A Simple address Autoconfiguration Mechanism for OLSR

Thomas Heide Clausen, Emmanuel Baccelli

A Simple address Autoconfiguration Mechanism for OLSR

Thomas Heide Clausen
PCRI – Pole Commun de Recherche en Informatique du plateau de Saclay
CNRS, Ecole Polytechnique, INRIA, Universite Paris Sud
Laboratoire d'Informatique, Ecole Polytechnique
91128 Paliseau, FRANCE
Email: T.Clausen@Computer.org

Emmanuel Baccelli
Project Hipercom, INRIA Rocquencourt
78153 Le Chesnay Cedex, FRANCE
Email: Emmanuel.Baccelli@Inria.fr

***Abstract*—**
In this paper, we develop a simple autoconfiguration mechanism for OLSR networks. The mechanism aims at solving the simple, but common, probem of one or more new nodes emerging in an existing network. We propose a simple solution, which allows these new nodes to acquire an address and participate in the network. Our method is simple, both algorithmically and in the requirements to the network. While we recognize that this is a partial solution to the general autoconfiguration problem, we argue that the mechanism described in this paper will satisfy the requirements from a great number of real-world situations.

I. Introduction

A Mobile Ad-hoc NETwork (MANET) is a collection of nodes which are able to connect on a wireless medium forming an arbitrary and dynamic network, routing traffic through multi-hop-paths in order to ensure connectivity between any two nodes in the network. Implicitly herein is the ability for the network topology to change over time as links in the network appear and disappear.

In order to enable communication between any two nodes in such a MANET, a routing protocol is employed. The abstract task of the routing protocol is to discover the topology (and, as the the network is dynamic, continuing changes to the topology) to ensure that each node is able to acquire a recent image of the network topology for constructing routes.

One of the proposed routing protocols for MANETs is OLSR [1]. A proactive routing protocol, OLSR ensures that all nodes at all times have sufficient topological information to construct routes to all destinations in the network. This is achieved through periodic message exchange.

An issue, complimentary to that of routing, emerges with respect to bootstrapping of the network. OLSR do very well with the task of discovering paths in a MANET , however a prerequisite to the correct functioning of OLSR, and indeed of any MANET routing protocols, is, that all nodes are identifiable by an unique IP-address. Subsequently, a mechanism for assigning (unique) addresses to MANET nodes is required.

A particularity of MANETs is, that the roles of "terminal" and "network forming node" (router) are not clearly seperate. In principle, all nodes may act in both capacities simultaneously. An additional constraint is, that no assumptions with respect to a preexisting infrastructure can be made. Traditional mechanisms for host autoconfiguration, such as DHCP [4] or ZeroConf [8] or similar mechanisms all assume the presense of a "server", which can coordinate and assign addresses. Further, these mechanisms work on the assumption that direct communication between the "server" and all hosts in the local network is possible. Due to the multi-hop nature of MANETs, direct communication between an arbitrary host in the network and (any) server cannot be assumed.

In order to ensure the true autonomy of MANETs, a specific mechanism – or adaptation of mechanism – for address autoconfiguration of MANETs is required.

A. Problem Statement

The issue of autoconfiguration in MANETs is complex since, for a complete solution, issues such as ensuring uniqueness of addresses in independant MANETs which later merge, must be addressed: independant MANET must somehow select non-overlapping address-spaces, duplicate address detection, conflict resolution – and the issue of how to deal with ongoing data streams without loosing data or the requirement of specific application behavior.

In this paper, we aim for a simple solution to a simple problem: the connected case. A common situation occurs, in which an efficient and simple address autoconfiguration mechanism is desireable and sufficient. This situation is, where an OLSR network acts as an edge-extension to the Internet. I.e., nodes are interrested in maintaining connection to each other and to the Internet. The implication is also, that nodes join or leave the OLSR network, but do not migrate (alone or in groups) between OLSR-networks with the expectation of maintaining connectivity. The topic of nodes migrating between OLSR networks may better be addressed through mechanisms such as NEMO [3].

The mechanism, developed in this paper, is therefore targeted explicitly at the connected case described above. We recognize that this is a particular solution to a particular problem, and we aim at developing a simple and light-weight mechanism, efficient for these stated scenarios.

