INTERNET-DRAFT
IETF MANET Working Group
Expiration: 14 August 2005

Cedric Adjih
Saadi Boudjit
Philippe Jacquet
Anis Laouiti
Paul Muhlethaler

INRIA Rocquencourt, France

14 February 2005

Address autoconfiguration in Optimized Link State Routing Protocol

draft-laouiti-manet-olsr-address-autoconf-00.txt

Status of this Memo

Copyright Notice

Abstract

One of the MANET protocols which have been recently promoted to experimental RFC is the OLSR routing protocol[3]. This document aims at complementing the OLSR routing protocol specifications to handle autoconfiguration. The corner stone of this autoconfiguration protocol is an advanced duplicate address detection algorithm. We propose a comprehensive autoconfiguration scheme whose basic idea is to avoid conflicts in the 2-hop neighborhood of each node. We have designed two algorithms to perform this task. These algorithms are shown to work in any case of multiple conflicts, especially during network mergers.

Table of Contents

1. Introduction

Mobile Ad hoc NETworks (MANETs) are infrastructure-free, highly dynamic wireless networks, where central administration or configuration by the user is very difficult. In hardwired networks nodes usually rely on a centralized server and use a dynamic host configuration protocol, like DHCP[1], to acquire an IP address. Such a solution cannot be deployed in MANETs due to the unavailability of any centralized DHCP server. For small scale MANETs, it may be possible to allocate free IP addresses manually.

However, the procedure becomes impractical for a large-scale or open system where mobile nodes are free to join and leave. Most of the autoconfiguration algorithms proposed for ad hoc networks are independent of the routing protocols and therefore, generate a significant overhead. Using the genuine optimization of the underlying routing protocol can significantly reduce the autoconfiguration overhead.

Research on automatic configuration of IP addresses for MANET is relatively less frequent. The IPv6 and ZEROCONF working groups of the IETF deal with autoconfiguration issues but with a focus on wired networks. Automatic address allocation is more difficult in a MANET environment than in wired networks due to instability of links, mobility of the nodes, the open nature of the mobile ad hoc networks, and lack of central administration in the general case. Thus performing a DAD (Duplicate Address Detection) generates more complexity and more overhead in ad hoc networks than in wired networks where protocols such as DHCP[1] and SAA [2] can be used.

In this document we will describe an autoconfiguration solution for the OLSR protocol. This solution is based on an efficient Duplicate Address Detection (DAD) algorithm which takes advantage of the genuine optimization of the OLSR protocol. We actually propose two solutions to handle multiple address conflicts. They have the same main basic idea which is to ensure the absence of conflicts in the 2-hop neighborhood of each node.

2. Terminology

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119 [5].

For the OLSR description and terminology, the reader should refer to the OLSR RFC [3].

3. Overview

Our proposed autoconfiguration algorithm is based on two steps. In the first step, an IP address is selected by the arriving node and this latter can join the ad hoc network. Numerous schemes can be used to select this IP address for instance the node can perform a random selection; another technique is that a neighbor node selects this IP address for the arriving node.

After this first step has been performed, the second step can take place. The aim of this step is to detect potential address duplications on run. To perform this task a DAD algorithm is started on this newly configured node. This DAD algorithm allows the newly configured node to state whether the selected address is duplicated or not in a proactive manner. If such a case occurs, a node can change its address with respect to some specified criteria.

In such a case a new address can be chosen. The DAD algorithm uses a special control packet called MAD for ‘‘Multiple Address Declaration’’. This control packet includes the node address and a node identifier. This packet is broadcast in the network, thus all the network nodes must receive this packet. The duplicate detection uses the node identifier. If a node receives an MAD message with a different identifier than its own, an address duplication is detected. To spare the channel bandwidth the MAD packet should be broadcasted using the MPR flooding.

On the other hand, duplicated addresses may be the origin of MPR election and flooding corruption, which may induce errors in packet delivery, particularly the MAD packets. Consequently, we need first to ensure the MPR election from those conflictual situations. We propose two algorithms to achieve this task:

1 In first algorithm, we suggest to ignore the MPR mechanism for the first hop, and relay each originated MAD message by all the neighbors to the 2-hop neighbors. In this manner possible address conflict could be detected and resolved. The direct consequence is that the MPR set will be fixed.

2 In the second one, we propose to modify the OLSR Hello messages and the MPR election algorithm. The calculation will be based on the address and the node identifier to guarantee the uniqueness of the direct neighbors and the 2-hop neighbors. This leads to a correct MPR set, hence,we are sure that the MAD messages reach all the nodes.

In the sequel we first present the MAD message used to detect the address conflict. Then, we introduce the two proposed DAD algorithms tailored for OLSR protocol. And, finally we give some address assignment and conflict resolution issues.

