HAL
open science

Yet Another Autoconf Proposal (YAAP) for Mobile Ad Hoc NETworks

Ulrich Herberg, Thomas Heide Clausen

To cite this version:

Ulrich Herberg, Thomas Heide Clausen. Yet Another Autoconf Proposal (YAAP) for Mobile Ad Hoc NETworks. 2010 Sixth International Conference on Mobile Ad-hoc and Sensor Networks (MSN 2010), Dec 2010, Hangzhou, China. pp.20-26, 10.1109/MSN.2010.48 . hal-02263421

HAL Id: hal-02263421

https://polytechnique.hal.science/hal-02263421

Submitted on 4 Aug 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Yet Another Autoconf Proposal (YAAP) for Mobile Ad hoc NETworks

Ulrich Herberg, Thomas Clausen
Laboratoire d'Informatique (LIX), Ecole Polytechnique, France
Email: Ulrich@Herberg.name, Thomas@ThomasClausen.org

***Abstract*—This paper addresses the issues of automatic address and prefix configuration of MANET routers. Specifically, the paper analyzes the differences between "classic IP networks" and MANETs, emphasizing the interface, link, topology, and addressing assumptions present in "classic IP networks". The paper presents a model for how this can be matched to the specific constraints and conditions of a MANET – *i.e.*, how MANETs can be configured to adhere to the Internet addressing architecture. This sets the stage for development of a MANET autoconfiguration protocol, enabling automatic configuration of MANET interfaces and prefix delegation. This autoconfiguration protocol is characterized by (i) adhering strictly to the Internet addressing architecture, (ii) being able to configure both MANET interface addresses and handle prefix delegation, and (iii) being able to configure both stand-alone MANETs, as well as MANETs connected to an infrastructure providing, *e.g.*, globally scoped addresses/prefixes for use within the MANET. The protocol is specified through timed automatons which, by way of model checking, enable verification of certain protocol properties. Furthermore, a performance study of the basic protocol, as well as an optimization hereto, is conducted based on network simulations.**

I. Introduction

A prerequisite for operation of a routed multi-hop network includes that the network interfaces of routers are appropriately configured with IP addresses, so as to enable exchange of routing protocol control messages. Another prerequisite is that routers providing connectivity for directly connected hosts have prefixes delegated – for configuring these hosts with IP addresses (to make enable these hosts to communicate), and for efficiently sharing information about these through the network routing protocol control messages (to render these hosts reachable also from across the multi-hop network). While these may sound as relatively benign tasks – after all, the Internet works – they are not entirely trivial even in the Internet: prefix delegation is centrally managed, on the top-level by ICANN [1], by manually "subnetting" by the network operator, otherwise. Router interface configuration for router is a function of the characteristics of these router interfaces and links. Furthermore, the "configuration of the Internet" is relatively stable: an individual link may disappear and reappear, but new links do not often form spontaneously and arbitrarily – a consequence of links typically being both planned and fixed, *e.g.* in form of cables.

A Mobile Ad hoc NETwork (MANET) is by nature different. A typical "academic" description of a MANET may be *"a collection of mobile routers, communicating among themselves over wireless links and thereby forming a spontaneous, dynamic, arbitrary graph"* – calling out (wireless) interface characteristics and arbitrary and spontaneous link formation as challenges, in contrast to the "planned and fixed" expectations of the Internet. Fundamentally, in a MANET, any pair of router interfaces may at some point in time be able to communicate directly with each other. An additional assumption commonly made is, that in a MANET there is no network operator. Consequently, MANET router configuration must be such that reconfiguration is rarely needed – and if needed, must occur automatically.

In order to allow proper operation of MANET routing protocols, the interfaces of MANET routers must be appropriately configured with IP addresses – and prefixes must be available on MANET routers wishing to provide connectivity through the MANET to attached hosts. In order to allow proper operation of a MANET as an IP network, *i.e.*, to enable the use of classic IP protocol stacks, and to thereby allow proper integration of MANETs into the larger Internet, the configuration of MANET router interfaces and attached hosts must conform to the Internet architecture. An interface being configured in a given manner entails that the IP protocol stack, as well as applications using that interface, make certain assumptions as to the connectivity over that interface. Failure to satisfy these assumptions may – popularly speaking – break the Internet.