The address autoconfiguration mechanism in this paper is developed as an extension to OLSR [1], taking direct advantage of the mechanisms and features of OLSR.

B. Paper outline

The remainder of this paper is organized as follows: section II presents OLSR with sufficient details to define the address autoconfiguration mechanism as an extension to OLSR. Section III presents an overview of the address autoconfiguration mechanism. Section IV, sections V and section VI, describe the beaconing employed, the mechanism through which a locally unique address is acquired, and the mechanism through which a globally unique mechanism is assigned to an arriving node.

Section VII evaluates the overhead of the mechanism proposed. The paper is concluded by section VIII.

II. OLSR Protocol Overview

In this section, relevant aspects of OLSR [5] are described, with the purpose of designing an address autoconfiguration mechanism which takes advantage of the possible optimizations in OLSR and integrates as a natural extension to the extension mechanism of OLSR. For further details on OLSR, as well as on performance characteristics of OLSR, please refer to [5] and [2].

OLSR is a proactive link-state routing protocol. I.e. it employs periodic control message exchange in order to accomplish topology discovery and topology maintenance. The result of this message exchange is a topology map in each node, from which a routing table can be constructed.

More specifically, OLSR employs two types of control messages: HELLO messages and TC messages.

HELLO-messages are exchanged periodically between neighbor nodes, to permit tracking the status links to neighbor nodes, including disappearing and emerging nodes.

TC-messages are likewise emitted periodically to diffuse link-state information to the entire network. TC-messages are flooded using an optimized flooding mechanism, specific to OLSR [1].

Control-messages, intended for all nodes in the network, are flooded using a mechanism called MPR-flooding. The objective of MPR-flooding is to reduce the cost of performing a flooding operation. This is achieved through having each node select a minimal set of "relay nodes" (called MPRs), responsible for relaying flooded packets. Figure 1 illustrates classical flooding (left) and MPR flooding (right), with the packet originated by the central node and retransmitted as indicated by the arrows. [7] has more details regarding MPR flooding.

As a platform for extensions, OLSR control traffic is transmitted in an unified packet format, allowing messages to be piggybagged together, as well as allowing extensions to take advantage of the MPR flooding mechanism. The OLSR packet format is given in figure 2.

As can be seen in figure 2, a packet is a collection of messages, each with individual headers, permitting individual treatment (including flooding behavior) of each message. Refer to [5] for further details.

[1] For details regarding TC message generation, MPR flooding etc., please refer to [5] and [7].

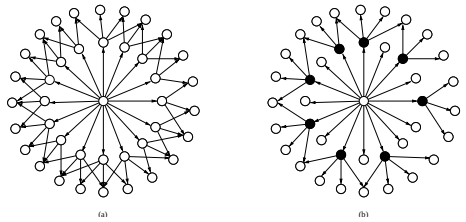

Fig. 1. Two hop neighbors and "multipoint relays" (the solid circles) of a node. (a) illustrates the situation where all neighbors retransmit a broadcast, (b) illustrates where only the MPRs of a node retransmit the broadcast.

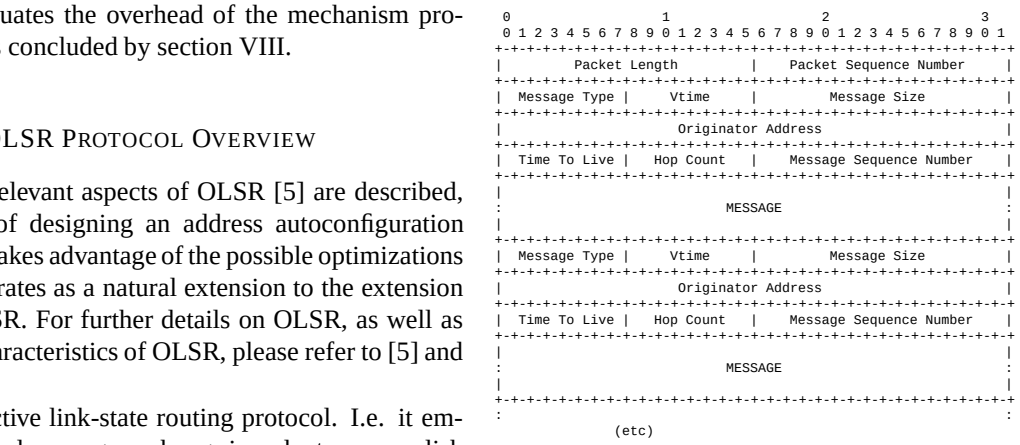

Fig. 2. Generic OLSR packet format. Each packet encapsulates several control messages into one transmission.

