the sequence numbers or discrepancy in the content of the TC messages with same sequence number.

The first rule applies to conflicting nodes (R4 (Section 4.3.5.1.1)), the second applies to other nodes in the network (R5 (Section 4.3.5.1.2)).

4.3.5.1.1. Rule R4

Rule: R4 (see Figure 5)

Context: A node A is in NORMAL_STATE. In node A{1} (or node C{1}): a TC with originator address {1} has been received. A{1} keeps track of the TC messages that it has sent.

Check: Verify whether A has actually sent that TC: the message sequence number should be the same as one message that A has sent in the past, and then the content should be the same.

Action: If it is not the case, A{1} is a conflicting node and must enter NO_ADDRESS state.

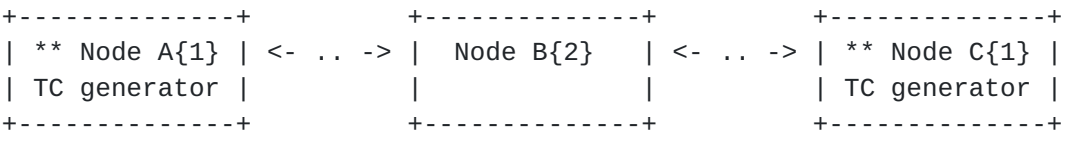

Figure 5

4.3.5.1.2. Rule R5

Rule: R5 (see Figure 6)

Context: A node B is either in TOPOLOGY_STATE or NORMAL_STATE. In node B{2}: an TC message with originator address {1} was received previously by the node, and another TC with originator address {1} is just received by B{2}

Check: Are the sequence numbers of the TC messages consistent (as defined in Section 4.4)? Is the content of the TC identical to the one(s) received before?

Action: If it is not the case, there are two or more nodes using the same address {1}: then the TC should be forwarded if the receiving node is a MPR(if it has not already been), but the content of the TC will be ignored and not processed

Rationale: This detects a conflict between TC generators. If the conflicting nodes are sending TC messages with same sequence number, standard MPR flooding might not allow the TC messages to reach the other node. Hence in case of conflict, the TC should be forwarded by default, if the receiving node is a MPR. Also, because a conflict has been detected, the received TC is guaranted to hold information which is inconsistent with the information already processed because it was issued by a different node ; and hence, the content of TC message should be ignored.

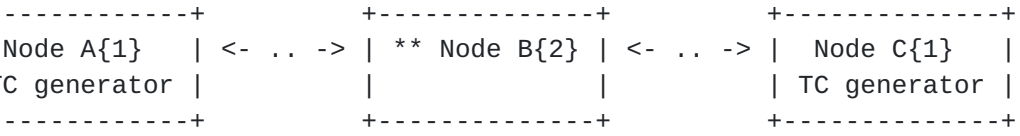

Figure 6

4.3.5.2. MANET-DAD Rules for Multihop Address Conflict with two non-generators

In this section, MANET-DAD rules are given for the case where none of the conflicting nodes is a TC generator. In such a configuration, the conflict is detected by means of by using the TC messages of the multi-point relays of the nodes. As one conflicting node selects some MPR, these MPR will send TC messages indicating this selection: when one of the TC messages reaches the other conflicting node, this node will detect inconsistency by discovering that it did not, actually, select the TC originator as MPR.

The MANET-DAD for intermediate nodes is, however more complex, because they cannot rely on sequence numbers as in Section 4.3.5.1, nor they can rely on knowledge of the actual MPR selection of every node like the nodes in conflict. Hence to detect occurences of such conflicts, another mechanism is used: it is based on the concept of familiar nodes. A node (an IP address) is familiar for another node, when the last one has had knowledge of existence of the first one for sufficiently long (see Section 4.6).

The hypothesis made now is that most conflicts occur because of network mergers. In such an address conflict, now, let's assume a node from one network is now sending TC messages including the address of one node (in conflict with this network) from another, newly merged, network. For instance, let us consider Figure 7, and let us assume that A{1}, C{2}, and E{4} were previously part of one network, while B{1} and D{3} (one of its MPRs) were part of another. It is reasonable to assume that D{3} will become the neighbor of few nodes of the first network, which it will advertise. Hence, most likely, the TC messages of D{3}, which advertise the conflicting node B{1}, also include mostly addresses of nodes from the merged network, which would be unfamiliar nodes for A{1}. Hence the MANET-DAD rule: ignore the information relative to familiar nodes, when it is inside TC messages from unfamiliar nodes, which also include too many unfamiliar nodes.

