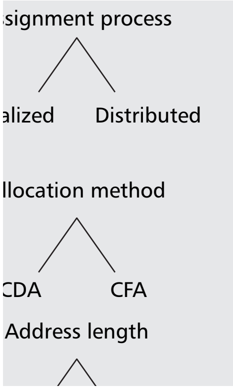
ADDRESS AUTOCONFIGURATION IN WIRELESS AD HOC NETWORKS: PROTOCOLS AND TECHNIQUES

NOVI INEKE CEMPAKA WANGI, R. VENKATESHA PRASAD, MARTIN JACOBSSON, AND IGNAS NIEMEGEERS, DELFT UNIVERSITY OF TECHNOLOGY



The authors look in depth at the problem of addressing in wireless ad hoc networks and the currently available techniques and protocols for both IPv4 and IPv6.

ABSTRACT

With the advent of smaller devices having higher computational capacity and wireless communication capabilities, the world is becoming completely networked. Although, the mobile nature of these devices provides ubiquitous services, it also poses many challenges. In this article, we look in depth at the problem of addressing in wireless ad hoc networks and the currently available techniques and protocols for both IPv4 and IPv6. We present an exhaustive study of the literature and summarize the features of each technique. We believe that this analysis will be helpful for network and application designers, as well as for researchers.

INTRODUCTION

Like a house number, postal code, and street address, each communicating entity on the Internet requires a unique network address with which it can be reached. With the advent of wireless technology and the seamless integration of wireless networks with the Internet, addressing techniques are drawing more attention from the communication research community. Unlike in wired networks, addresses of mobile or wireless devices change from time to time. Consequently, the wireless nodes, moving from one place to another have different points of presence in the Internet and belong to different subnets. An autoconfiguration method such as Dynamic Host Configuration Protocol (DHCP) [1] is commonly used to assign temporary IP addresses to such nodes. Unfortunately, this type of solution is not suitable for mobile and wireless devices. Another area where DHCP and other solutions for fixed networks, such as Internet Engineering Task Force (IETF) Zeroconf [2], do not provide a good solution is mobile ad hoc networks (MANETs). Such networks consist of mobile devices that communicate with each other with wireless communication technologies in an ad hoc fashion. The nodes also act as routers and relay packets that cannot reach the destination in one hop. This makes the network multihop, and this multihop characteristic of a MANET makes existing addressing solutions for fixed networks inadequate. Those protocols are designed for a single LAN topology, with the assumption that every node can reach all the others by link layer broadcasts or multicasts. Such assumptions are not true for most MANETs. All nodes are not guaranteed to be reachable through link local broadcasting. Different protocols are therefore required for MANETs.

The major requirement of ad hoc addressing schemes is ensuring the uniqueness of node addresses so that no ambiguity appears when they try to communicate. This is not as trivial as it seems, especially because of the dynamic topology of ad hoc networks. A MANET cloud can be split into several parts, and several MANET clouds can merge into one. Tens to thousands of nodes coexisting in a single network may participate concurrently in the configuration process. Moreover, the wireless nature, such as limited bandwidth, power, and high error rate makes the problem even more challenging. Besides handling a dynamic topology, the protocols must take into account scalability, robustness, and effectiveness. Finally, in IPv6, a protocol is expected to tackle not only the local addressing, but also the global addressing.

Since 1998, several address autoconfiguration protocols for IPv4 and IPv6 have emerged to attempt to meet these requirements. We observe that each protocol has tried to achieve a level of optimization for a particular aspect. The main objective of this article is to provide a comparison of the protocols based on their features. We select representatives of each popular method and describe the differences among them. A survey presented in [3] compared only early ad hoc protocols. Others concentrated on specific areas, namely, in the ambit of IPv4 [4] or based on a particular aspect, such as in [5]. Our work attempts to compare all the techniques of IPv4, as well as of IPv6; and highlight their merits, applicability, and usage for different scenarios to help analyze the general design of addressing protocols. We also present the important characteristics of each protocol that can serve as the

1536-1284/08/$25.00 © 2008 IEEE

	IPv4 ZeroConf [2]	IPv6 SAA [6]
Initial address	Derived from MAC	Unspecified address 0:0:0:0:0:0:0:0
Network identifier	165.254/16	FE80:::: prefix obtained from RA
Interface identifier	Randomly chosen	Derived from IEEE MAC address or randomly chosen
DAD	ARP requests and replies	NS and NA messages
Method to perform DAD	Link local broadcast	Link local multicast
Number of autoconfigured addresses per interface	Only one	Possibly more than one
Address state maintenance	Stateless (link local only) and stateful (global)	Stateless (link local and global) and stateful (global)

Table 1. *Comparison between IPv4 and IPv6 autoconfiguration protocols for wired networks.*

basis for future design and for the selection of suitable ones for particular applications.

This article is organized as follows. First, we briefly discuss protocols for wired networks to understand some of the basic autoconfiguration concepts. Then, we present the comparison of protocols for wireless networks. Before analyzing general design targets, we begin by comparing typical solutions for IPv4 and IPv6-based protocols. Finally, we present a consolidated view of the techniques and offer a conclusion.

Address Autoconfiguration Protocols in Wired Networks

The ZeroConf protocol [2] is designed for dynamic IPv4 local configuration to support simple plug-and-play networking scenarios. Imagine a group of people meeting in a room; without prior configuration by a network administrator, their devices can still obtain valid IP addresses, directly communicate with each other, and terminate the connection at the end. To obtain an address, a node performs the following steps:

- Selects a tentative random address from the IPv4 link local address range (164.254/16).
- Performs Duplicate Address Detection (DAD) by broadcasting Address Resolution Protocol (ARP) messages and waits for replies for a certain amount of time.
- If no duplicates are detected (no ARP replies), it claims the address; otherwise it repeats the process with a new tentative address.

An IPv6 solution for autoconfiguration, published in [6], basically adopts similar protocols as that of IPv4. The 128-bit IP address is separated into a network prefix and an interface identifier (I-ID). For local addressing, the network prefix is set to FE80::::. The I-ID can be derived in two ways. If its network interface has an embedded IEEE extended unique identifier (EUI), the I-ID is computed from that identifier. Otherwise, it is generated through other means, for example, using a random number generator. Similar to the IPv4 version, the next step is performing the DAD using neighbor solicitation (NS) and neighbor advertisement (NA) messages [7] that are multicasted to all nodes on that link. Unlike the IPv4 protocol, after successfully claiming a local address, the process continues to obtain a global address. To obtain information about the global prefix, the node may proactively send router solicitation (RS) messages or listen to periodic router advertisements (RA). After receiving an RA, a node forms the global IPv6 address by combining the announced prefix and the interface identifier, just as it did for the link local address. It should be noted that IPv6 may allow an interface to have more than one global address. If two routers exist on the same network and advertise different prefixes with RAs, the receiving node automatically receives both RAs to allocate different addresses on the same interface. The node may then select which prefix and which router to use.