In order to properly set the stage for development of a MANET router autoconfiguration mechanism, the remainder of this section will elaborate on the addressing and connectivity assumptions made for IP networks, formalize the particularities and challenges presented by MANETs in this regard, and discuss an addressing model which is both suitable for MANETs while respecting the IP network assumptions – *i.e.*, such that MANETs do not "break the Internet".

A. The Internet is a Graph: IP Link and Addressing Model

Fundamentally, the Internet consists of three distinct components: hosts, routers and links. Hosts are devices which act as end-points for communication, which run user applications, and which have no responsibility for maintaining the Internet infrastructure. As such, their interest (in the context of network formation) is limited to "being connected to a router" via a link. Routers and links are the core notions in the Internet, where a link is assumed to have the following properties:

- IP datagrams are not forwarded at the network layer when communicating between network interfaces which are on

the same link; hence

- TTL of IP datagrams is not decremented when communicating between network interfaces on the same link;
- IP datagrams with a TTL of 1 can be delivered to all network interfaces on the same link and;
- Link-local multicasts and broadcasts are received by all network interfaces on the same link – without forwarding.

A router is an entity which determines over which link a given IP datagram is to be forwarded and, in doing so, decrements the TTL of that IP datagram.

When assigning an IP address to a network interface, this network interface is also configured with a subnet prefix, such that the following constraint is respected:

- All network interfaces configured with addresses from within the same prefix `p::`, and with the same prefix `p::` assigned to the interfaces, can communicate directly with one another.

It follows from the above that the notion of "IP link" is tied with the notion of an "IP Subnet" (IPv4) or a prefix (IPv6), in that all network interfaces which are configured with the same subnet address or prefix are considered to be on the same IP link and thus that for communication between routers within the same subnet, no forwarding is required and no decrement of TTL/hop-limit is performed.

All network interfaces within the same subnet, are assumed to be connected to the same classic IP link, as described above. The inverse is not necessarily true: in some network configurations, interfaces connected to the same classic IP link may be configured with addresses from within different prefixes or subnets. Specifically for routers, communication is allowed between interfaces of two different routers, known to be on the same link, even if these interfaces are configured to appear within different subnets [2].

B. MANETs: The Missing Link

MANET interfaces are a specific class of network interfaces, which exhibit what is commonly denoted as *semi-broadcast characteristics*. This implies that otherwise neighboring routers may experience distinctly different local connectivity.

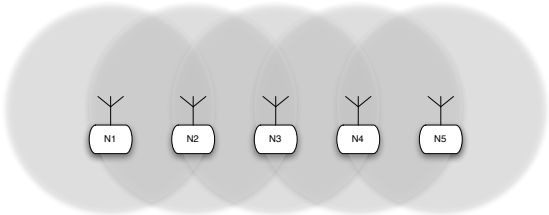


Figure 1. **MANET:** routers (N) with MANET interfaces. The light grey area indicates the **coverage area** of each MANET interface.

In figure 1, the MANET interface of N2 is able to directly communicate with the MANET interface of N1 and N3 (*i.e.*, no forwarding, TTL is not decremented, a transmission from the MANET interface of N2 is received by the MANET interfaces of both N1 and N3, including link-local broadcasts). Considering the properties listed for an IP link in section I-A, this might imply that the MANET interfaces of N1, N2 and N3 would be connected to the same link. However, the semi-broadcast nature of MANET interfaces implies that this is not so: transmissions from N1 will not reach N3 without forwarding – which entails TTL decrement, and that link-local broadcast from the MANET interface of N1 will not reach the MANET interface of N3.

As per the assumptions that connectivity between any pair of MANET interfaces may appear and disappear spontaneously and arbitrarily, any attempt at configuring interfaces "within radio range of each other" to be within the same subnet prefix would be only temporarily sound: connectivity between any such pair of MANET interfaces might disappear at any time, or another MANET interface might appear, with connectivity to one (but not all) otherwise so configured MANET interfaces.

The semi-broadcast nature of MANET interfaces therefore implies that:

- Any two MANET interfaces cannot be assumed to be connected to the same link; thus
- The IP address / prefix configuration of MANET interfaces must be such that their configuration (as per the constraint in section I-A) does not indicate that they can be assumed to be connected to the same link.

C. The Morphology of a MANET

A MANET router is a router having at least one MANET interface, *i.e.* an interface with semi-broadcast characteristics as indicated in section I-B, and otherwise retains the usual characteristics of a router. In particular, a MANET router:

- May have a prefix delegated to it;
- May delegate (part of) that prefix to other subordinate routers;
- May have networks attached to it, in particular these networks may be of any type, (including, but not limited to, other MANETs), thus respecting usual routing and addressing hierarchies; these networks may be configured by way of the prefix delegated to the MANET router.