Another rule is added for neighbors of the node A{1}, such as C{2}: because they have knowledge of the neighborhood of A{1}, they are able to directly check if D{3} is a neighbor of A{1}.

4.3.5.2.1. Rule R6

Rule: R6 (see Figure 7)

Context: A node A is either in TOPOLOGY_STATE or NORMAL_STATE. In node A{1}: a TC message with originator address {3} has been received.

Check: If this TC includes the address {1} of A, A checks whether it had recently selected {3} as MPR.

Action: If it is not the case, A{1} is a conflicting node and must enter NO-ADDRESS_STATE.

Rationale: If A{1} has not selected {3} as MPR, then another node with address {1} must have done so, hence there is an address conflict.

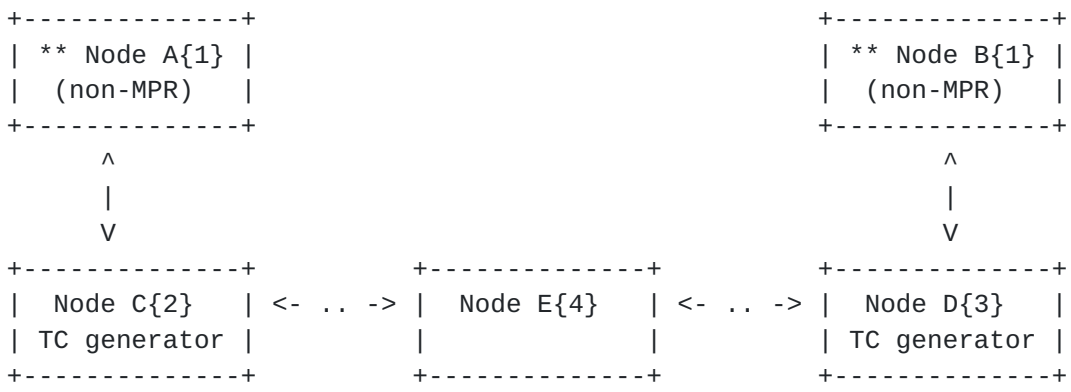

Figure 7

4.3.5.2.2. Rule R7

Rule: R7 (see Figure 8)

Context: A node E is either in TOPOLOGY_STATE or NORMAL_STATE. In node E{4}: a TC message from originator {2}, which is familiar for E, had been received. It included the familiar (for E) address {1}. Another TC, from originator {2}, an unfamiliar node for E, is including the same familiar address {1}.

Check: In this TC, check how many addresses are from familiar nodes. If too little addresses are familiar, then the TC is assumed to include an address {1} which is conflicting.

Action: If conflict is assumed, then the information of the TC of {3} about address {1} is ignored (the previous one from {2} will still be used), but all other content is kept.

Rationale: This is an heuristic for detecting conflict. Note that in any case, a route to {1} can still be computed using the TC message from {2}. Note also that after some time, {3} and all the nodes advertised by {3} will be familiar to E, ensuring that this rule will no longer apply.

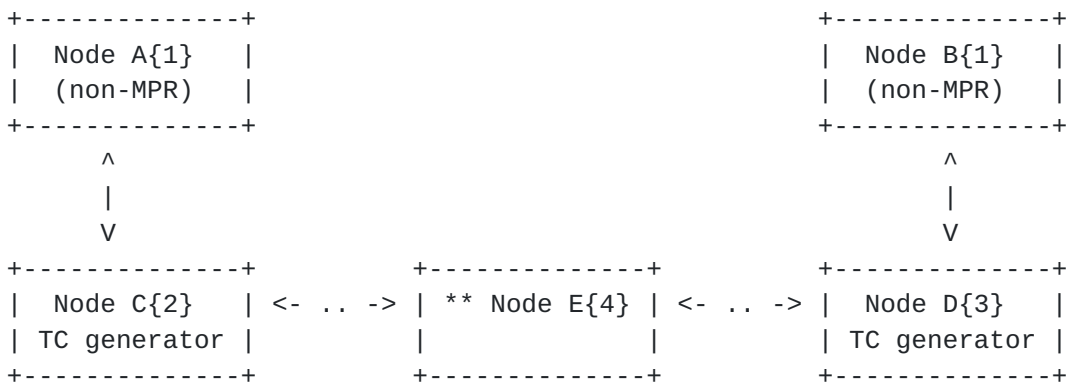

Figure 8

4.3.5.2.3. Rule R8

Rule: R8 (see Figure 9)

