

IT Licentiate theses
2002-008

Implementation and Real-world Evaluation of Routing Protocols for Wireless Ad hoc Networks

HENRIK LUNDGREN

UPPSALA UNIVERSITY
Department of Information Technology





UPPSALA
UNIVERSITET

**Implementation and Real-world Evaluation of
Routing Protocols for Wireless Ad hoc Networks**

BY
HENRIK LUNDGREN

December 2002

DEPARTMENT OF INFORMATION TECHNOLOGY
COMPUTER SYSTEMS
UPPSALA UNIVERSITY
UPPSALA
SWEDEN

Dissertation for the degree of Licentiate of Philosophy in Computer Science
with specialization in Computer Communication
at Uppsala University 2002

Implementation and Real-world Evaluation of
Routing Protocols for Wireless Ad hoc Networks

Henrik Lundgren

Henrik.Lundgren@it.uu.se

Department of Information Technology

Computer Systems

Uppsala University

Box 337

SE-751 05 Uppsala

Sweden

<http://www.it.uu.se/>

© Henrik Lundgren 2002

ISSN 1404-5117

Printed by the Department of Information Technology, Uppsala University, Sweden

Abstract

A *wireless ad hoc network* consists of a number of mobile nodes that temporarily form a dynamic infrastructure-less network. To enable communication between nodes that do not have direct radio contact, each node must function as a wireless router and potentially forward data traffic on behalf of the others. New routing protocols are needed as the existing protocols used in wired networks adapt too slowly to the frequent topology changes induced by mobility and are inefficient in terms of resource consumption. During the last five to ten years more than 60 different ad hoc routing protocols have been proposed, optimized and partially compared based on simulation studies. However, we believe that these simulation studies need to be complemented with *real-world experiments* in order to gain practical experience and reveal real-world effects that may not be visible in simulations.

This thesis surveys the field of ad hoc routing and related real-world testbeds. We also describe our research results from three included papers:

In our active networking approach to ad hoc routing, protocol logic is carried inside data packets, which enables new protocols to be independently deployed at run-time. As a proof of concept, we have implemented two ad hoc routing protocols and successfully used them for routing a real-time sensitive audio stream. This prototype gave us a good understanding of the practical aspects of wireless networking and prepared good ground for protocol comparisons.

We have implemented a testbed called APE which enables easy management and analysis of real-world experiments. In real-world experiments, with up to 37 mobile nodes, we used the APE testbed to assess the reproducibility between repeated testruns and compared three different routing protocols. This testbed has become a crucial tool for discovering flaws in proposed protocols, as well as for benchmarking different routing styles.

During our implementation of the AODV protocol we discovered a performance problem that we could relate to some invalid assumptions about the wireless link characteristics. We explored this “communication gray zone” problem and implemented three countermeasures which we compared by using the APE testbed. As a result we could eliminate the performance discrepancy between simulated AODV and its implementation in the real-world.

Keywords: ad hoc networking, wireless networks, routing protocols, protocol evaluation, testbed, active networks.

Acknowledgments

I would like to acknowledge a few people who have been on the path of my journey so far. First, I would like to thank my two supervisors. My main supervisor, Per Gunningberg, for his mentorship and continuous support, and my co-supervisor, Christian Tschudin, for his engagement and for continuously working closely together with me, both in Uppsala and from/in Basel. I would not have made it this far without both of them! I am very grateful for the contributions and the social company of the MSc thesis students and assistants who have been working in our projects: Henrik Gulbrandsen, David Lundberg, Erik Nordström, Mattis Fjällström, Josh Fenessey, Johan Nielsen, Björn Wiberg and Stefan Lindström. Special thanks goes to Erik Nordström who, after his MSc thesis, continued to work part-time on our project and have contributed with great implementation efforts.

I am grateful for being part of the Communication Research (CoRe) group who's members have played an important role, both socially and scientifically, in the daily life at work. Thanks to all of them, especially the senior researchers Mats Björkman and Arnold Pears, and my PhD-student colleagues so far: Björn Knutsson, Thiemo Voigt, Bob Melander and Richard Gold. I also thank Arnold Pears and Mats Björkman for proof-reading this thesis. Thanks to all other people at DoCS.

I acknowledge NUTEK/VINNOVA for funding our projects. I acknowledge Ericsson AB for partly funding our initial project and for donating the wireless ORiNOCO PCMCIA cards. I also acknowledge PCC++ for partly financing my graduate studies.

I thank my parents Inger and Björn, as well as my sister Lotta and Martin for querying me about what I do, so that I have to explain my research in an understandable way. Finally, my deepest thanks goes to my fiancée Chrissie who makes my life meaningful outside work.

List of Included Papers

- Paper A** Christian Tschudin, Henrik Gulbrandsen and Henrik Lundgren, *Active Routing for Ad-hoc Networks*, Published in IEEE Communications Magazine, Special issue on Active and Programmable Networks, April 2000.
- Paper B** Henrik Lundgren, David Lundberg, Johan Nielsen, Erik Nordström and Christian Tschudin, *A Large-scale Testbed for Reproducible Ad hoc Protocol Evaluations*, In Proceedings of The Third Annual IEEE Wireless Communications and Networking Conference 2002 (WCNC 2002), March 2002.
- Paper C** Henrik Lundgren, Erik Nordström and Christian Tschudin, *Coping with Communication Gray Zones in IEEE 802.11b based Ad hoc Networks*, In Proceedings of The Fifth ACM International Workshop On Wireless Mobile Multimedia 2002 (WoWMoM 2002), September 2002.

Comments on my Participation

Paper A: In the work related to paper A, I participated in implementing two active routing protocols and parts of an audio streaming demo setting. I co-authored the sections on the routing protocol description and the implementation.

Paper B: I am the principal author of paper B. All co-authors have participated in the practical work with different effort and focus. I have participated in all parts, with focus on design and implementation of scenarios, implementation of analysis tools, conducting tests and performing the analysis.

Paper C: I am the principal author of paper C and participated in all parts. This work includes an implementation of an ad hoc routing protocol where I initially took part in the implementation phase and later focused on verification and validation.