A common case of a MANET may look as in figure 2, incidentally similar to the Internet with a "routing domain" (white cloud) and an edge (gray cloud).

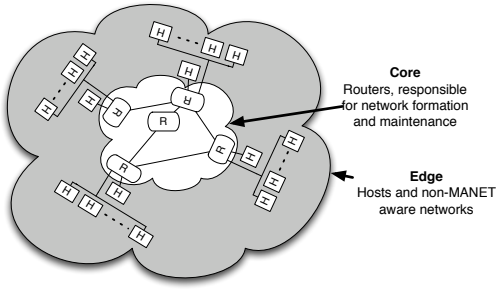


Figure 2. **MANET with attached networks:** routers (R) with MANET interfaces (white cloud), some of which with "classic IP links" and attached networks/hosts (gray cloud).

D. MANET Interface Addressing

With respect to configuration of MANET routers, the constraints regarding IP address and prefix configuration of network interfaces must be respected, also on the MANET interface(s). This entails that:

- No two MANET interfaces can be assumed to enjoy persistent connectivity to each other;
- Connectivity between any two MANET interfaces may appear and disappear at any time, spontaneously and arbitrarily; thus
- The IP address / prefix configuration of MANET interfaces must be such that their configuration (as per the constraint in section I-A) does not indicate that they can be assumed to be connected to the same link.

A translation of these properties into a set of consideration for IP address configuration of MANET interfaces is that:

- An IP address configured on a MANET interface should be unique, at least within the routing domain (within the MANET – the white cloud in figure 2); and
- No on-link subnet prefix is configured on a MANET interface.

The latter can be achieved by configuring a prefix length of /128 (for IPv6) or /32 (for IPv4) on the MANET interfaces – essentially, configuring a MANET interface to be in a "subnet, containing only itself".

E. Problem Statement

The challenges addressed in this paper are, to provide a MANET autoconfiguration mechanism, which (i) provides MANET interface addresses according to the considerations in section I-D, and (ii) enables prefix-delegation to MANET routers, such that a MANET router so desiring can configure networks and hosts, connected to it via "classic IP links" (the gray cloud in figure 2).

F. Paper Outline

The remainder of this paper is organized as follows: section II discusses autoconfiguration mechanisms, proposed in literature, for MANETs. Section III proposes a novel autoconfiguration mechanism, according to section I-E. Section IV discusses an optimization for this mechanism. Section VI presents a formal verification of the proposed mechanism, by way of model checking, and section VII presents a performance study of the mechanism. This paper is concluded in section VIII.

II. Related Work

Literature abound with MANET autoconfiguration mechanisms, both as academic publications and proposals to standardization bodies – none of which, however, at the time of this writing having been sanctioned as candidate for standardization. [3] provides a comprehensive survey of a large number of such mechanisms, classifying them according to their applicability domain (stand-alone MANETs vs. MANETs attached to an infrastructure) and capability to handle network partitioning and merger. Common for all these proposed mechanisms is, however, that they do not adhere to the considerations described in section I-D. Notably, none of the presented algorithms configures /128 or /32 prefixes to the MANET interfaces; rather, they assume the MANET to be a single subnet. Moreover, they do not consider the separation of routers and hosts, and therefore do not provide prefixes to the routers, which may then be used by attached hosts or networks to that MANET router. Some of the presented algorithms depend on specific MANET routing protocols to be in operation in the MANET and thus are not generally applicable. While autoconfiguration mechanisms from academic publications, such as [4], [5], [6], present efficient mechanisms to verify uniqueness of addresses by splitting the range of available addresses into parts, none of these mechanism adhere to the considerations described in section I-D.

The autoconfiguration mechanism, proposed in this paper, is inspired by Zerouter [7], [8], [9] and IPv6 Stateless Autoconfiguration [10], neither of which are discussed in [3] as these were conceived for use in wired "classic IP" networks and so not directly applicable to MANETs. Zerouter was an early proposal for enabling home users and small companies without dedicated network operation competencies to build arbitrarily complex and large networks without manual interaction. Different approaches were brought forward in the IETF, however none were eventually standardized. Commonly, the different approaches assume that a border router of the network receives available address space from an ISP and injects that address space into the network. That address space is available for use throughout the collection of zerouters, in order to assign addresses to the router interfaces, and to delegate prefixes to the hosts connected to these routers [8].