This type of self-configured address is called *stateless* address configuration because the address status is not managed by other nodes except itself. In contrast, a *stateful* address involves central assistance in the configuration process as in DHCP. The IPv4 stateless mechanism can be used to configure only link local addresses, whereas in IPv6, both local and global addresses can be configured. A summary of the comparison between the IPv4 Zero Conf and the IPv6 Stateless Address Autoconfiguration (SAA) protocol appears in Table 1.

Address Autoconfiguration Protocols in Wireless Ad Hoc Networks

The pioneer work of Perkins *et al.* on ad hoc address autoconfiguration [8] used a very simple method with similar concepts to the ones used for wired protocols. A newly enabled node configures its link local address and performs DAD with several retries until it succeeds and claims the use of the address. For IPv4, address request (AREQ) and address reply (AREP) messages are employed on an ad hoc routing protocol to find a route to the address being verified instead of ARP messages. For IPv6, some modifications to NS and NA were performed to fit the ad hoc

Features	IPv4-based protocols	IPv6-based protocols
Network identifier	Fixed (165.254/16, private address, or non IP)	Dynamic (FE80::: or leader/gateway prefix)
Assignment process	Centralized/distributed	Mostly distributed
Subnetwork	No	Yes
Leader main functionality	Address maintenance	Gateway
Handling network merging and partitioning	Mostly yes	Mostly no
Type of ad hoc network	Mostly standalone	Mostly hybrid
Addressing type	Local	Local and global
Internet access	Separate protocols	Mostly integrated

■ **Table 2.** *Comparison of typical designs between IPv4- and IPv6-based protocols for wireless networks.*

environment. Still, this simple method does not solve some important issues of ad hoc protocols, such as handling network partitioning and merging.

Comparison between IPv4 and IPv6-based Protocols

The IPv4 and IPv6 addressing presented previously were designed for wired networks, and we notice that the comparison of IPv4 and IPv6 addressing applied to ad hoc networks is similar to the wired networks. For example, network prefixes used in an IPv4 auto address configuration are mostly constant, whereas in IPv6, they could be dynamic and depend on available prefixes. Further, most IPv6 schemes use the mechanism of IP SAA [6] to configure interface identifiers, thus distributed address assignment is adopted by almost all schemes. Whereas most proposed IPv6 schemes consider the hybrid case, that is, the ad hoc network also being connected to the Internet via gateways; most IPv4 schemes only consider the standalone case, without any window to the external network. Therefore, only local addresses are assigned in IPv4 autoconfiguration, with the assumption that global addresses can be provided through solutions such as network address translator (NAT) or other means, which are beyond the scope of this article. Table 2 summarizes the differences between IPv4 and IPv6-based protocols.

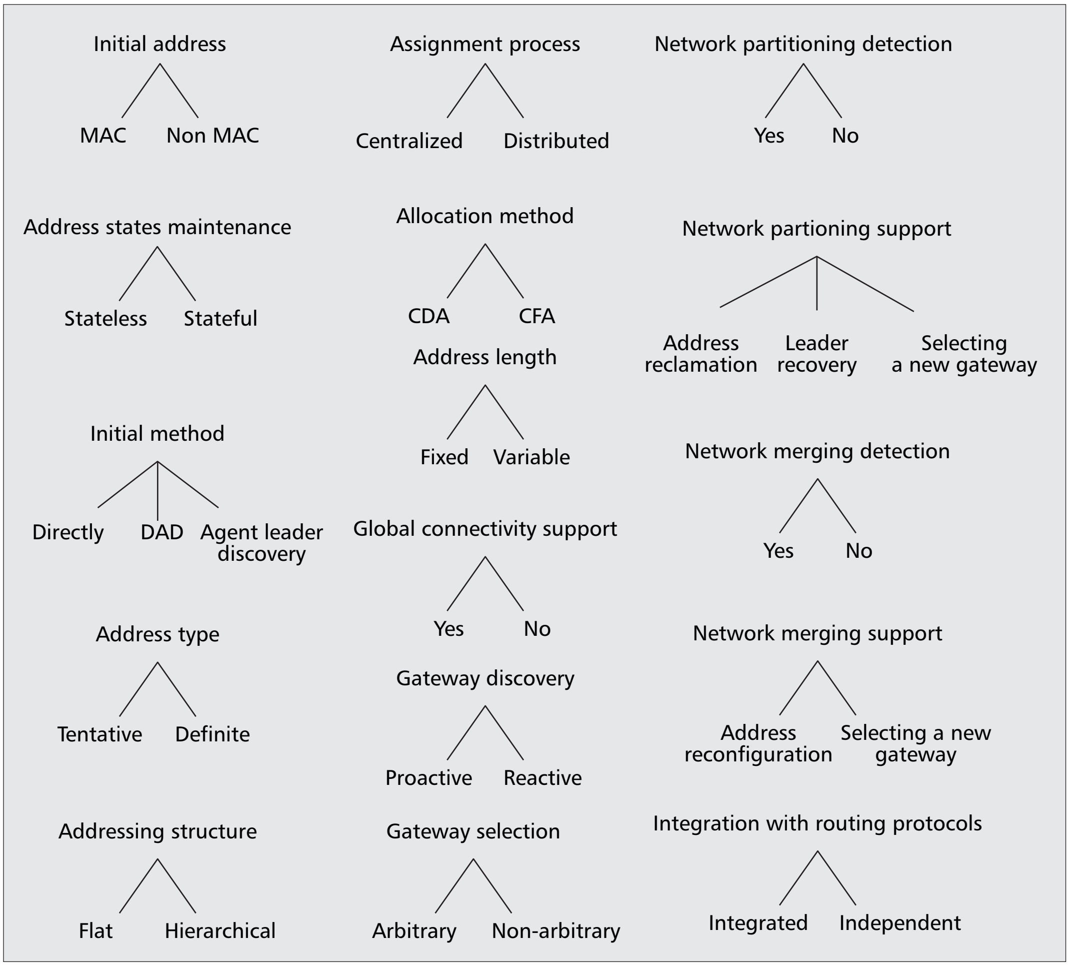
General Design

To ensure the properly working address autoconfiguration, several steps must be accomplished by the protocol. In this section, we analyze different ways of performing these steps, as well as discuss other supporting features. For a concise presentation, all of them are listed together with their subdivisions in Fig. 1. Table 3 summarizes them for a comparison in chronological order.

Initial Address — To obtain an address, a new node should have an initial address as its temporary identity. This can be classified either as a *medium access control* (MAC) or *non-MAC* address type. A non-MAC address is usually chosen randomly from a set of temporary addresses, such as 0-2047 of the 169.254/16 in IPv4 or fec0:0:0:ffff::/96 in IPv6. Some IPv6-based protocols allow new nodes to use the configured address directly, such as in [9-11]; thus, an initial temporary address is not required. Whereas in methods proposed by Jelger *et al.* [9] and Ruffino *et al.* [10] a new node configures an address using a mechanism similar to IP SAA [6], in passive autoconfiguration for mobile ad hoc networks (PACMAN) in [11], the new node configures the address using a probabilistic algorithm. In addition to the unspecified address used in the wired Neighbor Discovery Protocols (NDP) [7], Weniger *et al.* [12] use random source-ID (RS-ID) as the initial address.