Context: A node C is in NORMAL_STATE. In node C{2}: a HELLO message from node {1} was previously received, and a TC message from node {3} is now received.

Check: If the TC message from {3} includes {1} as MPR selector, the HELLO from {1} should also have included {3} as symmetrical neighbor (actually more: as MPR)

Action: If it is not the case, then a conflict is assumed for address {1}. Then the information of the TC message of {3} about address {1} is ignored (the previous one from {2} will still be used), but all other content is kept.

Rationale: This is another heuristic for detecting conflict for every node which is neighbor of the conflicting nodes.

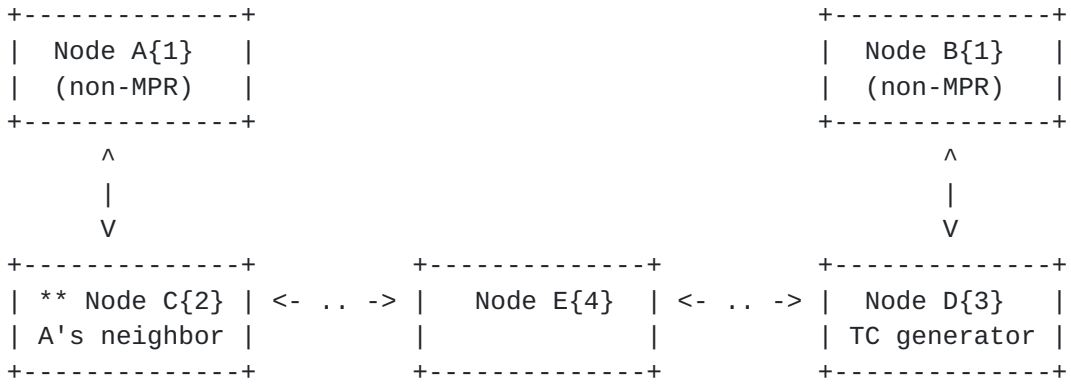

Figure 9

4.3.5.3. MANET-DAD Rules for Multihop Address Conflict with one TC Generator and one Non-Generator

In case one of the nodes in conflict is a TC generator while the other one is not, the conflict can be detected by as previously. The TC generator can conduct MANET-DAD by checking the TC messages of the MPR of the other node using Rule R6 (Section 4.3.5.2.1)(see Figure 10). The conflicting node that does not generate TC messages, can detect conflict with Rule R4 (Section 4.3.5.1.1)(see Figure 11).

However for intermediary nodes, a new case is possible. We still assume most conflicts occur because of network mergers. Then it is possible that one conflicting node is a TC generator in one network, while the other one is not in the other network. Using the same logic as previously, the TC message of that conflicting node would include many unfamiliar nodes, hence one MANET-DAD rule is to reject such TC.