4. Duplicate address detection and MAD message description

In order to detect address conflicts, each node diffuses periodically a special message that we call a MAD for ‘‘Multiple Address Declaration’’ to the entire network[5]. This control packet includes the node addresses and the node identifier. The node identifier is a sequence of bits of fixed length L which is randomly generated. Hence we are using the standard idea that the probability of two nodes having the same node identifier is low, and the probability of at least one address collision with N nodes, which is the well known ‘‘birthday problem’’, can be set arbitrarily low by choosing a large enough value of L (eight bytes are enough to code the random identifier if we consider a network with a maximum of 10000 nodes [4]). The MAD message format is depicted in the following figure.

```
  0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                  Originator node identifier                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    OLSR Interface Address                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    OLSR Interface Address                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

A node detects an address conflict when it receives an MAD message having the same address as its own, but with a different identifier. These situations may happen during network mergers. Actually other nodes will detect the conflict. These nodes could announce the conflict using a special control message. However this approach may induce broadcast storm since many nodes may announce the conflict and special care must be taken to avoid this effect. For that reason we do not recommend this way. An efficient manner to notify the address duplication to the nodes in conflict, consists in allowing the MAD packets to reach all the nodes in the network. To save the channel bandwidth the MAD packets should be broadcasted using the MPR flooding. Actually, applying OLSR relaying optimization rules as they are defined, may not be sufficient to ensure diffusion in some conflictual cases.

5. Duplicate Address Detection mechanism

Duplicate addresses are detected by the periodic exchange of MAD messages. We propose two different algorithms to ensure the delivery of these messages to the participants in the network in order to discover these conflicts. The first one suggests to modify the MPR flooding rules for MAD messages diffusion. Where in the second one, we propose to modify the OLSR MPR election, which is done on the basis of the node identifier and address interface. This guarantee correct MPR sets, that cover the 2-hop neighbors even in the presence of duplicate addresses.

5.1. First algorithm

In this first approach we will add rules to the OLSR MPR flooding.

5.1.1. First rule

The property that we will add is actually extremely simple. We weaken the relaying condition for nodes who are in the 2-hop neighborhood of a node who is sending an MAD message. When these neighbor nodes receive an MAD message, they must relay the MAD message irrespectively of the relaying conditions of the OLSR MPR flooding algorithm. We call this first rule, rule 1.

5.1.2. Second rule

When a node receives an MAD message from a neighbor node with a given IP address and a given Node-ID, and another MAD message with the same address but with a different Node-ID, it must relay this latter MAD message irrespectively of the MPR flooding rules. We call this second rule, rule 2.

5.1.3. Full algorithm

Let us recall the assumptions here.

Each node A periodically sends a message M including:

1 The originator address of A, Orig_A, in the OLSR message header.

2 The message sequence number, mssn, in the OLSR message header.

3 The node identifier ID_A (a string of bits) in the message itself.

The message is propagated by MPR flooding to the other nodes ; but for DAD-MPR Flooding, the duplicate table of OLSR is modified, so that it also includes the node identifier list in the duplicate tuple. That is, a duplicate tuple, includes the following information:

1 The originator address (as in OLSR standard duplicate table).

2 The message sequence number (as in OLSR standard duplicate table).

3 The list of node identifiers.

The detailed algorithm for DAD-MPR Flooding is the following:

1 When a node receives a message M from node B with originator Orig_A, with message sequence number mssn, and with node identifier ID_A, it performs the following tasks:

1.1 If a duplicate tuple exists with the same originator Orig_A, the same message sequence number, and ID_A is in the list of node identifiers, Then, the message is ignored (it has already been processed). The algorithm stops here.

1.2 Else one of the following situations occurs :

1.2.1
A duplicate tuple exists with the same originator Orig_A and the same message sequence number, but ID_A is not in the list of node identifiers: then, a conflict is detected (address Orig_A is duplicated). ID_A is added to the list of node identifiers.

1.2.2
A duplicate tuple exists with the same originator Orig_A, but with a different message sequence number and ID_A is not in the list of node identifiers: then, a conflict is detected (address Orig_A is duplicated). A duplicate tuple is created with the originator address, message sequence number and list of node identifiers containing only ID_A.

1.2.3
No duplicate tuple exists. A new one is created with the originator address, message sequence number and list of node identifiers containing only ID_A.

1.3 The MAD messages should be relayed if one or more of the following rules are met:

1.3.1
The node B is the source of the MAD message i.e. it has the originator address Orig_A.

1.3.2
B had chosen this current receiving node as an MPR.

1.3.3
One of the conflicting nodes is a neighbor of the node detecting the duplication. In such a case, the TTL value of the MAD message showing the conflict is set to one before its retransmission. This also applies even if the current node has not been selected as an MPR by the previous message sender.

5.2. Second algorithm

An issue with the previous approach is that the conflicts at 2-hop neighbors must be resolved before one can be sure that the MAD messages are successfully transmitted within the entire network. An ideal property would be that the MAD messages reach all the nodes in the network irrespectively of potential address duplications. This property can be achieved if the MPR flooding continues to work in presence of address duplication. A solution is then to base the selection of MPR not on addresses but on node identifiers. With the assumption that node identifiers are globally unique in the network, one can be sure that there will not be identifier duplications at two hops of a given node and thus the selection of MPRs will be correct. This solution can be simply implemented, the selection of the MPRs must follow the principle defined in the OLSR protocol except that the base for the selection must be the node identifiers i.e. the 2-hop coverage must be obtained not on the addresses but on the node identifiers.

To be able to do so, hello packets must be modified such that the following information will be given in the hello message :

1 the node identifier of the originator node of the hello message,

2 the node identifier of the nodes (actually the node interface addresses) advertised in the hello messages. The modified Hello message format is specified in the following figure.