Maintaining Address State — As explained previously, a configured address can be *stateless* or *stateful*, depending on who maintains the address. In ad hoc networks, even though dedicated servers are not always available or reachable, a stateful address still can be configured either in a centralized or distributed way. Furthermore, not always a full list of addresses is to be maintained; in fact, some protocols propose lightweight solutions by maintaining only the current highest value of the address [12, 13] or the address of logical neighbors [14].

Initial Method — After configuring an address, nodes in IPv6-based protocols, such as in [9-11], use the address *directly* to send a packet without necessarily verifying the uniqueness of the configured address. The authors believe that the probability of conflicts during the configuration process is extremely low, and the impact is minimal within a limited area. In addition, a passive detection is employed to resolve future possible conflicts (see "Allocation Method"). However, most protocols do not allow new nodes to directly claim a local address. Instead, by using an initial address, a newly joined node may attempt a local IP address by performing *DAD* or *agent discovery*. Stateless addressing requires the new node to select a tentative address and perform DAD by



■ **Figure 1.** *Taxonomy of compared mechanisms and features.*

A definite address requires no verification by other nodes in the assignment process. In schemes where the current address list is well maintained, the agent or leader easily can assign a non-occupied address to a new node.

broadcasting a request to all the nodes. To perform DAD, protocols for IPv4 use AREQ and AREP messages, whereas protocols for IPv6 use modified NS and NA messages. Stateful addressing usually requires a new node to find an agent or a leader that controls address configuration, either in a proactive or reactive way.

Choosing a New Address — A node may obtain an address in the form of a *tentative* or *definite* address. A tentative address, usually randomly chosen by a new node, is an address whose uniqueness still must be verified using a DAD mechanism. As mentioned in the previous section, protocols such as in [9–11] do not execute DAD, even when a tentative one is used.

A definite address requires no verification by other nodes in the assignment process. In schemes where the current address list is well maintained, the agent or leader easily can assign a non-occupied address to a new node. In Patchipulusu [14], a new node that requires an address is assigned the next address in the sequence. Most protocols that use address space allocation divide an initial address pool into non-overlapping spaces that is allocated by splitting the available spaces [15, 16]. Prophet [17] also allocates a definite address for a new node through a computational method, that is, a sequence of addresses is assigned by an agent as a result of computing a function.

Addressing Structure — The protocols either have a *flat* or *hierarchical* addressing structure. Protocols that use flat addressing do not require any server assistance. Protocols that use a hierarchical structure often are called leader-based protocols, where one or more leaders appear in the network. The task of the leaders is defined differently in each protocol, such as *assigner* — assigning an address to a new node; *maintainer* — maintaining the state of addresses, *network identifiers* or a combination of these. Generally, the leader is elected, either through distributed communication such as Belding-Royer *et al.* [25] and Weniger *et al.* [17], or the leader functionality is distributed as in Patchipulusu [14] and Mahony *et al.* [19]. Whereas almost all of the hierarchical protocols for IPv4 allow only the presence of one leader in each network; according to some protocols for IPv6, multiple leaders can be present concurrently in a network. In such protocols, a network is divided into several subnets, and the leader of each subnet supplies a network prefix using similar mechanisms, such as RA messages in IPv6 SAA [6]. In protocols designed for hybrid networks, leaders also behave as gateways, providing Internet connectivity for the other ad hoc nodes. Further, the organization of IPv6 hierarchical addressing can be *non-structured* or *structured*. In non-structured networks, there are no boundaries between leaders and their descendants in sub networks. Nodes

Features, properties, techniques, etc.	Perkins *et al.* [8]	Boleng [13]	Patchipulusu [14]	Nesargi *et al.* [24] (MANET-Conf)	Mohsin *et al.* [15] (Buddy)	Weniger *et al.* [12]	Mutka*et al.* [17] (Prophet)	Belding-Royer *et al.* [18]
IP family	IPv4, IPv6	Non IP	IPv4	IPv4	IPv4, IPv6	IPv6	IPv4	IPv4
Initial address	Non-MAC	MAC	MAC	Non-MAC	MAC	Unspecified address and RS-ID	Non-MAC	Non-MAC
Address state maintenance	Stateless	Stateful, distributed	Stateful, centralized	Stateful, distributed	Stateful, distributed	Stateless	Stateless	Stateful, centralized
Maintained state	Not Applicable (NA)	Highest value and address length	Highest address value	List of occupied addresses	List of address spaces	NA	NA	List of occupied addresses
Initial method	DAD	Agent discovery (proactive)	Leader discovery	Agent discovery (proactive)	Agent discovery (proactive)	DAD	Agent discovery (proactive)	DAD
Address type	Tentative	Tentative	Definite	Tentative	Definite	Tentative	Definite	Tentative
Addressing structure	Flat	Hierarchical, single leader	Hierarchical, single leader	Flat	Flat	Hierarchical, physically structured	Flat	Flat
Assignment process	Distributed	Centralized	Centralized	Distributed	Distributed	Distributed	Distributed	Distributed
Allocation method	CDA	CDA	CFA	CDA	CFA	CDA[5]	CFA	CDA
Address length	Fixed	Variable	Fixed	Fixed	Fixed	Fixed	Fixed	Fixed
Global connectivity support	No	No	No	No	No	No	No	No
Gateway discovery	NA	NA	NA	NA	NA	NA	NA	NA
Gateway selection	NA	NA	NA	NA	NA	NA	NA	NA
Partitioning detection	No	Yes, using PID	Yes, using PID	Yes, using ID	Yes, using PID	Yes, using PID	Yes, using PID	Yes, using PID
Partitioning support	NA	Address reclamation	Leader recovery	Address reclamation	Address reclamation	Leader recovery	No[6]	Leader recovery
Merging detection	No	Yes, using PID	Yes, using PID	Yes, using PID	Yes, using PID	Yes, using PID	Yes, using PID	Yes, using PID
Merging support	NA	Address reconfiguration (all nodes in the partition with the least number of nodes)	Address reconfiguration (all nodes in the partition with the least number of nodes)	Address reconfiguration (only nodes with duplicated addresses)	Address reconfiguration (nodes with larger address blocks)	Address reconfiguration (all nodes in one partition)	Address reconfiguration (all nodes in the partition with a smaller PID)	Address reconfiguration (only nodes with duplicated addresses)
Protocol overhead[11]	Assignment $O(N^2)$	Maintenance $O(N^2)$	Maintenance $O(N)$	Assignment $O(N^2)$	Maintenance $O(N^2)$	Assignment $O(N^2)$	Maintenance $O(N)$	Assignment $O(N^2)$
Integration with routing protocol (RP)	Independent (optimization with reactive RP)	Independent	Independent	Independent (optimization with proactive RP)	Independent	Independent (optimization with hierarchical RP)	Independent	Integrated with proactive and reactive RP
Scalability	Small	Small	Medium/large	Small	Small	Large	Large	Small

Table 3 is continued on next page...

that use prefixes of leaders can be placed anywhere in the network. In structured networks, the subnetworks are clustered physically or logically. Physical clustering uses physical boundaries, such as hop counts, as proposed in [12]. In logical clustering, nodes in the same cluster create a logical tree that places a gateway as the root and others as leaves, as in [9]. Figure 2 shows different IPv6 hierarchical organizations. Squares and diamonds represent leaders and descendants respectively. A, B, and C represent network prefixes identified by their respective leaders. Nodes using the same prefix indicate that they belong to the same cluster.