III. Overview and Philosophy

With the description of OLSR in section II, as well as the problem statement in section I-A, this section will outline the functioning of our address autoconfiguration mechanism.

We will use the following two terms for the remainder of this paper: a *new node* is a node which is not yet assigned an address, and thus not is part of an OLSR network. An *OLSR-node* is a node which is assigned an address and which is part of the OLSR network. A *configurating node* is an OLSR-node, which is currently assisting a new node in acquireing an address.

- OLSR-nodes behave as specified by [5]. Additionally, they emit ADDR_BEACON messages, to signal to new nodes that they may act as configurating nodes. This is detailed in section IV.
- New nodes do not emit HELLO and TC messages, however listen for ADDR_BEACON messages. From among the OLSR-nodes emitting ADDR_BEACON messages, one configurating node is selected, and a request-for-address-configuration is issued through an ADDR_CONFIG message. The goal is for the configurating node to provide the new node with first a temporary address, then a permanent global address. This process of acquireing a local, temporary address is detailed in section V, and the task of acquireing a global address is detailed in section VI.

IV. Local Beaconing

Each OLSR-node, must ensure that it has the ability to provide temporary addresses from a private address space to new

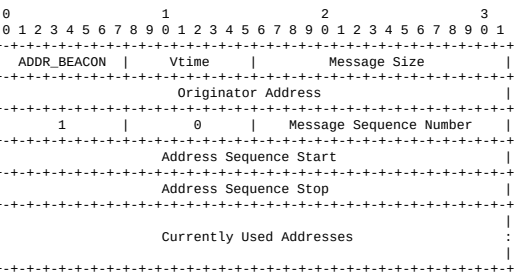

Fig. 3. ADDR_BEACON message format.

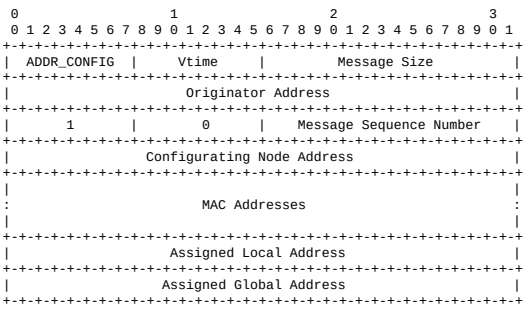

Fig. 4. ADDR_CONFIG message format.

nodes. It is important that, within a region, these temporary addresses are unique, i.e. that no two new nodes within the same neighborhood are assigned the same temporary address. In order to ensure this, a predefinined address space is allocated for use for "temporary addresses". The task is to ensure that this address space is divided, without overlap, between nodes in a region of the network:

- Each OLSR-node will, independently, select a continous address-sequence from the address-space allocated for "temporary addresses";
- Each OLSR-node will signal, in periodic ADDR_BEACON messages, this selected sequence. ADDR_BEACON's are transmitted to neighbor nodes only, i.e. are not forwarded;
- Each node will record the address-sequences, selected by all neighbor nodes.
- If, upon receiving an ADDR_BEACON message, a node detects that a conflicting address-sequence is selected, arbitration must happen.
  - If no nodes in the conflict are acting as configurating nodes, arbitration is carried out simply by having the conflicting node with the lowest ID (IP-address) select a new, unused address-sequence.
  - If one or more conflicting nodes are acting as configurating node(s), arbitration must aim at allowing ongoing configuration sessions to complete. To accomodate this, all configuration nodes "narrow" their selected address-sequence to contain only the address(es) which are currently assigned to new nodes. This is included in the next ADDR_BEACON. Nodes which are not currently acting as configuration nodes, select non-conflicting address sequences. If a conflict between two configurating nodes remains, the node which has the lowest ID (IP address) must yield.