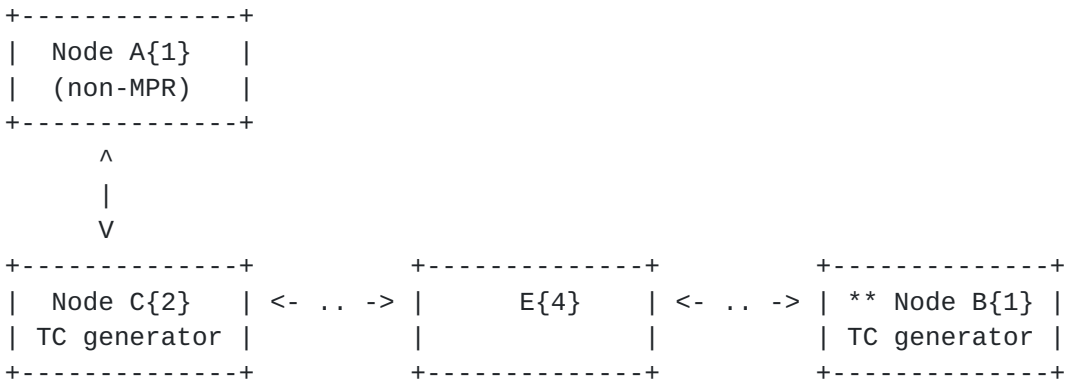

Figure 10

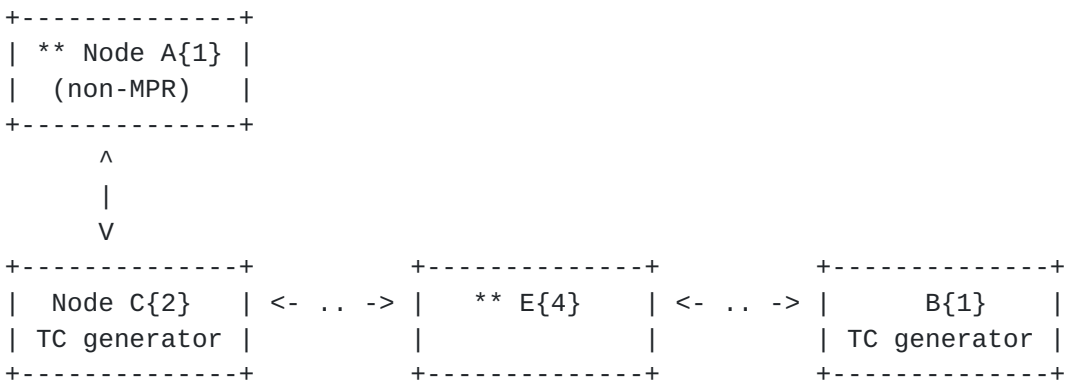

Figure 11

4.3.5.3.1. Rule R9

Rule: R9 (see Figure 12)

Context: A node E is either in TOPOLOGY_STATE or NORMAL_STATE. In node E{4}: a TC message from originator {2}, which is familiar for E, had been received and it included the (familiar for E) address {1}. Another TC message, from originator {1}, is received.

Check: In this TC, check how many addresses are from familiar nodes. If too little addresses are familiar, then the TC is assumed to be from an unfamiliar node from a merged network.

Action: If conflict is assumed, then the information of the TC is ignored (the previous one from {2} will still be used).

Rationale: This is an heuristic for detecting conflict. Note that in any case, a route to {1} can still be computed using {2} and note that in absence of conflict, anyway, after some time, all the nodes advertised by {1} will be familiar to E, ensuring that this rule will no longer apply.

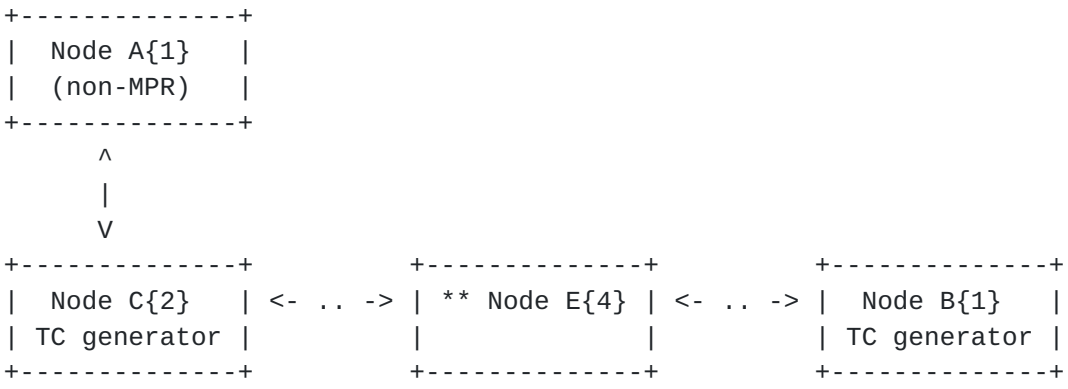

Figure 12

Additionally, still in the case of network merger, the nodes that are on the border of the network merger can actually use some heuristics for detecting conflicts. Indeed, if a node, is from another (merging) network, it is likely to have many unfamiliar nodes as neighbors. And those unfamiliar nodes will be present in the Hello messages of the node. Hence when a node detects that one of its neighbors has too many other neighbors that are unfamiliar, it can suspect the neighbor is from another network. In case the node is a TC generator, it will then mark the address of the node as unfamiliar.