Contents

1	Introduction	9
1.1	Problem Areas	11
1.2	Research Problems Addressed in This Thesis	12
1.3	Contributions	13
1.4	Thesis Organization	14
2	Routing in Ad hoc Networks	15
2.1	Proactive Routing	16
2.2	Reactive Routing	17
2.3	Other Approaches	17
3	Evaluations and Testbeds	19
3.1	Evaluation Techniques	19
3.2	Existing Real-world Testbeds and Experiments	20
4	Active Networking	23
5	Summary of Papers	25
5.1	Paper A: Active Routing for Ad-hoc Networks	25
5.2	Paper B: A Large-scale Testbed for Reproducible Ad hoc Protocol Evaluations	26
5.3	Paper C: Coping with Communication Gray Zones in IEEE 802.11b based Ad hoc Networks.	28
6	Summary and Conclusions	31
	Paper A	41
	Paper B	49
	Paper C	59

Chapter 1

Introduction

Wireless mobile communication for personal use is, and will continue to be, part of our everyday life. Standards for wireless communication have boosted the market; manufactures equip everything, from powerful laptops to small embedded devices, with hardware support for different radio technologies. The deployment of IEEE 802.11b [12, 13] wireless networks has increased dramatically the last few years, both in industry and in home networking. Bluetooth [27] is another technology, that aims at being small, cheap and built-in in almost everything. Both IEEE 802.11b and Bluetooth utilize the license-free 2.4 GHz frequency band and can operate in so called *ad hoc mode* (see below). The work presented in this thesis focuses on the ad hoc mode of IEEE 802.11b.

The term “ad hoc” often means improvised or for the needs of the moment for a specific purpose. In computer networking, we think of an *ad hoc network* as a wireless network without any infrastructure, e.g., wireless base stations. More thoroughly described, an ad hoc network consists of a number of nodes with wireless communication capabilities, that potentially move around and cause the network topology to change frequently. The type of device in ad hoc networks can range from embedded systems like sensors to powerful computers inside vehicles. The nodes can be scattered so that all nodes are not within radio range of each other. As per definition there is no available infrastructure, the nodes must rely on each other to relay the data packets in a multi-hop situation. Figure 1.1 illustrates the two types of networks; one with an infrastructure where traffic between the two clusters is forwarded and transported with help of base stations and pre-installed wires, and one infrastructure-less ad hoc network where intermediate nodes help to forward traffic. An ad hoc network often works autonomously, in isolation from other networks, but can very well be connected to other networks, e.g., the Internet, via a gateway. Unless stated otherwise, I will in this thesis work with the assumption that our ad hoc networks are isolated.

A typical usage scenario for ad hoc networks is a disaster area where pre-

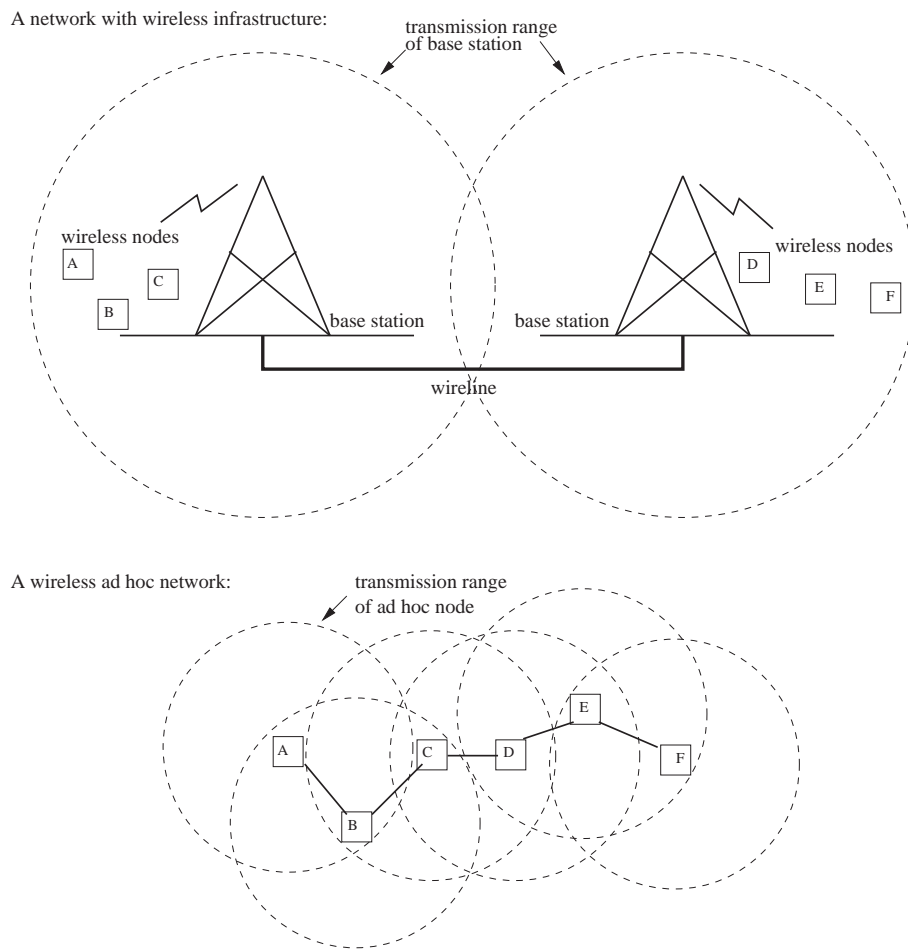


Figure 1.1: In the upper diagram the mobile nodes communicate via a managed infrastructure for wireless terminals. In the lower diagram the nodes form a wireless ad hoc network by themselves.

existing communication infrastructures have been eliminated due to e.g., an earthquake or a terror attack. Emergency personnel needs to quickly establish communication to different rescue teams as well as to people in distress. Given the maturity of the technology, an ad hoc network would be a good choice for deployment. All nodes – carried by people and/or vehicles – would continuously organize themselves and co-operate in order to forward each others traffic when needed. Another usage area for wireless ad hoc technology are sensor networks. For example, sensors can be dropped from airplanes over contaminated areas or earthquake zones and be used to register activity. Military scenarios can easily be imagined, e.g., a number of mobile units that need to maintain contact. A less dramatic example would be a university campus area with no, or limited, wireless LAN coverage,

where student groups might wish to establish temporary wireless networks in order to work on collaborative projects or laboratory exercises.

Possible applications in an ad hoc network do, of course, depend on the scenario. For a disaster area or a military scenario it could be voice communication or transmission of camera snap shots. In a sensor network it might be periodic transfer of collected measurement data, while at a conference or spontaneous meeting the applications could be file sharing, or in cases where Internet connectivity is present, even web browsing and email. In this thesis we use applications like ping, multimedia streaming and web browsing for our experimental work.