III. MANET Autoconfiguration Mechanism

The mechanism proposed in this paper respects the considerations given in section I-D, and provides a routing-protocol independent solution to the problem stipulated in section I-E: configuring unique IP addresses for MANET interfaces and providing unique prefixes to MANET routers. It is inspired by the prefix delegation mechanism of Zerouter (as described in section II), specifically by (i) constructing prefixes such that all addresses/prefixes within the MANET can be aggregated (*e.g.*, for injection into an external routing domain as a single entry), and (ii) by the idea of relying on a router (denoted "*initiating router*") for when a new router arrives in the network. The signaling and interface address configuration mechanism is based on IPv6 stateless autoconfiguration [10].

A. Algorithmic Overview

Each MANET router will acquire a prefix, constructed as `d:p:s::` with `d` being a prefix (possibly of size 0), common for the whole site (*e.g.*, a global prefix assigned administratively to a given site, or provided by an Internet gateway), `p` being common for all routers in this MANET, and `s` being unique to a specific router.

The task for a router, appearing in a MANET, can thus be summarized as:

- Acquiring `d` and `p`;
- Selecting `s`, unique within the MANET;
- Configuring own MANET interfaces with addresses from within `d:p:s::` (and with a prefix length of /128 or /32 as appropriate for IPv6 and IPv4, respectively).

A router appearing in a network and wishing to be configured, is denoted a *configuring* router. Through a *Prefix Solicitation* (PS) / *Prefix Advertisement* (PA) message exchange, the router learns of the already configured routers in its vicinity, and selects one as *initiator* – the router which will assist in acquiring a valid configuration. The *configuring* router extracts `d` and `p` from the PA received from the selected *initiator*, generates a *tentative prefix* `d:p:s::`[1] and, by way of a Router Solicitation (RS) message requests to its *initiator* that this *tentative prefix* be verified unique within the MANET. The *initiator* then issues and floods an RS, containing the *tentative address* of its *configuring* router, through the MANET. If the *initiator* does not (after due delay and retransmissions) receive a *Router Advertisement* (RA) indicating that the prefix is already in use, it will transmit an *Autoconfiguration Confirmation* (AC) message to the *configuring* router, confirming that the *configuring* router now "owns" `d:p:s::` and can now become a *defensive* router. If the *initiator* receives an RA indicating that the tentative prefix is already in use, it informs the *configuring* router by issuing an RA.

A *defensive* router has two tasks. First, if receiving a RS containing its own `d:p:s::`, it must respond by issuing an RA. Second, if receiving a PS, it must respond by a PA, thus accept becoming *initiator* and act as described above.

The *initiator* and *configuring* routers communicate using link-local multicast to the standardized link-local multicast address for MANET routers (`LL-MANET-Routers` [11]), with the *configurator* using the *unspecified* address as source. These two routers identify traffic destined to each other by way of Universally Unique Identifiers (UUIDs) [12], embedded in all messages exchanged between them. UUIDs are 16 octets long, and as they are exchanged in messages only between neighboring routers, they need only be locally unique. Network-wide messages (RS/RA) are "proxyed" by the *initiator*, which is already configured.

B. Formal Protocol Specification

The protocol is formally specified by way of timed automata, included in this section. The motivation for a formal specification is to prove (or disprove) certain properties, presented in section VI.

Figure 3 shows the timed automaton representing a *configuring* router. At the initial state, `INIT`, it starts the PS/PA exchange for acquiring the prefix `d:p::`. If, after several retries, no PA has been received, the router selects a random `s` and finishes configuration in the state `CONFIGURED`. If, however, it receives a PA from the *initiator* (`PA_RECVD`), it starts sending RS'es to the *initiator* (`RS_SEND`), until it receives an RA (`RA_RECVD`) in case of a conflict, or an AC otherwise (`AC_RECVD`). It then finishes configuration (`CONFIGURED`), becoming a *defensive* router.