Assignment Process — The assignment process describes how a new node obtains an address. This can be executed in a *centralized* or *distributed* manner. Most hierarchical addressing approaches use centralized assignment, whereas flat addressing approaches use distributed assignment. However, the existence of a leader does not always imply centralized assignment (see Table 3). In Belding-Royer *et al.* [18], the leader is intended for maintaining the address states, not assigning addresses. On the other hand, in Prophet [17], even though the assignment process is distributed, addresses that will be assigned to nodes that join later depend strongly on the initiator, that is, on the seed value that the initiator chooses in the beginning.

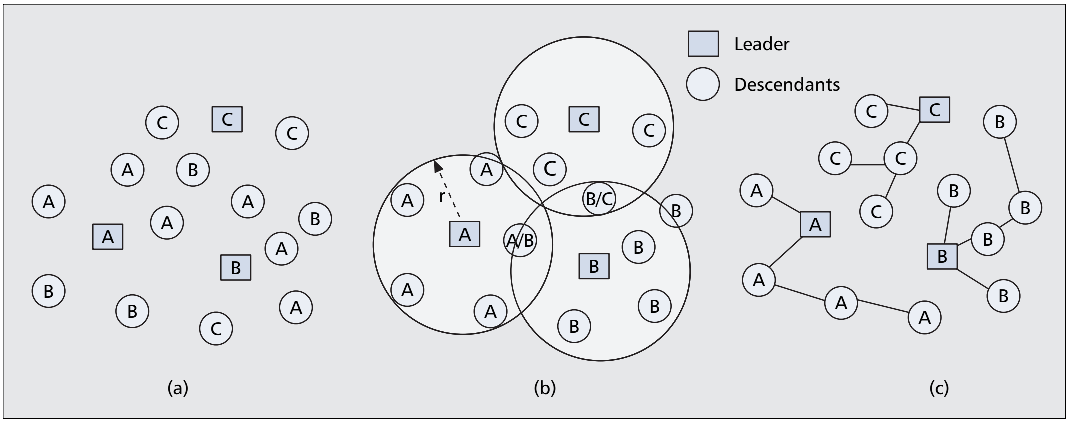
Allocation Method — The protocols accomplish address allocation using either *conflict detection allocation* (CDA) or *conflict-free allocation* (CFA). The CDA method is based on select-

Features, properties, techniques, etc.	Mahony *et al.* [19]	Park *et al.* [20]	Hu *et al.* [16] (ZAL)	Bacelli*et al.* [26]	Jelger *et al.* [9]	Weniger [11] (PACMAN)	Jeong [23]	Ruffino *et al.* [10]
IP family	IPv4	IPv6	IPv4	IPv6	IPv6	IPv4, IPv6	IPv4, IPv6	IPv6
Initial address	Non-MAC	MAC	MAC	MAC	MAC[1]	Non-MAC[1]	Non-MAC	MAC/ Non-MAC[1]
Address state maintenance	Stateful, centralized	Stateless (local-scope)/Stateful, Centralized (network-scope)	Stateless	Stateful, distributed	Stateful, distributed	Stateful, distributed	Stateless	Stateless[2]
Maintained state	List of occupied addresses	List of occupied addresses[3]	NA	List of address sequences	List of logical neighbor addresses	NA	NA	Announced Prefixes from GAs
Initial method	Agent discovery (proactive)	DAD	Agent discovery (proactive)	Agent discovery (reactive)	Directly	Directly	DAD	Directly
Address type	Definite	Tentative	Definite	Tentative	Tentative	Tentative	Tentative	Tentative
Addressing structure	Hierarchical, single leader	Hierarchical, physically structured	Flat	Flat	Hierarchical, logically structured	Flat (IPv4) and Hierarchical non structured (IPv6)	Flat (IPv4) and Hierarchical non structured (IPv6)	Hierarchical non structured
Assignment process	Centralized	Distributed	Distributed	Distributed	Distributed	Distributed	Distributed	Distributed
Allocation method	CFA	CDA/CFA Strong DAD (local addresses), CFA (global addresses)[5]	CFA	CDA	CDA Passive DAD (optional)	CDA Passive DAD	CDA Strong DAD for local and global addresses	CDA[7]
Address length	Fixed	Fixed	Fixed	Fixed	Fixed	Fixed, variable (optional)	Fixed	Fixed
Global connectivity support	Yes, but no further explanation	Yes	No	No[8]	Yes	Yes	Yes	Yes
Gateway discovery	NA	Proactive and reactive	NA	NA	Proactive, with restrictive flooding	Reactive	Proactive and reactive	Proactive
Gateway selection	NA	Arbitrary	NA	NA	Non-arbitrary (based on distance or network stability)	Arbitrary	Arbitrary	Non-arbitrary (based on routing metrics)
Partitioning detection	Yes, using PID	No	Yes, using PID	No	Yes, using GAs	No	No	Yes, using GAs
Partitioning support	Leader recovery	NA	[9]	NA	Selecting a new gateway	NA	NA	Selecting a new gateway
Merging detection	Yes, using PID	No	Yes, using PID	No	Yes, using GAs	Yes (passive DAD)	Yes (weak DAD)	Yes, using GAs
Merging support	Address Reconfig. (Only nodes with duplicated addresses)	NA	Address reconfig. (the partition with the least number of nodes)	NA	Selecting a new gateway[10]	Address reconfig. (only nodes with duplicated addresses)	Address reconfig. (only nodes with duplicated addresses)	Selecting a new gateway[10]
Protocol overhead[11]	(leader) Maintenance $O(N^2)$ or $O(N)$	Assignment $O(N^2)$	Maintenance C	Assignment $O(N)$	Maintenance $O(N)$	Maintenance C	Assignment $O(N^2)$	Gateway discovery $O(N^2)$
Integration with Routing Protocol (RP)	Independent	Independent	Independent	Integrated with OLSR	Independent	Integrated with proactive and reactive RP	Independent	Integrated with OLSR
Scalability	Small/medium	Large	Medium/large	Medium/large	Large	Large	Small/medium	Small/med.