1.1 Problem Areas

The characteristics of ad hoc networks and the nodes of which they are composed impose a number of interesting design and evaluation problems. The nodes in an ad hoc network are normally battery-driven and therefore it is important to design a system with energy conservation in mind. Another scarce resource is the shared wireless medium which has links with limited and variable bandwidth. This puts constraints on the bandwidth consumption for both control overhead and applications. The wireless medium also imposes a number of security threats like eavesdropping and denial-of-service attacks. For stub networks, where the nodes have Internet access via a gateway, the addressability of nodes inside the ad hoc network from nodes outside the network is not straightforward. Other configuration issues of ad hoc networks are complex too: the nodes need to agree on frequency channel and addresses, advertise services such as gateway functionality and possibly exchange cryptographic keys.

However, the most fundamental and probably the hardest problem, which in high degree is related to the issues above, is *routing*. A routing protocol uses some kind of distributed algorithm to acquire information about how to reach other nodes in the network and uses this to build up a routing table. A node consults the routing table whenever it has to decide how to forward a packet towards its destination. Today's standard Internet routing protocols do not perform well for volatile network topologies. The main problems with using such protocols in the ad hoc context are slow convergence and high bandwidth consumption. Besides this, CPU processing time and battery power are limited resources that must be taken into account, e.g., by designing routing protocols that permit nodes to go into sleep mode occasionally.

1.2 Research Problems Addressed in This Thesis

It is a commonly accepted view that new routing protocols have to be designed to cope with the unique requirements of wireless ad hoc networks. We believe that given the diverse scenarios and applications for ad hoc networking it is impossible to design a single protocol that is best suited for every situation. Our initial work was driven by the question:

- *How can we design and implement more flexible and adaptive ad hoc routing protocols?*

Our approach is to utilize the concept of active networking. By using a hybrid active networking approach the forwarding state is still maintained in the routers but the routing logic is moved out of the routers and into the data packets. We call this *active routing*. This approach enables the nodes to define and deploy routing logic at run-time in order to adapt to special circumstances and requirements. It is also possible to run several routing protocols in parallel. We have built an experimental testbed and implemented two routing protocols to demonstrate the feasibility of this approach. We continued our experimental work and focused also on ordinary (“passive”) ad hoc routing protocols.

Standardization work is ongoing within the Mobile Ad hoc Networking (MANET) working group in the Internet Engineering Task Force (IETF) [52] and currently there exist about a dozen “classical” protocol proposals. These protocol proposals must be carefully evaluated before any standardization decision can be taken. So far, most evaluations and comparisons have been performed in simulation environments. One of the main benefits of simulations is that one can easily alter different parameters, like the number of nodes and their mobility pattern. This kind of evaluations are crucial in the early design phase of the protocols as one quickly can explore the parameter space of a new protocol and determine its viability in the context of other protocol proposals. The importance of accurate models increases as the protocols become mature and the research community strive towards reaching consensus regarding standardization. Existing simulation models have problems to accurately model the physical and link layers. We argue that real-world experiments are necessary and will provide most useful insights as well as providing knowledge how to improve simulation models. Experiments must be carefully designed and the results compared with simulations.

When planning experiments aimed at comparing different protocols the objectives and goals of an experiment are often clear. However, carrying out an experiment is far from trivial. One of the most difficult challenges that

will, to some extent, be dealt with in this thesis is:

- *(How) can we make the real-world mobile ad hoc routing experiments reproducible?*

We present the Ad hoc Protocol Evaluation testbed (APE) and a new mobility metric called *virtual Mobility* that is based on signal quality variations as perceived by the nodes. An experimentation environment was established at the Uppsala University IT department where the department provides IT students with laptops. The Communication Research group [14] provides wireless cards in exchange for the students' participation in our ad hoc networking research experiments. We present results from experiments involving up to 37 nodes where we successfully assessed the reproducibility.

While some design flaws may be obvious in simulation, other context problems may not be discovered until one is faced with technical implementation details and real-world conditions. Furthermore, some shortcomings may be subtle and require high confidence in the protocol implementation under test. Our third question is:

- *How can we identify and capture the impact of real-world effects that are not visible in simulations?*

The first step towards finding answers to this question is to start “doing the real thing”. We spent much effort in implementing a routing protocol for which we performed extensive testing to assure compliance to the specification. During the evaluation of that protocol we discovered performance behaviors that did not correspond to simulation results. Further investigation revealed that the broadcast HELLO beacons, necessary for the protocol, are not completely dependable as an indicator of communication potential in certain areas which we call “communication gray zones”. An incomplete link layer implementation prevented this effect from manifesting itself during protocol simulations.

1.3 Contributions

The main contributions in this thesis are:

- We have experimentally demonstrated the feasibility of using an active networking concept in an ad hoc networking context. Data is passively forwarded at layer 2.5 (i.e. below IP) while active packets (mobile programs) are responsible for adjusting the nodes' forwarding behavior.
- We have implemented a real-world testbed for ad hoc routing protocol evaluation with the means to assess experimental reproducibility. The testbed has been made publicly available and has so far been downloaded approximately 800 times [3].

- A high quality implementation of the AODV routing protocol for the Linux operating system was produced. It was successfully tested at the First AODV Interop Event [5] and has been made publicly available with around 1000 downloads [2] so far.
- We have experimentally shown that IEEE 802.11b based ad hoc networks potentially suffer from communication gray zones and examined the effectiveness of three suggested techniques for addressing this problem.

1.4 Thesis Organization

This thesis is organized as follows. Section 2 gives an introduction to ad hoc routing. Section 3 describes different evaluation techniques and compares existing real-world testbeds. Section 4 introduces the concept of active networking. Section 5 summarizes the papers included in this thesis. Conclusions and an outline of future work appear in Section 6 before the reprint of the three published research papers.

Chapter 2

Routing in Ad hoc Networks

It is the task of the routing protocol to create, maintain and re-create routes which preferably should be durable. To run smoothly, an ad hoc network requires a quick and adaptive routing protocol that at the same time does not consume too much of the already scarce wireless bandwidth.

Existing routing protocols used in the Internet do not function well in such an environment. More specifically, the Routing Information Protocol (RIP) [49] suffers from slow convergence and the well-known count-to-infinity problem. The Open Shortest Path First (OSPF) [55] converges only slightly faster than RIP and in addition it has high bandwidth consumption.

Over the years several candidate protocols have been proposed both inside and outside the IETF MANET WG [52] and more than a handful of prominent candidates are still active. These protocols must now face the next step towards standardization, namely real-world implementation and interoperability. One of the requirements for an Internet Draft for moving to Experimental RFC is that there must exist several independent real-world implementations that interoperate.