[1] `s` being generated locally by the *configuring* router, *e.g.*, by a pseudorandom generator

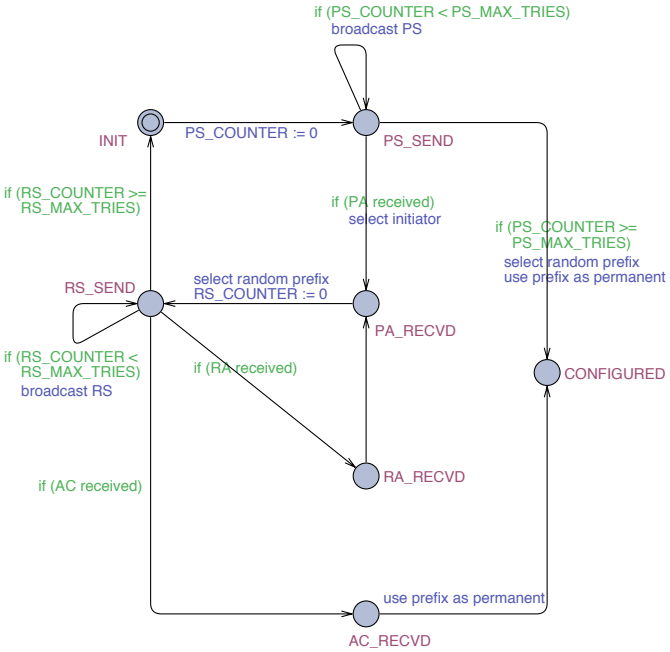


Figure 3. State machine of a *configuring* router

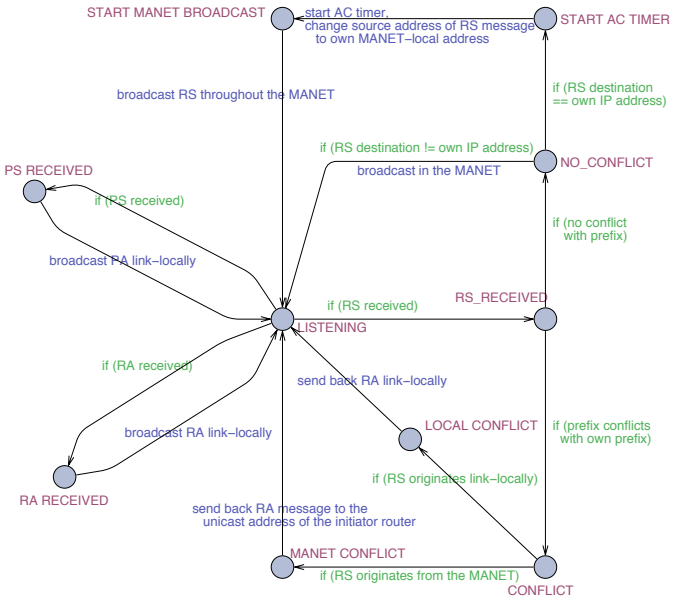


Figure 4. State machine of a *defensive* router

Figure 4 shows the timed automaton representing a *defensive* router, initiated in the state `LISTENING`. When the router receives an RS, it checks for a conflict with its own prefix (`CONFLICT`) – if one is identified, it responds by sending an RA, either via a link-local transmission if the *defensive* router is the *initiator* for the *configuring* router (`LOCAL_CONFLICT`), or through the MANET otherwise

(MANET_CONFLICT). If the tentative prefix does not conflict with its own prefix (NO_CONFLICT), the defensive router either starts the AC timer, if it is the *initiator* for the *configuring* router (START_AC_TIMER), or otherwise forwards the RS through the MANET (back to LISTENING).

IV. Proxying Extension

In this section, an optimization to the protocol proposed in section III is presented, for the purpose of increasing the efficiency of the protocol.

Rather than broadcasting RS'es through the MANET, an intermediate router can reply with an RA on behalf of a conflicting router if that intermediate router already know that there will be a conflict. This may be if the intermediate router already has a route recorded to the prefix contained in the RS (*e.g.*, as provided by a proactive routing protocol such as OLSR [13]), or by an intermediate router temporarily caching RS and RAs previously seen. This may reduce network congestion on a large scale, quantified in section VII.

V. Additional Considerations

This section addresses several issues that may occur in the protocol and proposes ways how to solve these problems.

A. Duplicate UUIDs

A question may arise as to the behavior of the protocol in case two MANET routers select the same UUID. While this might seem quite improbable, given that a UUID is 16 bytes long, the ability to generate a truly random UUID depends on the quality of the random number generator available in the router. Typically, routers use a pseudo-random generator algorithm based on a seed and some environmental settings. Thus, when two routers initiate the random generator with the same seed, they may create the same sequence of random numbers. If that is the case, duplicate UUIDs appear.