Notes:
[1] A new node directly joins the network using a configured address and address conflicts are monitored reactively. [2] The configured address is stateless but each node maintains a list of prefixes from GAs to facilitate gateway selection mechanism. [3] Each gateway maintains a list of occupied addresses in its subnetwork. [4] Each member of a logical tree maintains a table containing all addresses that are used within the tree. [5] The local-scope DAD is performed individually by each node within a sub-network; the network-scope DAD is performed by leaders of subnetwork. [6] Address reclamation is not necessary since same addresses will be generated by the sequence after some intervals. [7] The author suggested using reactive methods to detect address conflicts, yet did not specify any particular mechanism. [8] The protocol only considered the local addressing; the author suggested using a modification of DHCP proxy for global address configuration. [9] When detecting a partitioning, nodes keep their current addresses to anticipate remerging events. [10] Nodes might be required to change their prefixes each time new gateways resulting from merging appear. [11] We consider only the foremost protocol overhead.

■ **Table 3.** *A comparison of different protocols and techniques (named after the authors who proposed them)*

Almost all protocols for IPv6 consider ad hoc networks as hybrid networks and thus, providing connectivity to the Internet is one of the important issues in their designs.



■ **Figure 2.** *Hierarchical organization of IPv6-based protocols: a) nonstrucured; b) physically structured; c) logically structured.*

ing an address from a pool of available addresses and then performing DAD. In IPv4, DAD is executed only on one level to verify the uniqueness of a link local address. However, the global address configuration in IPv6-based protocols requires DAD to be executed on two levels. The first local DAD is intended to detect conflicts with the local addresses. The second, network-wide or global DAD, is executed after a node obtains a prefix and configures the global address to test the validity of the configured address. Whereas some protocols require the nodes to perform the second level DAD individually, in Weniger *et al.* [12] and Park *et al.* [20] leaders of the subnetwork cooperate to complete this task. DAD implemented by most CDA protocols is often called Strong DAD, where DAD is ultimately part of the initial configuration process. Alternatively, a routing protocol modification, Weak DAD, is proposed in [21], by which the messages can still be delivered correctly even when duplicated addresses occur. Passive DAD [22] used by Jelger *et al.* [9] and Weniger [11] (PACMAN) is another form of DAD that is performed by inspecting incoming packets to detect address conflicts. Nodes that detect conflicts must notify relevant nodes, namely, nodes currently using duplicated addresses. In both protocols, DAD is not included in the configuration process. Jeong [23] implements both Strong and Weak DAD. Strong DAD is used in the initial allocation process, and Weak DAD is used to detect address conflicts during network merging.

In contrast to CDA, no duplicate detection is performed by the CFA method. The uniqueness of the allocated address can be assured without any cross check. For example, in Mahony *et al.* [19], a non-duplicated address can be determined by the leader because it maintains a list of currently used addresses. In Prophet [17], the duplicated address can be detected a priori and marked by the initiator. Splitting the address space is also a technique used in CFA. In Buddy (by Mohsin *et al.* [15]) and zero-maintenance address allocation (ZAL) (by Hu *et al.* [16]), each new node is assigned half of the address space that belongs to the agent.

Address Length — Most of the protocols use a *fixed* address length. Usually, it is 32 or 128 bits long to be compatible with the IP architecture. The major drawback of having such a long address is the excessive overhead caused by flooding or periodic signaling adopted by the addressing protocols. Therefore, an address configuration scheme using a *variable* length is proposed in Boleng [13]. Although starting with an extremely short length of four bits that provides 16 unique addresses, it can extend the length up to 64 bits long, thus providing more than 10^{19} addresses. However, each time an address space expansion is required, the address length will be incremented by four bits, and all the existing nodes must repeat the configuration process. Motivated by this work, in PACMAN, Weniger [11] proposed an autoconfiguration mechanism with an optional compression technique. A fixed length address is encoded in a variable length MAC address by the sender and decoded back to the original address as soon as it arrives at the receiver. However, to compute this properly, mobile nodes must have the ability to perform several different coding schemes.

Global Connectivity and Gateway Discovery — Almost all protocols for IPv6 consider ad hoc networks as hybrid networks and thus, providing connectivity to the Internet is one of the important issues in their designs. A principle that is similar to IPv6 SAA [6] is adopted here: the network prefixes of nodes acting as gateways are used to configure global addresses for other ad hoc nodes. We note two interesting aspects, namely, *gateway discovery* and *selection*, when multiple gateways are present.

Like IP SAA [6], the schemes use a reactive or proactive approach to discover gateways. In the reactive approach, the newly arrived node broadcasts gateway solicitations (GSs) to acquire a network prefix on demand. In the proactive approach, gateway nodes periodically flood the network with gateway advertisements (GAs) containing prefix information. Both mechanisms are used by Park *et al.* [20] and Jeong [23]. To prevent a broadcast storm of GAs, a restricted flooding mechanism is proposed by Jelger *et al.* [9]. A node forwards GAs only from the gateway it is currently using and discards GAs from other gateways. This results in logical arrangements of nodes that use the same prefixes.

When multiple gateways are present, basically each interface of a node can be assigned with different prefixes. However, because most ad hoc routing protocols do not consider having multiple addresses for one interface, only one global address can be assigned to each interface. Obviously, selecting the "best" prefix among the available ones will improve the routing performance. Yet, we notice that all protocols for IPv6 (that are presented here) do not deal with this in their designs. Using *arbitrary* selection, nodes are free to select their routes to the Internet. Jelger *et al.* [9] and Ruffino *et al.* [10] are two exceptions that propose non-arbitrary selection based on routing metrics, whose information is embedded in the GAs. In Jelger *et al.* [9], nodes must choose the gateway with the shortest distance. In Ruffino *et al.* [10], nodes cache GAs, compare metrics of each pair of gateways, and sort them before finally making a choice. Considering the dynamically changing ad hoc topology, the main drawback of this approach is that a node might be required to change its prefixes frequently, possibly resulting in interruptions of the active session. To solve this problem, Jelger *et al.* [9] proposed a method that takes into account the network stability, and Ruffino *et al.* [10] proposed an alternative to use a default gateway method.

Network Partitioning: Detection and Support — Network partitioning occurs when a node leaves the original network. If a group of nodes leaves the network gracefully, then the nodes may notify others about their departure. Hence, the addresses can be reused in the original network by other nodes that join later. In many protocols, the node must send "bye" messages to release its address; either only to its neighbors or to all the nodes. However, when the node leaves abruptly, most protocols use the partition identifier (PID), which is periodically announced by the leader (in a hierarchical structure) or checked between neighboring nodes (in a flat structure) as a means to detect such events.