The multitude of independent implementations assesses the readability and completeness of the protocol description. On the other hand, real-world implementations also force the protocol designers to deal with specific real-world implementation issues. For example, several protocol proposals use link-layer feedback as a way to determine connectivity with neighbors. This is easy to implement in a simulation environment, but currently there is no hardware support for this information to be retrieved from wireless network adapters. Therefore, the protocols are forced to use their secondary choice for this functionality, usually neighbor sensing through periodic broadcast HELLO beacons. It is important to realize that real-world constraints affect the protocol design and thereby also potentially its performance.

There exist a number of strategies to choose from when designing an ad hoc routing protocol. One design decision to take is whether to be proactive and always maintain routes to every possible destination, or to be reactive

and only discover and maintain routes on demand. Another decision is to choose between flat and hierarchal routing structures. The use of geographic location data (e.g., via GPS) and the exchange of this information between nodes is another choice among even more possibilities. The goal is to design a protocol that has one or more of the following properties: low overhead, quick adaptation, power-awareness, reliability, security and scalability.

2.1 Proactive Routing

The concept of proactive routing means that all nodes (routers) exchange route information periodically, or whenever the network topology changes, in order to maintain a consistent, complete and up-to-date view of the network. Each node uses the exchanged route information to calculate the costs to reach all possible destinations. Thus, if a destination can be reached, a route is always at hand; which avoids the delay associated with finding a route on demand.

Proactive routing techniques can be divided into either *distance vector* or *link-state* algorithms. Both techniques require each router to periodically broadcast route information and to calculate the shortest path to others. In distance vector, each router maintains a vector with the distance to *all routers* and periodically broadcasts this vector to each of its *neighbor routers*. Each router updates its own routing table by calculating the shortest path to each router using the distance vectors received from others.

In link-state, the relationship between whom to collect state about and whom to disseminate this information to is reversed compared to distance vector; each router maintains the state of its links to its neighbors only and broadcasts this information to all other routers. Using the link-state information from all other routers, each router computes a complete picture of the network and calculates the shortest path to all nodes.

The main advantage with proactive routing schemes is that there is no initial delay when a route is needed. On the other hand, they typically have higher overhead and longer route convergence time than the reactive schemes presented in the next subsection, especially in the case of high mobility. For better performance in ad hoc networks both distance vector and link-state algorithms have been modified. Examples of distance vector protocols are Routing Information Protocol (RIP) [49], Destination-Sequenced Distance Vector (DSDV) [67, 66] and Wireless Routing Protocol (WRP) [56]. Examples of link-state protocols are Open Shortest Path First (OSPF) [55], Optimized Link State Routing (OSLR) [11, 36], Topology Broadcast Reverse Path Forwarding (TBPRF) [59], Source Tree Adaptive Routing (STAR) [21], Global State Routing (GSR) [35], Fisheye State Routing (FSR) [64, 24] and Landmark Routing Protocol (LANMAR) [65, 23].

2.2 Reactive Routing

Reactive routing is generally not dependent on exchanges of periodic route information and route calculations. Instead, whenever a route is needed the node has to perform a route discovery (disseminate a route request throughout the network and wait for a route reply) before it can send any packets to the destination. The route is maintained thereafter until the destination becomes inaccessible or is no longer needed. Thus, a reactive protocol does not spend resources on exchanging route information or maintaining inactive routes. On the other hand, the route discovery process adds a significant initial delay to a transmission and has high resource consumption. In fact, if there are many data sources in the network the overhead increases and will be comparable with, or even exceed, those for proactive schemes. Examples of reactive protocols are Ad hoc On-demand Distance-Vector (AODV) [68, 69], Dynamic Source Routing (DSR) [39, 38], Associativity Based Routing (ABR) [82, 81], Signal Stability based Adaptive routing (SSA) [18], Temporally Ordered Routing Algorithm (TORA) [62] and Relative Distance Micro-discovery Ad hoc Routing (RDMAR) [1].

2.3 Other Approaches

There exist several other approaches to the design of ad hoc routing protocols. For example, there are hybrid approaches like the Zone Routing Protocol (ZRP) [63, 31] which utilize both proactive and reactive concepts: each node maintains a zone (e.g., with a 2 hop radius) wherein it uses proactive routing [29], while for nodes outside this zone it uses reactive routing [30, 28]. The overhead is limited as the proactive route maintenance only concerns the local neighborhood of a node and the reactive route search is limited to querying a set of selected nodes.

It is common that the routing protocol has a flat architecture, but some protocols are based on clustering and hierarchical architectures, such as Clusterhead Gateway Switch Routing (CGSR) [10] and Distributed Mobility-Adaptive Clustering (DMAC) [4]. Advantages with such schemes are more robust routes and generally fewer routing control messages, although they might require periodic messages to maintain the clusters. A clear disadvantage is the centralization of routes through cluster leaders which put heavy load on these nodes and at the same time makes them “single points of failure” (of course, in the case of failure, etc. they are replaced but that takes time and many data sessions are affected simultaneously).

The route discovery procedure in reactive protocols like AODV and DSR is based on some variation of flooding and is usually relatively costly (despite various optimizations). The LAR [42] protocol aims at limiting this cost further by using location information e.g. via GPS. A source node can then

determine the expected location of a destination node. Based on this, it defines a request zone and intermediate nodes only forward a route request if it is located within the request zone. The drawbacks of this scheme are that it does not take obstructions for radio transmission into account and that the exchange of location information increases the overhead.

Another way of reducing the costly discovery procedure is to have multiple paths available as a backup, thereby reducing the need for a new search when a route breaks. DSR has this built-in by adding destinations from overheard transmissions to a route cache, while there exists no similar functionality in AODV. AODV-BR [43] and AOMDV [53] experiment with the use of multiple paths for AODV and report on improvements for scenarios with high mobility and low to moderate traffic load.

Reduced control overhead usually means better scaling and less power consumption. There exist several approaches that relate more closely to power consumption. PAR [77] is an approach that takes battery life time into account and selects routes that favor long overall battery life. Another approach is to adapt (minimize) the transmit power, which also reduces interference and increases the spatial reuse. Usually such power-aware protocols work in close cooperation with the underlying MAC protocol. For example, PARO [25, 26] assumes that the nodes can adjust the transmission power per packet and uses this to create routes with low power consumption. However, the transmit power has to be carefully selected; different transmit power levels in the network potentially introduce uni-directional links and lowering the transmit power increases the risk of bit-error. There are also approaches to power consumption that are not directly related to routing, e.g., to temporarily put the wireless card into sleep mode when the node will not participate in a nearby transmission [76].