The ADDR_BEACON message has the format specified in figure 3. [5] specifies the values of Message Size, Originator Address, Message Sequence Number and Vtime.

In case of "narrowing down" the address-sequence to only currently used addresses, the "Address Sequence Start" and "Address Sequence Stop" are both set to zero.

Each node will send ADDR_BEACON messages, listing both its address-sequence and the addresses which currently in use'. In case of a conflict, a recipient node can detect if the node with which it is conflicting is active as configurating node and if both nodes are active as configurating nodes, detect a conflict in the addresses actually selected.

ADDR_BEACON messages are transmitted piggybagged in the same OLSR packet as OLSR HELLO messages.

V. Acquireing a Local Address

The first task of a new node is to associate itself with an OLSR-node. Thus, the new node listens for ADDR_BEACON messages and selects one "configurating node". An ADDR_CONFIG message is then created and transmitted, in order to request address configuration from the selected configurating node. Abcent an IP-address, the MAC address of the new node must be included, to uniquely identify the new node.

Upon receiving an ADDR_CONFIG message, the configurating node assigns a local address to the new node, and signals this assignment through an ADDR_CONFIG message. Additionally, the configurating node marks the assigned address as "used" in its ADDR_BEACON messages.

Upon receiving a local address through an ADDR_CONFIG message, the new node will start sending HELLO messages, including only the configurating node as neighbor. This allows the new and configurating node to track each other (i.e. allows both nodes to "reset", should the link disappear before a global address was assigned to the new node), while does not cause the new node to be advertised to the network: adverticing a node with a non-unique address might lead to data loss, loops etc.

If a new node does not receive an ADDR_CONFIG reply, it may either retransmit the ADDR_CONFIG to the same configurating node – or give up and select an alternative configuration node. Absent the HELLO message exchange described above the configuration node may retransmit its ADDR_CONFIG reply – or give up, in which case any temporarilly assigned addresses will be reclaimed.

An ADDR_CONFIG message has the format specified in figure 4. [5] specifies the values of Message Size, Originator Address, Message Sequence Number and Vtime.

If the "Assigned Local Address", "Assigned Global Address" and "Originator Address" fields are both set to zero, the ADDR_CONFIG message is a request to the "Configurating Node" to perform local address assignment.

If the "Assigned Local Address" is non-zero (i.e. contains an actual address) and "Originator Address" is non-zero, but the "Assigned Global Address" field is set to zero, the ADDR_CONFIG message is an assignment of a temporary local address. I.e. this is the reply, generated by a configurating node.

The "Assigned Global Address" field is discussed in section VI.

VI. Global Address Assignment

When the HELLO message exchange commences between the new and configurating node, local address assignment is completed, and acquireing a global address can commence. The configuration node is in charge of acting on behalf of the new node, wrt. acquireing a global address. Since the configuration node is already part of the OLSR network, a multitude of different mechanisms can be employed. One such mechanism for acquireing a global adddres would be for the configurating node to act as a modified DHCP proxy [6] and transmit a request to an existing DHCP server in the network.

Another option would be to consult the nodes topology table. This table (in a stable state) contains all destinations (thus addresses) of the network. The configurating node can thus pick a non-used address and assign to the new node. To prevent duplicate address assignment, the configurating node includes the selected address in a few TCs. If a node receives a TC containing its own address (or an address, which the node has claimed for a new node) AND if the originator of the message is not the node itself nor an MPR of the node, a duplicate address assignment is detected. The detecting node can then communicate this to the originator of the "offending" TC, with the purpose of resolving the conflict.

Once the configurating node has acquired a globally unique address, this is assigned to the new node through an ADDR_CONFIG message, containing the "Assigned Local Address" and "Originator Address" as before, but with a non-zero address in the "Assigned Global Address" field. This is then the ticket for the new node to participate fully in the OLSR-network.