The effect of network partitioning is a bit different in the cases of protocols for IPv4 and IPv6. In IPv4, a node that disappears from a network will take away its address so that it cannot be used by others. Therefore, the effect is considered significant only to protocols that have small address spaces; or they are sensitive to address leakage, such as Buddy [15]. For hierarchical protocols, a different problem appears in the case of a departing leader. When the leader departs without informing other nodes, the task of a leader will be affected. In IPv6, partitioning may cause more serious problems to global addressing. Suddenly, if a MANET breaks into two separate parts, some nodes in each part might be required to change their network prefixes when current gateways are unreachable. Nevertheless, no protocol has so far solved this problem.

When partitioning is detected, Buddy [15] executes an *address reclamation* procedure by automatically merging the address block of departing neighbors with itself. Most CDA protocols do not implement this procedure because it indirectly happens when a new node performs DAD. Despite this, Boleng [13] and Nesargi *et al.* [24] (MANETConf) still perform reclamation by applying caching and a clean-up procedure, respectively. Hierarchical protocols should perform *leader recovery* in the case of the departure of a leader. If the leader is elected in a distributed fashion, as in Patchipulusu [14] and Mahony *et al.* [19], the last joining node (indicated by a sequence number) automatically becomes the new leader. In Belding-Royer *et al.* [18], a backup leader will become the main leader. However, the election must be repeated when both leave the original network simultaneously.

Although most protocols for IPv6 do not specify methods to handle partitioning, a typical solution is that nodes should wait for a certain period of time to discover whether the connection to the current gateway is lost and then discover another gateway. To reduce the latency that arises from this recovery process, Ruffino *et al.* [10] proposed a mechanism where each node caches information about each GA and ranks them according to routing metrics. The prefix of the gateway that has the highest rank is selected, and the prefix of the gateway with the second highest rank automatically is used when the current one vanishes.

Network Merging: Detection and Support — Conflicting addresses is a major problem when networks merge. That is, some nodes in each network may currently be using the same address. When the autoconfiguration protocol considers only link local addressing, address conflicts can occur in the protocols for both IPv4 and IPv6. Eventually, this also will be the case of global IPv6 addresses. If the same network prefixes are used in the different fragments, address collisions can occur whenever those fragments merge. However, many protocols assume that each gateway will use a topologically different and correct network prefix through manual configuration or dynamic set-up mechanisms as described in [25]. Thus, the uniqueness of the address can still be guaranteed in the case of network merging.

Similar to partitioning, IPv4-based protocols generally use a PID to detect merging. However, most IPv6-based protocols do not have a mechanism for the detection of merging, even for local address configuration. Most likely, the reason is that a 64-bit interface identifier is considered to be long enough, and DAD performed previously is adequate to guarantee conflict-free addresses. Nevertheless, PACMAN [11] uses Passive DAD [22] to monitor incoming traffic continuously. Similarly, Jeong [23] uses the combination of IP address and a key proposed by Weak DAD [21] to derive hints about address conflicts.

When two networks merge, a simple solution adopted by most of the protocols for IPv4 is to have *all nodes of one partition* release their old addresses and reconfigure, while the addresses of the nodes from another partition remain unchanged. The choice of the partition that changes addresses is based on a particular criterion, for example, the partition with the least number of nodes, smaller PID, and so on. A more elegant solution is proposed in Belding-

Network partitioning occurs when a node leaves the original network. If a group of nodes leaves the network gracefully, then the nodes may notify others about their departure. Hence, the addresses can be reused in the original network by other nodes that join later.

The total number of nodes in the ad hoc network determines the bandwidth consumed by autoconfiguration protocols. We consider this aspect to compare the different protocols for their suitability for large-scale deployment.

Royer *et al.* [18] and Mahony *et al.* [19]. After detecting network merging, leaders of each partition communicate to compare the address lists. As a result, *only nodes with duplicate addresses* must change their addresses and restart the configuration process. This solution can be achieved only if the leaders maintain the full list of current addresses. Jeong [23] and IPv6-based protocols like PACMAN [11] treat merging differently. If intermediate nodes discover that two nodes are using the same address, they alert one of them to change its address. As stated earlier, due to the uniqueness of prefixes used by each gateway, the presence of new gateways resulting from merging will not introduce any possibility of a collision for global address configuration. However, protocols such as those proposed by Jelger *et al.* [9] and Ruffino *et al.* [10], which stipulate nodes to select the best gateway, require nodes of merged networks to include new gateways caused by merging in the selection process.

Protocol Overhead — Protocol overhead is caused by processes that must be completed by each protocol. Consider an example of a network of N nodes. Almost all the CDA protocols introduce *Assignment* overhead of $O(N^2)$ due to Strong DAD in the beginning of the configuration process, whereas CFA protocols introduce only $O(N)$ overhead because communication takes place only with neighboring nodes. Whereas protocols for IPv4 typically require network-wide flooding; in IPv6, it is limited only to the local scope of the subnetwork. *Maintenance* overhead is produced by most stateful protocols that depend on the mechanism used by the protocol. Whereas centralized maintenance usually requires hard state information; in distributed maintenance, to check network address states, one-hop messages are periodically exchanged by neighboring nodes — either logically or physically. Even though in some protocols a leader is automatically elected, others require overheads for *leader election and maintenance*. Another overhead to consider in the case of network merging is to check and handle address collisions. When merging happens quite often, this may significantly influence the protocol performance. Finally, most protocols for IPv6 require extra overhead for *gateway discovery and selection*. With proactive discovery, GAs flooded to the entire network will produce additional overhead of O(N2).

Integration with Routing Protocol — Although most MANET research focuses on developing efficient routing protocols, address configuration issues emerge as complementary research to support ad hoc routing. As a consequence, control traffic from routing and configuration protocols concurrently traverse everywhere in the network. To reduce overhead caused by both protocols, several *integrated* approaches have been proposed [10, 11, 18, 26]. The main idea of the cross-layer designs is to reuse information from ongoing routing protocol traffic to support the addressing protocol or vice versa.

Generally, addressing protocols are *independent* of the underlying routing protocols. The reason is that many routing protocols exist — each optimized for a special network setting; therefore, addressing schemes should adapt to any of those. Even though not restricted to a specific routing solution, some addressing schemes will achieve optimizations over particular routing protocols. For example, requiring each node to maintain address lists, MANET-Conf [24] is inherently more suitable for proactive routing protocols. From the structural point of view, the physical cluster arrangement adopted by Weniger *et al.* [12] will be advantageous when hierarchical routing protocols are used. Ros *et al.* [27] showed that when the prefix distribution of IPv6-based protocols exploits the routing information, the total protocol overhead can be reduced significantly.

Scalability — The total number of nodes in the ad hoc network determines the bandwidth consumed by autoconfiguration protocols. We consider this aspect to compare the different protocols for their suitability for large-scale deployment. When network-wide broadcasting is required to perform DAD or gateway discovery, it has poor scalability. This is the reason why almost all CDA protocols for IPv4 are suitable only for *small* networks. In some protocols for IPv6, like Weniger *et al.* [12] and Park *et al.* [20], network-scope or global DAD is performed by leaders, and therefore, they can cover *larger* networks. Finally, protocols will have better scalability if only local communication is used, namely, simple one-hop broadcasting rather than flooding the whole network.