4.3.5.3.2. Rule R10

Rule: R10 (see Figure 13)

Context: A node C is in NORMAL_STATE. In node C{2}: a TC message is being generated, and it includes neighbor {1}.

Check: In the neighborhood of A{1} (which is obtained from the Hello messages, in the two-hop tuple set) check how many addresses are from familiar nodes. If too little addresses are familiar, then the neighbor is assumed to be an node from a merged network.

Action: If too little address are familiar, the address {1} is advertised as being "with too many unfamiliar neighbors".

Rationale: This is an heuristic to avoid routing table contamination. Note that the address {1} is still advertised and can be used by node E{4} and B{1} to detect the conflict.

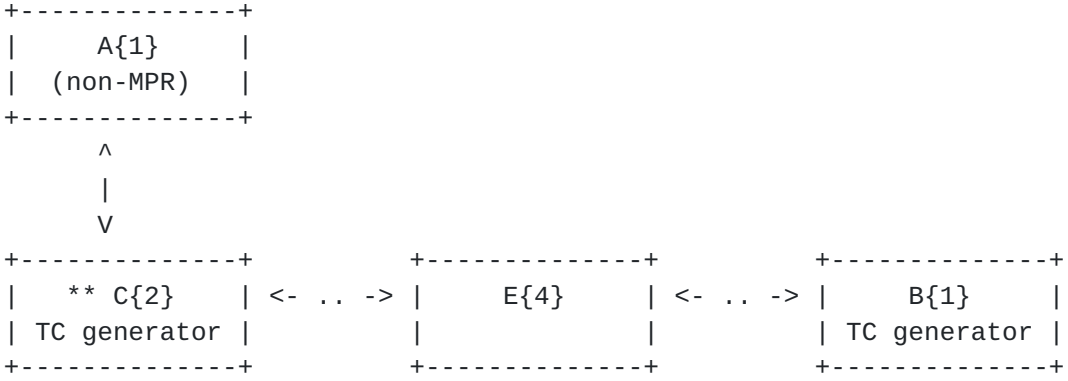

Figure 13

The following rule uses the information transmitted by the previous one:

4.3.5.3.3. Rule R11

Rule: R11 (see Figure 14)

Context: A node E is either in TOPOLOGY_STATE or NORMAL_STATE. In node E{2}: a TC message has been received from originator {2} and it includes neighbor {1} marked as ``with too many unfamiliar neighbors'', by rule R10 in node {2}.

Check: -

Action: The address {1} should be ignored in the processing of the TC message. But the other addresses may still be used, and the TC should still be forwarded with standard MPR flooding.

Rationale: This is an heuristic to avoid routing table contamination, using information from rule R10.