Normally the basic protocol designs do not take security into account. They simply assume that all nodes are friendly and behave well. However, this assumption may not be true in a real-world situation. Interest in this area has increased lately and efforts have been spent on identifying security threats together with counter-measures in general [7, 92, 34, 90] and for routing in particular [54, 33, 32, 89, 75].

The survey above includes relevant examples of different routing protocols. However, it is not a complete list of all ad hoc routing protocols that ever existed. See [46, 87] for a list with more than 60 routing protocol proposals, including old out-dated Internet drafts. Additional surveys on ad hoc routing protocols and techniques can be found in [73, 74, 20].

Chapter 3

Evaluations and Testbeds

A number of evaluation considerations are discussed in RFC 2501 [15]. This RFC document lists quantitative metrics like end-to-end data throughput and delay, route acquisition time, out-of-order delivery percentage and efficiency in terms of number of data and control bits transmitted compared to data bits delivered. It also points out that the networking context is important; with parameters like network size, network connectivity (i.e., the average number of neighbors), rate of topological change, traffic patterns and mobility as key elements. Other metrics exist and could for example be application specific. The metrics selected for investigation in a given experiment should be chosen in relation to the scenario, the applications used and the purpose of the experiment.

3.1 Evaluation Techniques

The most common technique used to evaluate ad hoc routing protocols is to perform experiments in a simulation environment [58, 22, 61]. Simulation enables repeatable experiments and variables to be easily altered and examined. However, the accuracy of simulation results depends heavily on the quality of the models used for the physical layer and the link layer. Results in [78] show that depending on the modeling of the physical layer, simulation results can vary significantly and even change the relative ranking of the compared routing protocols. The authors in [9] state that standalone simulations do not really fit the actual needs of wireless application developers and that there is an important lack of real experiments that prove the feasibility of wireless protocols. Another important issue is that the routing protocol implementations are often tailored to the simulation environment and do not take the interaction with the ordinary protocol stack into account. Typical studies that have used simulations include [6, 37, 70, 16, 44].

Another approach is to evaluate protocols in their native OS environment and to emulate the lower protocol layers in order to steer connectivity in a

controlled way [40, 91, 19]. This removes the unwanted uncertainty associated with varying radio conditions, but also hides potential effects thereof. The mobility has to be artificially generated by changing the network topology using some kind of MAC address filtering utility. Thus, with controlled mobility and a signal propagation model it is possible to reproduce testruns in “emulation mode”.

An alternative to MAC address filtering is to collect traces from simulations or real-world experiments, and use them to run trace-based emulations [57].

To cover all aspects of protocol behavior it is ultimately necessary to run experiments in real-world settings. Only then can all implementation aspects be taken into account and inaccurate models of the physical layer and the link layer be eliminated. The main drawback is that radio conditions are difficult to control, which makes reproducibility a difficult task. There is also a cost consideration here, as the size of the test network topology is often limited by the availability of hardware and manpower.

3.2 Existing Real-world Testbeds and Experiments

Most real-world ad hoc testbeds described in the literature work in small-scale settings and are often tied to a specific routing protocol. Large-scale testing imposes the problems of hardware availability, software configuration and participant management.

The CMU testbed [51, 50] is built upon a FreeBSD implementation of the DSR protocol [39] and consists of five moving nodes, two stationary nodes and one additional moving node running Mobile IP. The moving nodes are transported in cars at 25km/h to 40km/h in an area 700 by 300m. As the experiments took place outdoors they equipped each node with a GPS unit and logged GPS information together with other information in order to be able to replay the testruns in a visualization tool. In the experiments the authors used UDP communication to emulate a “push-to-talk” communication and FTP to perform file transfers. Packet loss rate and packet jitter was examined as well as an heuristic for adapting the retransmission timer. The authors pointed out the importance of in-lab testing and for this purpose they implemented a simple MAC-filter utility.

At Georgia Tech they report on a five node testbed [83]. Their setting is stationary and mobility is generated by unplugging the network cards and thereby forcing topology changes. The authors evaluated their ABR implementation [82] and measured end-to-end delay, throughput and route discovery time with respect to varying packet size, beaconing interval and hop count. The main conclusions drawn were that beaconing had very little impact on the measured metrics, but packet size affects the end-to-end delay.

In the DAWN testbed at BBN [72] specialized hardware is used that is

based on Nokia Wireless Routers. The authors report on real-world experiments with up to ten nodes where ping throughput and ping round trip time were examined with varying offered load on the network. Emulation was used to scale up the number of nodes when evaluating a variety of their own link state routing protocols. The authors noted that the throughput was only about 1/10 of the nominal bit rate and that after a certain threshold of offered load the throughput decreased drastically.

“Testbed on a desktop” [41] is a small scale testbed built for wireless or wired ad hoc networks. The wireless testbed consists of nodes with low-gain antennas and special shielding hardware that enable experimenters to set-up multi-hop configurations within just a few meters. The wired part of the testbed emulates the wireless channel over special cables with attenuators and combiners/splitters, thereby controlling the radio propagation effects. As these cables are connected directly to the wireless network interface the nodes can still use real wireless network interface cards and real wireless network interface drivers. As the testbed consists solely of hardware it is independent of protocol and operating system. It could be used in any of the previous described testbeds, but requires an external antenna interface on the wireless network interface card. The authors outline a few possible multi-hop configurations, but the flexibility of the wired part of the testbed is limited by the 3-port signal combiners.

In Table 3.1 we compare our APE testbed (described in Section 5.2 and paper B) with the above testbeds except for “Testbed on a desktop” since, in contrast to the other, it only consists of special hardware.

	<i>CMU</i>	<i>GA tech</i>	<i>BBN</i>	<i>APE</i>
Properties:				
Real-world	yes	yes	yes	yes
Wireless emulation	yes	no ¹	yes	yes
Hardware	laptops + GPS	laptops	Nokia Rooftop	laptops
OS	FreeBSD	Linux	DAWN/FreeBSD	Linux ⁵
Routing Protocol	DSR	ABR	NLS ³	Independent ⁶
Visualization	yes	no	no	yes
Analysis tools	yes	yes ²	yes ⁴	yes
Software Distribution	no	no	no	yes
Reproduc.	no	no	no	yes
Experiments:				
Environment	outdoor	outdoor	outdoor	indoor
Mobility	cars	NIC removal	walking	walking
# nodes	8	5	10	37
Protocol(s)	DSR	ABR	NLS	AODV, OLSR, DSR, TORA, LUNAR
Traffic	ping, ftp, synthetic voice	ping	ping	ping, ftp, http, MP3
Metrics	packet loss, jitter	throughput, e2e delay, route disc. time	throughput, RTT, QoS	packet loss, hop count, comm. gray zone

Table 3.1: A comparison of real-world ad hoc testbeds including reported experiments

¹Not reported on, but should generally be possible.