Consider the example depicted in figure 5 with two *configuring* routers and one *initiator* router.

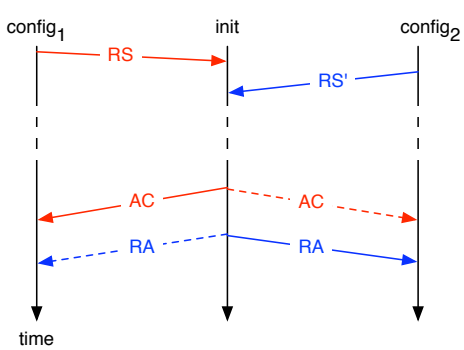


Figure 5. Two routers $config_1$ and $config_2$ having the same UUID lead to ambiguous destinations of ACs

The only case when having duplicate UUIDs interferes with the autoconfiguration protocol is when the *initiator* router sends an AC to one of its *configuring* routers, and there happen to be two *configuring* routers in radio range of the *initiator*, and both of these *configuring* routers want to configure the same *tentative prefix*. In the example, both routers $config_1$ and $config_2$ send their RS requesting the *initiator* to verify uniqueness of the same *tentative prefix*, at close to the same time. The *initiator* replies by sending an AC to $config_1$ (assuming there is no prefix conflict in the MANET) and then an RA to $config_2$, with the intent of informing $config_2$ that its *tentative prefix* already is in use. As both of the two *configuring* routers have the same UUID and have requested the same *tentative prefix*, they will both receive the AC – and will therefore both configure with that duplicate prefix. The later arriving RA will be ignored by both routers; they already are configured and thus no longer listening for such.

The objective is to detect a UUID conflict, when it occurs, and once detected avoid entering a CONFIGURED state. To that end, after a router having received an AC or an RA corresponding to its UUID, it will wait a small amount of time. If during that time the router receives a second AC or RA for the same UUID and tentative prefix, the router can deduct that a UUID conflict has occurred and take corrective action: re-seeding its random number generator and restarting the autoconfiguration process.

B. No initiator router exists

This problem occurs when two or more routers want to configure a prefix almost simultaneously when there is not yet an already configured MANET router. In figure 6 an example of that case is depicted. Both routers $config_1$ and $config_2$ start sending their PS s almost simultaneously. As there is no configured router yet, no PA replies will arrive. After PS_MAX_TRIES tries, both routers will become *initiator* routers but not being part of the same MANET (*i.e.* not having the same prefix part p and not having verified uniqueness of their prefixes).

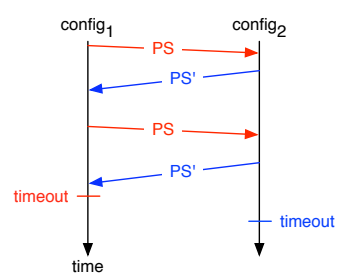


Figure 6. Problem when several routers are configured almost simultaneously without *initiator* routers

There can be two different cases as soon as the *configuring* routers from the example become *initiators* and choose the random prefix p:s:

1) If the routers choose different prefix parts p, they cannot be aggregated using CIDR. They may join the same prefix region by merging.
2) If both routers choose the same prefix part p, they are part of the same prefix region but did not verify the uniqueness of the prefix part s. They could for

example passively detect collisions (as proposed by a number of "passive" autoconfiguration algorithms in the survey [3]).

To solve this problem, *configuring* routers can listen for incoming PSs from other routers. The router with the lower UUID waits at least `AC_TIMEOUT` seconds before continuing the autoconfiguration process. Thus, the adjacent router can get configured in the meantime.

VI. Formal Validation using a Model Checker

The proposed protocol, as formally specified in section III-B, has been modeled as Timed Automaton and subjected the UPPAAL model checker. This model checker has previously and successfully been applied for other communication protocols (e.g. Zeroconf [15]).

A. Assumptions and Simplifications of the Model

The model used in the model checker has been slightly simplified in comparison to the specification and the implementation. The following assumptions have been made:

- **Simplified broadcasting**
  When the initiator router broadcast a RS, it instantly received by all other configured routers in the MANET, *i.e.*, assuming a perfect and instantaneous broadcast operation.
- **No two routers are configured at the same time**
  If two routers configure in the same time, with no *initiator* router, no RA or AC exchange can be performed and both routers would become initiator routers without having checked address uniqueness. In an initial model of the protocol, this had happened. So using the model checker, a solution for this problem has been developed (as described in section V-B).
- **Routers do not have a MANET prefix**
  Routers all have addresses of the form x where $x \in I\!N$.