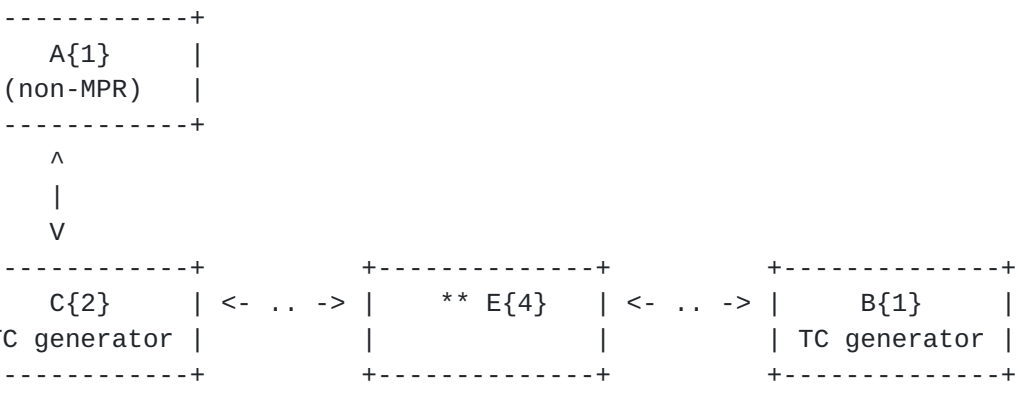

Figure 14

4.3.5.4. Three-hop DAD, Specific Case

It has been noted that in some cases the MPR selection process can fail because of duplicate addresses (see [2]). As a result, the MPR flooding mechanism may fail to deliver a message to the entire network, and then the previous MANET-DAD rules may fail to detect address conflict. This situation is illustrated in Figure 15. A specific rule can be devised to prevent this situation and allow proper MPR selection: in the figure, the node B{2} is able to detect that there is an inconsistency in the neighborhood advertised by {1} and {3}, which may possibly arise from {1} being a duplicate address. In this case, the MPR selection of B would be deficient: so B can still preventively select {3} as MPR by itself. That way, the messages from A{1} going through B will reach D{1} (triggering one of the previous MANET-DAD rules R6 and R8).

4.3.5.4.1. Rule R12

Rule: R12 (see Figure 15)

Context: A node B is in TOPOLOGY_STATE or NORMAL_STATE. In node B{2}: a HELLO from node {1} had been received, and now an HELLO from node {3} is received.

Check: If the HELLO from {3} includes {1} as symmetrical neighbor, the HELLO from {1} should also have included {3} as symmetrical neighbor.

Action: If it is not the case, there is an inconsistency and the node B should select {3} as MPR.

Rationale: Such inconsistencies should never happen in a static network, unless there is a conflict. Note also that due to topology changes, they may do so even if there is no conflict. In that case, note that the only penalty is an temporary increase of the number of MPR selected. It is still an excellent heuristic that will solve the MPR selection problem when the network is static.

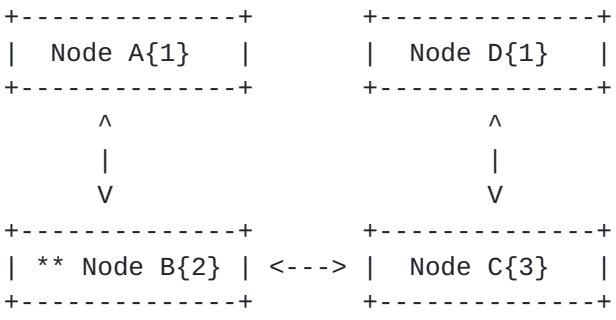

Figure 15

4.4. Sequence Number Consistency

In [2], the use of sequence numbers to verify consistency has been used in some general cases. Here, sequence number consistency is checked for the OLSR protocol, and consist really of two cases: HELLO sequence number consistency, and TC sequence number consistency.

4.4.1. Minimum Wrap-Around Limit

In the OLSR protocol [3], it is implicitly assumed that the sequence number of one node will wrap-around within an interval of time greater than DUP_HOLD_TIME. Hence this value is a good reference for the minimum expected interval before a wrap-round the sequence number of any node in the network, denoted MIN_WRAP_AROUND_INTERVAL.

4.4.2. HELLO Sequence Number Consistency

In case of HELLO messages, it is assumed that they would be received in the same order as they are transmitted (because they are not forwarded). In this case, a node observing the HELLO messages from a

neighbor will see that their sequence numbers are permanently increasing. Now if there are two neighbors B and C of one node A, the node A will receive alternatively messages from B and C, because each is transmitting indefinitly. Hence A must receive a sequence of packets from B, then some packets from C, then some packets from B, and so on. Let's assume that ultimately a sufficiently long sequence is received without packet loss, and which then will be in this order:

- one packet B1 from B (possibly the last one of a sequence of packets from B)
- some packets from C
- one packet B2 from B (possibly the first one of a sequence of packets from B)

Now because there was no packet loss, the sequence number of the packet B2 is the sequence number of the packet B1 plus 1. As a result, considering the sequence number of any packet from C:

- If it is greater than the sequence number of B1, then: the sequence number of the packet B2 will be less or equal to the sequence number of the packets from C.
- Otherwise it is equal to or less than the sequence number of B1.

In both events, A observes a decrease or a repetition of the sequence numbers of B.