²Not reported on, but at least some limited form of tools must exist.

³Near-sighted Link-state and other link-state variants.

⁴Not reported on, but at least some limited form of tools must exist.

⁵APE boots directly from a CD and does not require any pre-installed OS.

⁶Any protocol developed under Linux can be incorporated.

Chapter 4

Active Networking

A basic hypothesis would be that the flexibility which ad hoc routing protocols must demonstrate is best matched by flexibility in the way one implements these protocols. This link between wireless and active networking, which 3 years ago was the starting point of our studies in the field of ad hoc routing, was also recently picked up e.g. in [71]. In this section we briefly recapitulate the major elements of the active networking approach.

Active networking is a concept where customized programs can be injected into the network in order to steer or extend the functionality of a given active net. This is achieved by inserting program code into data packets. The code in the data packets can then be processed at a remote node or even at every node it traverses (see Figure 4.1 for an illustration). This enables remote programming and configuration of network nodes. The nodes in the network can in turn perform computations based on the packet's content, modify the packet content or create new packets. Generally, the concept of active networking enables quick adoption of new standards and algorithms, easy extension of network nodes and protocol flexibility because of instant deployment.

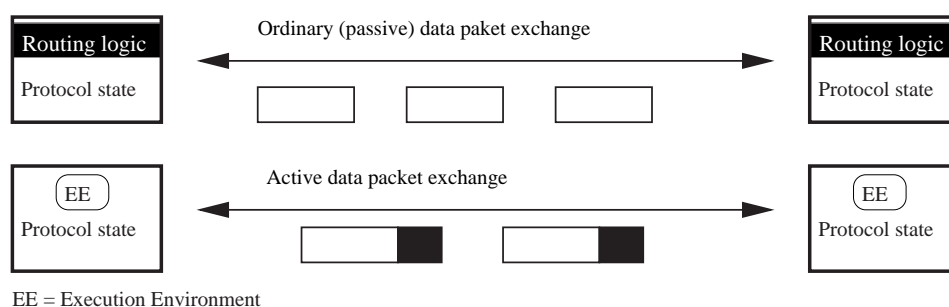


Figure 4.1: The routing logic is moved out of the routers and into the data packets (which carries executable code and thus become active).

The first use of active networking was within the SOFTNET distributed packet radio network [88]. Packets in SOFTNET contain programs which are executed as soon as they arrive. As an example, it is possible for packets to retransmit themselves to other nodes. However, SOFTNET's active networking approach fell into disuse. Ten years later the messenger paradigm of communication was presented in [84]. The protocol stack with its machinery and passive protocol data units were replaced with an execution environment and messenger packets that carried code that instructed the node how to proceed, e.g., change states within the node and how to interpret incoming messengers.

The term "active networking" was first coined in [80]. The authors distinguished between two common approaches of active networking: the programmable switch approach and capsule approach. In the programmable switch approach one sends special packets to inject programs into network nodes. These packets contain a special tag within each packet which will trigger processing of a packet's content. In the capsule approach *every* packet is a program. This approach is somewhat similar to the Messenger concept. Each packet has a set of instructions which are executed at the host machine. Capsules can leave states or even a complete program on each node. A network node executing capsules can for example act on data or output new capsules. Good surveys over active networking efforts are available in [79, 8].

A third alternative is to use a hybrid approach between these two. We utilize this in paper A to perform active routing. In our case, the routing state is kept at each node in form of special forwarding entries, while the complete routing logic is carried as code inside active control packets. These packets are executed at each node they traverse, modifying the routing state of those nodes. The application data packets are still ordinary (passive) packets.

The motivation behind bringing active networking into the ad hoc networking field is based on our belief that no single ad hoc routing protocol can be universally pre-eminent or best suited for all different situations. By using an active networking approach it is possible to deploy new or context specific routing protocols at run-time and run instances of different protocols in parallel. Beside the task of expressing ad hoc routing protocols by means of mobile programs only, another interesting challenge was the use of slow active packets that steer the forwarding of delay sensitive audio packets.

Chapter 5

Summary of Papers

This section contains summaries and retrospective discussions of the papers included in this thesis.

5.1 Paper A: Active Routing for Ad-hoc Networks

We have implemented a hybrid active networking environment where active routing packets are used to configure a passive data forwarding layer. The paper describes the forwarding architecture, our active networking execution environment and one of the implemented active routing protocols used for routing a delay sensitive audio stream.

With hybrid active networking we mean that the routing state is partitioned. Routers maintain special forwarding entries, but the routing logic and routing data is carried inside active control packets which are used to set up these forwarding entries. Passive data packets can then choose to use these special forwarding entries. In order to distinguish these special forwarding entries an additional packet header field is needed. We have chosen to adopt the simple active packet format (SAPF) [17] and use it at "layer 2.5" between the link layer and network layer. The SAPF header consists of a single 63 bit selector field and the selector is used to index entries in the forwarding information data base. The entries can point to e.g. outgoing interfaces, special packet processing software such as the IP stack, or sockets from which applications can read the packet's payload. Using SAPF at layer 2.5 can speed up forwarding decisions as e.g. IP header processing can be avoided.

The execution environment we have implemented simply listens on a SAPF socket for incoming active control packets. As a first programming language for the active control packets we chose Unix Bash scripts: A separate UNIX process is started to execute each received bash script. As a first active ad hoc routing protocol we implemented an "active" version of Cellular IP that we call ACIP. It is composed by three sub-protocols; a

neighbor discovery, a Minimum Spanning Tree (MST) wave-algorithm and the delivery path setup using the entries installed by the MST wave.

The use of Unix Bash scripts as a programming language is a rather crude approach and suffers from slow performance and security issues. However, our experiments show that this is sufficiently fast to track topology changes and do active routing in an ad hoc environment. With a demo we showed that it is even possible to stream delay sensitive audio in a dynamic ad hoc network without major hick-ups.

Discussion

We have implemented two active routing protocols as proof of concept. Although debugging BASH scripts is not a pleasant activity, the implementation was relatively straightforward as it avoided kernel programming and interaction with the IP-stack. However, to seriously continue on the interesting track of active ad hoc routing using SAPF the primary issue would be to change the programming language to achieve better performance and to enhance the security properties of the protocols.