Discussions

The summary of our study is presented in Table 3. We compared sixteen protocols with the criteria listed in the previous section. The table shows that, except for Boleng [13], all protocols are compatible with IP. Some of them initially use only the MAC address to probe for an address from the neighborhood or directly start with a self-configured address. We also compared the protocols with respect to stateful or stateless address maintenance. Whereas nodes in stateless protocols are not required to store anything, stateful protocols require storing the list of IP numbers or spaces or the highest assigned address. Some of the protocols require DAD, and others necessitate the discovery of an agent or leader. We can see that in many stateless protocols, the self-configured node must ascertain that the address is not in use, whereas others assume that conflicts never happen, at least not during the initial assignment. On the other hand, with a stateful protocol, the leader automatically would allocate a unique address, or the agent would split its non-overlapped address space. The initial address can be tentative if the address is not allocated by a central authority. Whereas all stateless protocols are distributed in nature, stateful protocols can use a centralized or distributed structure.

Further, the allocation method can be either CDA or CFA; interestingly, CDA is performed with different types of DAD. Strong DAD requires a new configured address to be verified against all nodes in the network, typically using network-wide flooding and thereby, resulting in significant protocol overhead. Instead of explicitly asking approvals from other nodes, in Weak DAD, each node must modify its routing protocol and routing table entries to prevent packets from being transmitted to the wrong destinations. In Passive DAD, each node must maintain a table that enlists occupied addresses together with their corresponding conflict probability. The table must be modified each time a conflict is identified. If a certain threshold is exceeded, the conflict resolution must be performed. Both types of DAD are integrated with the underlying routing protocols. However, even though the traffic load is significantly reduced, both approaches require more capabilities and resources that might be heavy for handheld and embedded devices. Compared to CDA, CFA protocols offer a simple allocation procedure without requiring message flooding to detect conflicts. Assigning a conflict free address can be accomplished because address states are maintained in the network, and therefore, most stateful protocols use this allocation procedure. Nevertheless, the cost of the CFA approach is to maintain the states. In a distributed environment, there is a requirement for periodic signaling, whereas in a centralized case, there is a requirement to perform leader election and maintenance of a leader. Although the overhead of the CFA centralized approach is relatively low compared to that of a decentralized approach, the dynamic movement of MANET nodes may cause frequent changing of the leader, thus bringing unsteadiness to the address assignment and management.

The handling of the detection of network partitioning and merging is another feature that must be considered here. Whereas, typically, the PID is used to detect the events, hierarchical (leader-based) and flat-structured protocols perform the detection in different ways. In hierarchical protocols, the leader plays an important role in distributing packets with a PID announcement periodically, so that a node might detect partitioning (if it misses the announcements from its leader or receives announcements from another leader) or merging (if it receives announcements from more than one leader). In flat protocols, exchanging the PID information usually happens between neighbors, either physical or logical neighbors. As seen in Table 3, when detecting a network partitioning, only [9, 10, 12] among the IPv6 protocols do not perform reclamation. For IPv4, it looks quite different; most protocols either perform address reclamation or leader discovery. When the partitioning happens quite often, the reclamation procedure will not significantly impact the protocol operation, that is, the protocol still works properly to assign a unique address to a new node. The exception is the excessive overhead caused by messages that are sent by nodes to check the validity of or to release its address. However, when the leader recovery happens continuously because of frequent network partitioning, the assignment and management operation again will be affected greatly. Most IPv4 protocols perform address reconfiguration to solve conflicting addresses caused by merging. In the best case, only one of two nodes with a conflicting address must reconfigure its address, as in PACMAN.

Support for global connectivity goes hand in hand with gateway discovery and selection. The proactive discovery obviously introduces more overhead than the reactive approach because the GA must be sent to all nodes in the network at regular intervals. However, in the proactive solution, nodes will acquire more up-to-date information about gateway states than in the reactive. Acquiring fresh information about available gateways is not so crucial for protocols adopting the arbitrary gateway selection. However, it is critical for protocols adopting non-arbitrary selection because such nodes require the information in the GAs to facilitate the selection process. On the other hand, non-arbitrary gateway selection is more efficient in terms of the number of hops that are required to send a packet from the node to its gateway, as nodes are stipulated to choose the gateway having the shortest distance. Consequently, any movement of nodes might cause other nodes to change their gateways according to which gateway currently is closest to the nodes. Hence, such mechanisms will not work well in MANETs with highly dynamic topologies. For this reason, protocols adopting the distance-based gateway selection also have an alternative mechanism, that is, the network stability-based selection [9] or the default gateway mechanism [10]. In any case, all approaches require extra processing capabilities from the wireless nodes, because they must perform calculations from time to time while selecting the most appropriate gateway to be attached to.

Finally, the table also marks the overhead for each of the protocols in terms of the number of nodes and scalability. Although some protocols exchange a constant number of messages for the allocation of an address, periodic flooding of this information would cause a large number of redundant packet floods that make these protocols suitable only for a small network size. Thus ,one must take into account the protocol overheads before selecting a suitable protocol for a particular size of network.

Conclusion

We have presented an exhaustive comparison of different autoconfiguration protocols by outlining several issues that must be considered in wireless networks. We described various available techniques and explained the different criteria for comparing them. The relative merits and drawbacks are summarized to help practitioners and researchers in selecting the right protocol for their purpose. It is difficult to decide which protocol is the best, because each of them has its own uniqueness, complexity, advantages, and disadvantages. Furthermore, the choice of a proper protocol will involve many different factors, such as the routing protocol; the size of the network; and various metrics such as packet loss, network type (hybrid or standalone), the context and applications, and so on. Although each protocol has its own positive and negative aspects, we think each of them has a

The choice of a proper protocol will involve many different factors, such as the routing protocol; the size of the network; and various metrics such as packet loss, network type (hybrid or standalone), the context and applications, and so on.

rightful place in the field of wireless networking. Although we do not intend to recommend any one of these protocols, one can easily determine techniques that are suitable for a particular situation from this study. This article should help further research in the field of address autoconfiguration in wireless networks.

Acknowledgments

This research is funded by the Schlumberger foundation through the Faculty for the Future Program, the EU-funded MAGNET Beyond Project, and the Freeband PNP2008 Project. We also thank the reviewers for providing us their valuable input and suggestions to improve this article.

References

[1] R. Droms, "Dynamic Host Configuration Protocol," IETF RFC 2131, Mar. 1997.

[2] E. Guttman, "Autoconfiguration for IP Networking: Enabling Local Communication," *IEEE Internet Comp.*, vol. 5, no. 3, May–June 2001, pp. 81–86.