B. Verification of Properties in UPPAAL

The following properties of the autoconfiguration algorithm were of interest to be proven correct: The algorithm...

- **converges:** The algorithm should terminate in finite time.
- **configures unique prefixes to all MANET interfaces**

In order to prove the correctness of the above-mentioned properties, they have to be "translated" into logic queries supported by UPPAAL (using Computation tree logic, CTL). The following queries have been verified on the model of the autoconfiguration algorithm:

- **A[] not deadlock**
  This query makes sure that in no state (A[]) a deadlock occurs. A state is a deadlock state if there are no outgoing action transitions neither from the state itself or any of its delay successors.
- **A[] forall (i : UUIDType) forall (j : UUIDType) IP[i] == IP[j] imply (i == j or IP[i] ==0)**
  This query assures that in all states (A[]), for all routers (j : UUIDType), IP addresses are either not yet configured, or not the same as on any other router in the MANET.
- **E<> forall (i : UUIDType) IP[i] != 0**
  This query checks whether there exists a state (E<>) for every router in the MANET (i : UUIDType), in which that router has a configured IP address. That means that the autoconfiguration algorithm has to facilitate that every router can successfully configure an address.

The last statement is weaker than proving that all routers *must* be configured after a finite time:

- A[] E<> forall (i : UUIDType) IP[i] != 0.

UPPAAL does not support to verification of this statement. It is currently unclear whether this behavior of UPPAAL is a deficiency of the particular model checker or generally infeasible due to the state explosion and the huge amount of necessary calculation power.

VII. Simulation

In this section, the performance of the proposed autoconfiguration mechanism is studied, by way of network simulation using NS2 and using relatively standard scenario parameters (1km^2 square, no mobility, very light traffic and a varying number of routers randomly distributed across the area, simulations averaged over 20 runs per data point, current timestamp used as seed for random number generator). Further simulation parameters are listed in table 7(a). Routers, and so autoconfiguration operations, are started consecutively on each router every second, *e.g.* the router with the ID 17 engages the autoconfiguration process 17.0s after simulation start. Each router acquires the MANET prefix either by receiving a Prefix Advertisement (PA) message or, failing that, by choosing a random prefix if it is the first router in the MANET. All routers will, for the purpose of evaluating worst-case performance, want to configure their prefix `p:s::` to be `0:1::`. Thus, from the second router appearing in a MANET, address collisions will appear. Two different versions of the autoconfiguration algorithm have been tested, the basic version presented in section III and an extended version with proxying, as described in section IV.

A. Simulation results

Both versions of the autoconfiguration protocol resulted in all routers being configured with unique prefixes, eventually. However, the number of messages exchanged and number of messages lost due to collisions differ. In the basic version of the algorithm, RS and RAs are flooded throughout the MANET, whereas in the proxy version, routers cache prefixes extracted from RSs for the duration of the simulation. Additionally, routers store a temporary reverse route back to the *initiator* when forwarding RSes, so as to enable unicast RAs when possible.

Figure 7(b) depicts the total number of transmissions in both versions for the MANET. Note that this number does not include ARP or MAC overhead. As the proposed mechanism does not rely on routing, no IP forwarding is applied. Messages used in this protocol are forwarded on an application layer, which means that they are counted as a new transmission in the figure, each time a message is forwarded or newly generated.

As can be seen, proxying significantly reduces the number of transmissions. Consequently, the drop rates, depicted in figure 7(c), are lower due to less channel contention.

VIII. Conclusion

This paper presents architectural considerations for address configuration of router interfaces and prefix delegation to routers in IP-based MANETs. Special attention is given to understanding how to configure MANET routers so as to, on the one hand, accommodate the particularities of the MANET interface type and the spontaneous and arbitrary nature of "links" in MANETs while, on the other hand, respecting the assumptions and expectations that applications present to an IP-based network. A simple set of principles are presented which, if respected, satisfies both.