Hence, for HELLO messages, it is sufficient to check if the HELLO received from one address is equal to, or less than, the sequence number of the previous HELLO received from this address.

However, because a node may not be constantly a neighbor (and hence, quite naturally, a large number of successive HELLO messages may not be received), this condition should be checked only when there was no wrap-around, hence when the interval between the previous HELLO received and the last HELLO received from the same address is less than MIN_WRAP_AROUND_INTERVAL.

4.4.3. TC Sequence Number Consistency

Because TC messages are forwarded with the MPR flooding mechanism, first, the same message may be received several time, secondly, the packet order can be changed, especially with the use of jitter. Hence the algorithm used previously for checking consistency of HELLO messages (Section 4.4.2) can not be used as is.

Hence the following principles are used:

- The sequence number and the receving time of the last TC message for each originator is recorded.

- Each time a TC message is received from a given originator, with a given sequence number, the node checks whether if a TC message with similar identification already was received. If it was, it checks that the previous content is identical to the current content.

- If the sequence number difference (in absolute value) between the new TC and previous TC from the same originator is above a given threshold MAX_TC_DIFF_SEQ_NUM, then duplicate address can be suspected. Such an event is possible, for instance if another node sends many non-TC messages or cease to be TC generator for some time ; thus an additional check is performed on the message rate: an approximation of the message rate is computed as the "sequence number difference divided by the reception time difference". If this message rate is greater than a threshold MAX_MESSAGE_RATE, then the TC Sequence Number are deemed inconsistent.

If precise adjustement is desired for the values of MAX_TC_DIFF_SEQ_NUM, and MAX_MESSAGE_RATE (peak rate), it can be observed that one of the worst case occurs when two nodes are in conflict, and one is using the same sequence numbers of the other with a delay a little greater than DUP_HOLD_TIME.

4.5. Autoconfiguration State

4.5.1. Introduction

Each node has an "autoconfiguration state". This state is an indicator of how long the node has been in the network. The central idea, is that each time a node generates a tentative address, it should enter the network gradually, running a restrained version of the OLSR protocol. By this way, that the node can detect which addresses are being used, checking for duplicates of its own address, while avoiding to disrupt the routing tables of the other nodes, in the event that its address is actually found to be in conflict.

4.5.2. Functionning

There are exactly 4 autoconfiguration states, in each of which the behavior of the node is:

NO_ADDRESS_STATE: In this state a node does not have its own address, and it MUST NOT process and forward routing control messages genarated by other nodes. It MUST NOT participate in data forwarding. It MAY generate a tentative address by means of a pre-determined address generation method. When it generates its tentative address, it enters the HELLO_STATE.

HELLO_STATE: In this state, a node generates HELLO messages, but not topology control (TC) messages. It does not participate in MPR selection nor MPR flooding, and does not participate in data packet forwarding either. It doesn't fill the topology set nor the routing table. When it detects that it has an address conflict with other nodes based on received hello messages (rules R1 to R3, and rule R12), it MUST return NO_ADDRESS_STATE. When a pre-determined time has elapsed, in this state, without detecting address conflict, the node enters the TOPOLOGY_STATE.

TOPOLOGY_STATE: In this state, a node generates HELLO messages, but not TC messages. It processes TC messages, and performs MPR selection, but cannot be MPR itself and hence, does not forward TC messages. It fills the network topology set but not the routing table, and does not participate in data packet forwarding. When it detects that it has an address conflict with another node (based rules R1 to R12 applied to received messages), it MUST return NO_ADDRESS_STATE. When a pre-determined time elapses in the TOPOLOGY_STATE without detecting address conflict, the node enters the NORMAL_STATE.

NORMAL_STATE: In this state, the node is running the "normal" OLSR protocol, completed with the algorithms specified in this document , and without message processing/generation restrictions associated to the state. More precisely, the node generates both HELLO messages and TC messages as usual. It processes TC messages generated by other nodes and forwards them as usual based on MPR flooding. It fills the topology set, calculates routing tables and participates in data forwarding. Only nodes in the NORMAL_STATE are selected as the intermediary nodes (forwarders) in the routing table calculation. When the node detects that it has an address conflict with other nodes (according to one of the rules R1 to R12), it MUST return the NO_ADDRESS_STATE.