5.2 Paper B: A Large-scale Testbed for Reproducible Ad hoc Protocol Evaluations

During the work described in paper A we gained experience in both implementing and running (active) ad hoc routing protocols. From this point forward our work shifted towards protocols proposed within the MANET working group in the IETF, while still keeping our focus on real implementations. In paper B we describe the architecture and implementation of our Ad hoc Protocol Evaluation (APE) testbed and define a new mobility metric that we use in the analysis of our experiments.

Simulation is a natural and necessary step in the design and development of ad hoc protocols. However, these simulations need to be complemented with real-world experiments, as existing simulations do not accurately model the dynamics of real-world propagation and tend to dismiss issues of a real implementation e.g., how to implement a link-layer notification. Previous testbed projects report on experiments with 10 nodes or less, and often use stationary settings when measuring performance. To the best of our knowledge, no testbed project reports on how to reproduce physical mobility or on comparisons of different protocols.

The APE testbed has a modular architecture allowing an experimentalist to plug in different protocols and traffic generators. We use strict choreography instructions to the participants. A new mobility metric, derived from measured signal quality, gives us means to assess the reproducibility of repeated non-stationary testruns, independent of the routing protocol used.

The measured signal quality data can also be used to replay the topology changes during the testrun in our tool called APE-view. Furthermore, we expect such data to be useful when validating simulation models e.g., for running trace-driven simulations.

APE is designed for easy installation, deployment and handling. Installation is as simple as “downloading and extracting” a single file. It is a self-contained system that can be booted on arbitrary machines independent of pre-installed operating systems. During the testruns the participants only need to follow on-screen movement instructions and afterwards the collected measurement data is uploaded to a collect-node, all with minimal user intervention. This enables us to scale up experiments without significantly increased administrative overhead. At the time of writing paper B, three protocol implementations MAD-HOC AODV [48], OLSR-INRIA [60] and UMD-TORA [85] were available and supported by the APE testbed. We report on our initial experiments where we scaled up to 37 node settings and used the new mobility metric to assess the reproducibility. Analysis of successful Ping request/reply during mobile scenarios revealed that OLSR-INRIA behaves as expected with successful delivery over multiple hops and up to 10 seconds re-route time, while MAD-HOC AODV often fails to re-route in multi-hop topologies. UMD-TORA was unstable and we experienced problems with crashes. Therefore, further comparative analysis was limited at the time. We conclude that the APE testbed enables accurate side-by-side protocol comparisons in mobile settings and that it also can be useful in the early debugging phases of a protocol implementation. To make full use of the APE testbed we now needed fully functional routing protocol implementations.

Discussion

We have planned for a “grand comparison” to test all available protocol implementations. However, this test has not yet taken place. A number of factors have delayed this activity. First, the number of high quality implementations is still very low. There are a number of good AODV implementations, but we want to compare different protocols. Also, the implementations often seem to be a one time effort and as Internet Drafts are continuously changing the implementations soon get outdated which makes potential results less useful. Second, the cost of involving a large number of people and obtaining sufficient hardware is still high. A large-scale test requires substantial planning and a clear picture about the value of the expected outcome. Third, simple traffic patterns have to be replaced with advanced traffic generators, which in turn requires enhanced analysis tools. Fourth, TCP has to be more thoroughly investigated in small-scale settings first, because TCP severely suffers from the interactions with ad hoc routing even in small-scale settings.

Our virtual Mobility metric (vM) provided relatively good grounds for assessing the reproducibility. When running small-scale test and with nodes placed on wheel-tables, the similarities were very clear. However, during large-scale tests the result can become “blurred” as such tests involve many people and all people may not move exactly in the same way throughout all testruns. In such situations we have found that analysis of the link changes during the testrun can provide additional useful insights.

An open issue with the vM metric is whether to take signal strengths into account only from adjacent nodes or to include signal strengths over several hops. Another issue is whether to calculate vM over all links or over links that were used to forward traffic. When providing vM -graphs for testruns we have scaled the values according to a path loss formula for the indoor environment. Another formula has to be developed and applied before performing tests outdoors.

We have noted that extensive logging potentially affects the routing daemons’ behavior. For example, the sending of beacons may be delayed if the processor is loaded with other OS tasks. Therefore, one must carefully think about what and how to log.

5.3 Paper C: Coping with Communication Gray Zones in IEEE 802.11b based Ad hoc Networks.

We implemented the AODV protocol and used our APE testbed to evaluate its performance. At specific geographic locations we observed that although the neighbor sensing mechanism indicated reachability, data packets were not necessarily successfully transmitted to or from that node. We call these areas “communication gray zones”.

For AODV, in particular, this problem lies hidden in the properties of the HELLO messages that real-world AODV implementations are forced to use for neighbor sensing. When transmissions occur at lower bit rates each bit contains more energy than it would at higher rates. Thus, transmissions at lower bit rates reach further than at high rates. HELLO messages are broadcast at a basic bit rate and unicast data transmissions are sent at higher rates (up to 11Mbit/s in IEEE 802.11b) which is the main reason why gray zones appear. Furthermore, broadcast messages are not acknowledged and hence receiving a HELLO message is not a guarantee of a bidirectional link. Over weak links, the reception of a HELLO message does not necessarily indicate a useful link as AODV’s HELLO messages are small in size and thus have higher probability than ordinary (longer) data packets of being successfully transmitted. Finally, fluctuating links at the transmission borderline can lead to spurious HELLO messages which, once received, do not reflect correctly whether persistent communication between two nodes is possible

or not. As a consequence this means that stable and longer routes may be replaced by shorter but unreliable ones.

Three different mechanisms that help neighbor sensing and reduce gray zones were evaluated in a real-world setting: First, *Exchanging neighbor sets* enables receiving nodes to deduce whether the link to the sender is bidirectional or not. This addresses the unidirectional link problem, but not the difference between unicast and broadcast transmissions. Second, *Requesting N consecutive HELLO messages* from the same source before accepting it as a neighbor will bring stability into the changes in neighbor sets and ultimately the routes. Typically, N would be set to 2 or 3. This addresses the problem of fluctuating links, but not the difference in unicast/broadcast sensitivity. Finally, a third way to improve neighbor sensing is to set a *SNR threshold for control packets*: control packets are discarded when they are received with a signal quality that is below some threshold. Intuitively this will counteract the gray zone problem as it forces AODV to choose a longer route when link quality is so bad that data supposedly cannot get through while HELLO messages still can.

The negative effect of gray zones decreases for all three mechanisms and the SNR threshold approach performed best. For simple ping traffic the successful packet delivery ratio increased from 91.9% to 99.1% and for a streaming MP3 application it increased from 97.9% to 99.7%.