An autoconfiguration protocol for MANET interface address assignment and prefix delegation to MANET routers is then presented, inspired by both Zerorouter and IPv6 Stateless Autoconfiguration. This protocol is fully distributed, and enables automatic configuration of both stand-alone MANETs, as well as MANETs connected to an infrastructure. The protocol is specified by way of a set of timed state machines, which enables subjecting it to model checking using the UPPAAL model checker – identifying that the protocol works, notably that (i) there are no deadlocks in the distributed protocol, (ii) that all configured routers ultimately ends up with distinct addresses and disjoint prefixes and (iii) that given a network with sufficient addresses available, the protocol will converge to a state where all MANET routers are properly configured.

In order to understand the performance of the protocol, a simulation study is presented, testing a performance-improving, overhead-reducing extension to the protocol. These performance results show that the protocol has a relatively low overhead, and that using a proxy extension to the base protocol can further reduce packet overhead and drop rate in the network.

References

[1] Internet Corporation for Assigned Numbers (ICANN), "Webpage," http://www.icann.org/.

[2] R. Hinden and S. Deering, "IP Version 6 Addressing Architecture," December 1995, RFC 1884, Standards Track.

[3] C. Bernardos and M. Calderon, "Survey of IP address autoconfiguration mechanisms of MANETs," November 2008, Internet Draft, work in progress, draft-bernardos-manet-autoconf-survey-04.txt.

[4] M. Thoppian and R. Prakash, "A distributed protocol for dynamic address assignment in mobile ad hoc networks," *IEEE Transactions on Mobile Computing*, pp. 4–19, 2006.

[5] A. Tayal and L. Patnaik, "An address assignment for the automatic configuration of mobile ad hoc networks," *Personal and Ubiquitous Computing*, vol. 8, no. 1, pp. 47–54, 2004.

[6] M. Mohsin and R. Prakash, "IP address assignment in mobile ad hoc networks," in *MILCOM*, vol. 2. Citeseer, 2002, pp. 856–861.

[7] J. Linton, "Automatic Router Configuration Protocol," March 2002, Internet Draft, draft-linton-arcp-00.txt.

[8] Y. Noisette and A. Williams, "A framework for Zerouter operations," February 2003, Internet Draft, draft-noisette-zerouter-frmwk-00.txt.

[9] A. White, "Zero-Configuration Subnet Prefix Allocation Using UIAP," October 2002, Internet Draft, draft-white-zeroconf-subnet-alloc-01.txt.

[10] S. Thomson, T. Narten, and T. Jinmei, "IPv6 stateless address autoconfiguration," September 2007, Standards Tracks RFC 4862.

[11] I. Chakeres, "IANA Allocations for Mobile Ad Hoc Network (MANET) Protocols," March 2009, RFC 5498, Standards Track.

[12] P. Leach, M. Meallin, and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace," July 2005, RFC 4122, Standards Track.

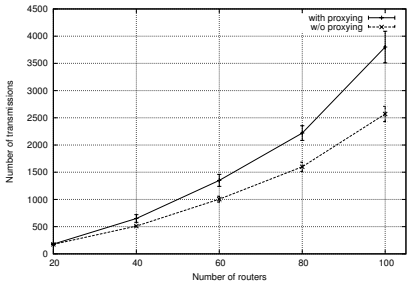
[13] T. Clausen and P. Jacquet, "Optimized link state routing protocol (OLSR)," October 2003, Experimental RFC 3626.

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

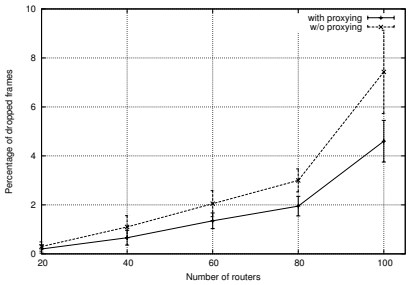
[15] B. Gebremichael, F. Vaandrager, and M. Zhang, "Analysis of the zeroconf protocol using UPPAAL," in *EMSOFT '06: Proceedings of the 6th International conference on Embedded software*, 2006.

Parameter	Value
Area	1Km x 1Km
Simulation duration	800 s
Number of routers	20 to 100)
Initial topology	Rand. uniform distrib.
Mobility pattern	none
MAC	802.11b
Wireless range	250m
Antenna	Omnidirect.
Propagation model	TwoRayGround

(a) NS2 simulator settings



(b) Number of sent IP packets of the autoconfiguration agent - Total number of transmissions (90% confidence intervals)



(c) IP Packet drop ratio: MAC frame drop ratio (90% confidence intervals)

Figure 7. NS2 Simulations