The behavior in each state is summarized in the following table:

State	NO_ ADDRESS_ STATE	HELLO_ STATE	TOPOLOGY_ STATE	NORMAL_ STATE
Address generation	yes	no	no	no
Selectable as MPR	no	no	no	yes
MPR selection	no	no	yes	yes
TC message forwarding	no	no	no	yes
TC message processing	no	no	yes	yes
MPR flooding	no	no	no	yes
TC message generation	no	no	no	yes
Routing table calculation (and forwarding)	no	no	no	yes
DAD rules	no	R1, R2, R3	R1-R3, R5-R7, R9, R11,R12	R1-R12
State duration (if no address change)	as long as no address	HELLO_ STATE_ DURATION	TOPOLOGY_ STATE_ DURATION	forever

4.6. Node Familiarity

The concept of "node familiarity" is introduced for use of some heuristics in MANET-DAD rules. The definition is the following: a node (or more precisely, an IP address) is "familiar" for another node, when the last one has had knowledge of existence of the first one for sufficiently long. An node which is not familiar is "unfamiliar".

In NOA-OLSR, a node (more precisely, an address) considered familiar when the time elapsed since the first time that its address has

appeared in any OLSR message, is greater than a fixed time interval NODE_FAMILIAR_TIME.

5. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [1].

6. Security Consideration

As the standard OLSR does not specify any special security measure, it makes a target for various attacks (see section 20 of the OLSR specification [3]) ; NOA-OLSR is subject to the same attacks, but also to other attacks: such as forging messages in order to deliberatly trigger some DAD rules, hence forcing an address change, or increasing OLSR control traffic. However the conditions in which such attacks can be sucessfully conducted are some conditions in which more severe attacks can be conducted with the standard OLSR protocol. Hence, in practice, vulnerability of NOA-OLSR protocol against deliberate attacks, is identical to the vulnerability of the standard OLSR protocol.

7. Acknowledgements

This work was funded by Strategic Information and Communications R&D Promotion Programme (SCOPE), Ministry of Internal Affairs and Communications, Japan.

The authors would also like to thank Sota Yoshida, Masoto Goto, Takashi Hasegawa for their valuable contributions to NOA-OLSR, along wth Yasuhiro Owada, and many other students of Information and Communication Network Laboratory for other various aspects for developping and testing of this protocol.

(document generation date: Mon Feb 6 11:34:24 2006)

8. References

8.1. Normative References

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

8.2. Informative References

[2] Weniger, K., "Passive Duplicate Address Detection in Mobile Ad hoc Networks", IEEE Journal of Selected Areas of Communications(JSAC), vol.23, No.3, 2005.

[3] Clausen, T. and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)", RFC 3626, October 2003.

[4] Ogier, R., Templin, F., and M. Lewis, "Topology Dissemination Based on Reverse-Path Forwarding (TBRPF)", RFC 3684, February 2004.

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

[6] Johnson, D., "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)", draft-ietf-manet-dsr-10 (work in progress), July 2004.

[7] Clausen, T., "The Optimized Link-State Routing Protocol version 2", draft-ietf-manet-olsrv2-00 (work in progress), August 2005.

[8] Chakeres, I., Belding-Royer, E., and C. Perkins, "Dynamic MANET On-demand (DYMO) Routing", draft-ietf-manet-dymo-03 (work in progress), October 2005.

[9] Ruffino, S., Stupar, P., and T. Clausen, "Autoconfiguration in a MANET: connectivity scenarios and technical issues", draft-ruffino-manet-autoconf-scenarios-00 (work in progress), October 2004.

[10] Mase, K. and C. Adjih, "A common framework for autoconfiguration of stand-alone ad hoc networks draft-mase-autoconf-framework-01", Feburary 2006.

Index

Authors' Addresses

K. Kenichi Mase
Niigata University
Niigata 950-2181,
Japan

Phone: +81 25 262 7446
Email: mase@ie.niigata-u.ac.jp
URI: http://www.net.ie.niigata-u.ac.jp/

Cedric Adjih
INRIA
(Domaine de Voluceau, Rocquencourt, France)

Email: cedric.adjih@inria.fr

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.