[3] K. Weniger and M. Zitterbart, "Address Autoconfiguration in Mobile Ad hoc Networks: Current Approaches and Future Directions," *IEEE Network Mag.*, vol. 18, no. 4, July 2004, pp. 6–11.

[4] Y. Sun and E. Belding-Royer, "A Study of Dynamic Addressing Techniques in Mobile Ad Hoc Networks," *Wireless Commun. and Mobile Comp.*, vol. 4, no. 3, Feb. 2004, pp. 315–29.

[5] C. Bernardos, "A Survey of IP Address Autoconfiguration Mechanisms for MANETs," IETF Internet draft, July 2005.

[6] S. Thomson and T. Narten, "IPv6 Stateless Address Autoconfiguration," IETF RFC 2462, Dec. 1998.

[7] T. Narten, E. Nordmark, and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)," IETF RFC 2461, Dec. 1998.

[8] C. Perkins *et al.*, "IP Address Autoconfiguration for Ad Hoc Networks," IETF draft, 2001.

[9] C. Jelger and T. Noel, "Proactive Address Autoconfiguration and Prefix Continuity in IPv6 Hybrid Ad Hoc Networks," *Proc. IEEE COMSOC*, Sept. 2005.

[10] S. Ruffino and P. Stupar, "Automatic Configuration of IPv6 Addresses for Nodes in a MANET with Multiple Gateways," IETF Internet draft, Feb. 2006.

[11] K. Weniger, "PACMAN: Passive Autoconfiguration for Mobile Ad Hoc Networks." Special issue, *IEEE JSAC*, *Wireless Ad Hoc Networks*, vol. 23, Mar. 2005, pp. 507–19.

[12] K. Weniger and M. Zitterbart, "IPv6 Autoconfiguration in Large Scale Mobile Ad Hoc Networks," *Proc. Euro. Wireless*, vol. 1, Feb. 2002, pp. 142–48.

[13] J. Boleng, "Efficient Network Layer Addressing for Mobile Ad Hoc Networks," *Proc. Int'l Conf. Wireless Networks*, June 2002, pp. 271–77.

[14] P. Patchipulusu, "Dynamic Address Allocation Protocols for Mobile Ad Hoc Networks," M.Sc. thesis, Comp. Sci., Texas A&M Univ., 2001.

[15] M. Mohsin and R. Prakash, "IP Address Assignment in a Mobile Ad Hoc Network," *Proc. MILCOM*, vol. 2, Oct. 2002, pp. 856–61.

[16] Zhihua Hu and Baochun Li, "ZAL: Zero-Maintenance Address Allocation in Mobile Wireless Ad Hoc Networks," *Proc ICDCS '05*, June 2005, pp. 103–12.

[17] M. Mutka, L. Ni, and H. Zhou, "Prophet Address Allocation for Large Scale MANETS," *Proc. IEEE INFOCOM*, vol. 1, Mar. 2003, pp. 423–34.

[18] E. Belding-Royer and Y. Sun, "Dynamic Address Configuration in Mobile Ad Hoc Networks," UCSB tech. rep. 2003-11, June 2003.

[19] D. O Mahony and S. Toner, "Self Organizing Node Address Management Protocol for Ad Hoc Networks," *LNCS*, 2003, pp. 476–83.

[20] I. Park, Y. Kim, and S.Lee, "IPv6 Address Allocation in Hybrid Mobile Ad Hoc Networks," *Proc. 2nd IEEE Wksp. Software Technologies for Future Embedded and Ubiquitous Sys.*, p. 58.

[21] N. H. Vaidya, "Weak Duplicate Address Detection in Mobile Ad Hoc Networks," *Proc. ACM MobiHoc '02*, June 2002, pp. 206–16.

[22] K. Weniger, "Passive Duplicate Address Detection in Mobile Ad Hoc Networks," *Proc. WCNC*, Feb. 2003.

[23] J. Jeong, "Ad Hoc IP Address Autoconfiguration," IETF Internet draft, Jan. 2006.

[24] S. Nesargi and R. Prakash, "MANETconf: Configuration of Hosts in a Mobile Ad Hoc Network," *Proc. IEEE INFOCOM*, 2002, pp. 1059–68.

[25] O. Troan and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) v. 6," IETF RFC 3633, Dec. 2003.

[26] E. Baccelli and T. Clausen, "A Simple Address Autoconfiguration Mechanism for OLSR," *Proc. IEEE Int'l Symp. Circuits and Sys.*, May 2005.

[27] F. Ros, P. Ruiz, and A. Gomez-Skarmeta, "Performance Evaluation of Existing Approaches for Hybrid Ad Hoc Networks Across Mobility Models," *J. Networks*, vol. 1, no. 2, June 2006, pp. 9–17.

Biographies

Novi Ineke Cempaka Wangi (n.i.cempakawangi@ewi.tudelft.nl) received her B.Sc. (cum laude) from Bandung Institute of Technology, Indonesia, and her M.Sc. from Delft University of Technology, The Netherlands, both in electrical engineering. She is currently a Ph.D. student at Delft University of Technology. She belongs to the Wireless and Mobile Communication group. Her research interests include ad hoc addressing, service discovery, and the federation of personal networks.

R. Venkatesha Prasad (vprasad@ewi.tudelft.nl) received his Bachelor's and Master's degrees in electronics and communication and industrial electronics from the University of Mysore, India, and his Ph.D. from the Indian Institute of Science, Bangalore. Since 2005 he has been with the Wireless and Mobile Communications (WMC) group of the faculty of Electrical Engineering, Mathematics, and Computer Science (EEMCS) at Delft University of Technology, participating in several Dutch and European research projects. From 2003 to 2005 he led a team of engineers at Esqube Communication Solutions Pvt. Ltd., Bangalore, who worked on various networking solutions. From 1999 to 2003 he worked as a consultant for CEDT and IISc, Bangalore, on VoIP application developments for a Nortel Networks project.

Martin Jacobsson (m.jacobsson@ewi.tudelft.nl) graduated with an M.Sc. in computer science from the University of Linköping, Sweden, in 2002. In 2003 he joined the Wireless and Mobile Communications group at Delft University of Technology, where he is working toward a Ph.D. degree. He has participated in several Dutch and European research projects. His research includes ad hoc and self-organizing wireless networking techniques in combination with infrastructure-based networks for personal networks.

Ignas Niemegeers (i.niemegeers@ewi.tudelft.nl) received a degree in electrical engineering from the University of Gent, Belgium, in 1970. In 1972 he received an M.Sc.E. in computer engineering and in 1978 a Ph.D. from Purdue University. Since May 2002 he has held the chair of Wireless and Mobile Networks at Delft University of Technology. From 1995 to 2001 he was scientific director of CTIT at the University of Twente. From 1981 to 2002 he was a professor with the Computer Science and Electrical Engineering faculties of the University of Twente. From 1978 to 1981 he was a designer of packet-switching networks at Bell Telephone Mfg., Antwerp, Belgium.