The configurating node will continue to transmit this ADDR_CONFIG message periodically until either the HELLO messages from the new node's assigned local address cease, or until an ADDR_CONFIG message from the new node is received, listing the new nodes global address in both the originator field and the "Assigned Global Address" field, and with the "Assigned Local Address" and "MAC address" as before.

VII. Overhead evaluation

The overhead, incuring from the mechanism specified in this paper, comes from primarilly three sources: periodic beaconing of ADDR_BEACON messages address request/replies through ADDR_CONFIG messages and discovery of a globally unique address.

ADDR_BEACON messages and ADDR_CONFIG messages are local, i.e. no flooding operations incur. ADDR_CONFIG messages are furthermore only transmitted while nodes are being configured, and are of limited size (24 bytes + size of MAC address). Each configuration cycle incurs 4 messages. The overall overhead, incured through this procedure, is negligible.

ADDR_BEACON messages are transmitted in the same OLSR packets as OLSR HELLO messages (MTU permitting), thus the number of transmissions required remains constant as compared to OLSR. Except when an node configuration is ongoing, the additional overhead incured from ADDR_BEACON amounts to 20 bytes.

Discovery of a globally unique message depends on the mechanism employed. Assuming the decentralized mechanism, where an unused address is picked from the topology table and is probed through including this address in a TC emission, the additional overhead per TC message for that node is 4 bytes. This is offset by the fact that if address is assigned to the new node, topological information is already present in the network, allowing the node immediate participation.

VIII. Conclusion

In this paper, we have presented a simple solution to a common case of autoconfiguration of a new node in an existing OLSR network. Our approach is based on a "new" node associating with an existing node, and thereby acquireing first a local, then global, address by way of this existing node. We notice, that our approach permits both the use of "serverless" autoconfiguration, taking advantage of the fact that OLSR is a proactive table-driven protocol in order to select global addresses and verify their uniqueness, as well as the use of existing configuration servers, such as DHCP-servers.

Our proposed solution is simple, both conceptually and in terms of the overhead incuring, however solves the problem of autoconfiguration in an OLSR network, where the situation is that a node emerges in an existing network.

We have described the situation addressed by this protocol in terms of "the connected case": a requirement for the mechanism to work is, that a new node can associate itself with a node which is already part of the network. The implication of this is, that we avoid the "disconnected case", where two MANETs configure independantly, then merge at a later point with potential addressing conflicts which must be resolved. We believe that the "connected case" is a quite common situation (e.g. when OLSR networks are applied as extensions to an existing infrastructure – in which case the network formation naturally happens from the existing infrastructure), and the a simple solution, such as the one presented in this paper, is required.

References

[1] Thomas Clausen, Gitte Hansen, Lars Christensen, and Gerd Behrmann. The optimized link state routing protocol - evaluation through experiments and simulation. In *Proceeding of Wireless Personal Multimedia Communications*. MindPass Center for Distributed Systems, Aalborg University, Fourth International Symposium on Wireless Personal Multimedia Communications, September 2001.

[2] Thomas Heide Clausen, Philippe Jacquet, and Laurent Viennot. Comparative study of routing protocols for mobile ad-hoc networks. In *Proceedings of IFIP Med-Hoc-Net 2002*. IFIP, September 2002.

[3] V. Devarapalli, R. Wakikawa, A. Petrescu, and P. Thubert. Network mobility (nemo) basic support protocol. Internet Draft, draft-ietf-nemo-basic-support-03.txt, June 2004, Work in progress.

[4] R. Droms. Dynamic host configuration protocol, March 1997. RFC 2131.

[5] Thomas Clausen (ed) and Philippe Jacquet. RFC 3626: Optimized link-state routing protocol. Internet Engineering Task Force, Request For Comments (experimental), October 2003.

[6] M. Patrick. Dhcp relay agent information option, January 2001. RFC 3046.

[7] Amir Qayyum, Laurent Viennot, and Anis Laouiti. Multipoint relaying: An efficient technique for flooding in mobile wireless networks. Technical report, Project Hipercom, INRIA Rocquencourt, 2000. INRIA research report RR-3898.

[8] A. Williams. Requirements for automatic configuration of ip hosts. Internet Draft, draft-ietf-zeroconf-reqts-12.txt, September 2002, Work in progress.