```
  0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Reserved             |     Htime     |  Willingness  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                  Originator node identifier                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Link Code   |   Reserved    |       Link Message Size       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Neighbor Interface Address                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                     Neighbor identifier                       |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|                  Neighbor Interface Address                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                     Neighbor identifier                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                             .  .  .                           :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Link Code   |   Reserved    |       Link Message Size       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Neighbor Interface Address                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                     Neighbor identifier                       |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
|                  Neighbor Interface Address                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                     Neighbor identifier                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:                                                               :
                              (etc.)
```

The drawback of this mechanism is that it introduces a significant extra overhead.

6. Address assignment

We have two main ways to allocate an address to a newly arriving node. The first way is to assign this node a random address in the pool of addresses that can be allocated and then to rely on the DAD algorithm to discover potential conflicts. The second way is to ask for the help of a neighbor node. This neighbor will be able to propose to the newly arriving node a configuration address. Since a neighbor node must in principle receive the MAD messages of all nodes in the network, it can maintain a pool of non-affected addresses. A newly arriving node can choose between these two approaches.

7. Pool of addresses

The pool of addresses could be for local use only. For example, it could be reserved by the IANA authority for local MANET forwarding (i.e, those addresses must not be forwarded outside the MANET network, nor reached from outside). A second possibility consists in relying on some machines which will announce the prefix to use for address autoconfiguration for this MANET network. These machines could be connected to the internet, and act as gateways. In this case, the addresses may be global addresses, and could be seen from outside.

8. Resolution of a conflict

When two nodes A1 and A2 are configured with the same IP address and assuming that there is no packet loss, each of these two nodes will receive the MAD message of the other node. Thus the nodes where the conflict lies are bound to discover the conflict. A simple rule to solve this conflict will be: the node in conflict with the smallest identifier changes its address. Since this node knows via the reception of the MAD control messages the already assigned addresses, the new address must be selected at random among the addresses that are believed to be free.

9. Security Considerations

This memo does not specify any security considerations.

10. Authors' Addresses

The authors are listed in alphabetical order.

Cedric Adjih
Project HIPERCOM
INRIA Rocquencourt
BP 105
78153 Le Chesnay Cedex, France
Phone: +33 1 3963 5215
Email: Cedric.Adjih@inria.fr

Saadi Boudjit
Project HIPERCOM
INRIA Rocquencourt
BP 105
78153 Le Chesnay Cedex, France
Phone: +33 1 3963 5039
Email: Saadi.Boudjit@inria.fr

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

Anis Laouiti
Project HIPERCOM
INRIA Rocquencourt
BP 105
78153 Le Chesnay Cedex, France
Phone: +33 1 3963 5088
Email: Anis.Laouiti@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

11. References

[1] R. Droms, Ed.,J. Bound, B. Volz, T. Lemon, C. Perkins, M. Carney, ``Dynamic Host Configuration Protocol for IPv6 (DHCPv6)'', IETF RFC 3315, July 2003.

[2] S. Thomson, T. Narten, ``IPv6 Stateless Address Autoconfiguration'', IETF RFC 2462, December 1998.

[3] T. Clausen, Ed., P. Jacquet, Ed., Optimized Link State Routing Protocol. Request for Comments (Experimental) 3626, Internet Engineering Task Force, October 2003.

[4] S.Boudjit, A. Laouiti, P. Muhlethaler, C. Adjih, "Duplicate address detection and autoconfiguration in OLSR", INRIA Research Report-5475, Jan 2005.

[5] A. Laouiti, S. Boudjit, P. Minet and C. Adjih, "OLSR for IPv6 networks", Proceedings of Med-Hoc-Net, June 2004,Bodrum, Turkey.

12. Full Copyright Statement