Discussion

The SNR threshold approach yielded the best result. The first two approaches also significantly increased the performance: Their advantage is that they do not need driver specific SNR information. However, the "Exchanging Neighbor Sets" and "Requesting N -consecutive HELLO" introduce a delay in the neighbor sensing. In mobile scenarios this delay helps to shrink the gray zone area, but it would not have the same effect if nodes were stationary inside a gray zone. The SNR threshold approach is not affected by this problem.

The 99.1% packet delivery ratio for the SNR approach is higher than what is achieved in simulation of plain AODV. It is also better compared to what we generally expect from back-of-the-envelope calculations. This indicates that there is a gain in using signal-to-noise information provided by lower layers. In our case the improvement stems from the fact that when control packets are artificially discarded the data packets may still be delivered until the route times out. This is a kind of connectivity change prediction and could be enhanced by e.g., proactively searching for new routes once an existing route's link quality drops below a certain threshold.

Chapter 6

Summary and Conclusions

This thesis presents three publications that document my experimental research within the area of wireless mobile ad hoc networks. In the first paper we demonstrated the use of active networking as a way of performing ad hoc routing. Although the active networking approach is surprisingly well suited and provides benefits like flexible routing and easy implementation, issues like performance and security remain to be more thoroughly investigated.

Our APE testbed covered in the second paper enables easy testing and comparison of routing protocols. The virtual Mobility metric enables assessment of the degree of similarity between repeated testruns. From our experiments we conclude that this method provides good grounds for such an assessment, but could benefit from being complemented with an analysis of link changes.

In the third paper we have explored a real-world problem that we call communication gray zones. An approach where we discard control packets with too low signal-to-noise quality very effectively eliminated the gray zone problem for AODV. We conclude that all ad hoc routing protocols where broadcast messages are involved in neighbor sensing must be revisited in order to investigate how gray zones affect them.

My overall conclusion is that ad hoc networking still contains important research issues to be addressed in the coming years. The importance of mature implementations and real-world experiments for protocol comparisons will increase in the next 2 to 3 years. Two crucial issues that need to be solved before ad hoc networking could be adopted by the public is easy configuration and interworking with the fixed Internet. A somewhat longer term issue is the interaction of multiple wireless technologies (e.g., IEEE 802.11, Bluetooth and GPRS) to provide seamless mobility.

In the next phase of my PhD research I will continue with the real-world evaluations and complement these studies with simulations and trace-based simulations [86]. One goal is to use the APE testbed and perform a “grand comparison” that includes all available ad hoc routing protocols.

Bibliography

- [1] G. Aggelou and R. Tafazolli, *RDMA: A Bandwidth-Efficient Routing Protocol for Mobile Ad Hoc Networks*, In Proceedings of ACM International Workshop on Wireless Mobile Multimedia (WoWMoM), August 1999.
- [2] *AODV-UU webpage*
<http://www.docs.uu.se/scanet/aodv/>
- [3] *APE testbed project webpage*
<http://apetestbed.sourceforge.net>
- [4] S. Basagni, *Distributed Clustering for Ad hoc Networks*, Proceedings of ISPAN, June 1999.
- [5] E. Belding-Royer, *Report on the AODV Interop*, UCSB Technical Report 2002-18, University of California Santa Barbara, June 2002.
- [6] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, J. Jetcheva, *A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols*, In Proceedings of ACM International Conference on Mobile Computing and Networking 1998 (MobiCom'98), October 1998.
- [7] L. Buttyan and J.P. Hubaux, *Report on a Working Session on Security in Wireless Ad Hoc Networks*, Mobile Computing and Communications Review (MC2R), Vol. 6, Number 4, 2002.
- [8] A. Campbell, H. Meer, M. Kounavis, K. Miki, J. Vicente and D. Villela, *A survey of programmable networks*, ACM SIGCOMM Computer Communications Review, vol.29, no. 2, 1999.
- [9] D. Cavin, Y. Sasson and A. Schiper, *On the Accuracy of MANET Simulators*, In Proceedings of the Workshop on Principles of Mobile Computing (POMC'02), 2002.
- [10] C.-C. Chiang, H.-K. Wu, W. Liu and M. Gerla, *Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel*, Proceedings of SICON, April. 1997.
- [11] T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, a. Qayyum et L. Viennot, *Optimized Link State Routing Protocol*, IEEE National Multi-Topic Conference (INMIC 2001), December 2001.

- [12] IEEE Computer Society LAN MAN Standards Committee. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std. 802.11-1997, 1997.
- [13] IEEE Computer Society LAN MAN Standards Committee. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std. 802.11b-1999, 1999.
- [14] *Communication Research group at Uppsala University*, www.docs.uu.se/docs/dsdc/
- [15] S. Corson and J. Macker, *Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations*, Request For Comments 2501, January 1999.
- [16] S. Das, R. Castaneda, J. Yan, and R. Sengupta, *Comparative Performance Evaluation of Routing Protocols for Mobile, Ad hoc Networks*, In Proceedings of IEEE ICCCN'98, October 1998.
- [17] D. Decaspar and C. Tschudin, *Simple Active Packet Format (SAPF)*, Active Networks Group RFC draft, Aug 1998, available from <http://www.docs.uu.se/~tschudin/pub/cft-1998-sapf.txt>
- [18] R. Dube, C. D. Rais, K. Wang, and S. K. Tripathi, *Signal stability based adaptive routing (SSA) for ad hoc mobile networks*, IEEE Personal Communication, February 1997.
- [19] K. Fall, *Network Emulation in the VINT/NS Simulator*, In Proceedings of 4th IEEE Symposium on Computers and Communications, July 1999.
- [20] L. Feeney, *A Taxonomy for Routing Protocols in Mobile Ad Hoc Networks*, SICS Technical Report T99:07, 1999.
- [21] J.J. Garcia-Luna-Aceves and M. Spohn, *Source Tree Adaptive Routing in Wireless Networks*, In Proceedings of IEEE Conference on Network Protocols (ICNP'99), September 1999.
- [22] *GloMoSim*, <http://pcl.cs.ucla.edu/projects/glomosim/>.
- [23] M. Gerla, X. Hong L. Ma and G. Pei *Landmark Routing Protocol (LANMAR) for Large Scale Ad Hoc Networks*, Internet Draft, draft-ietf-manet-lanmar-05.txt, *work in progress*, November 2002.
- [24] M. Gerla, X. Hong and G. Pei, *Fisheye State Routing Protocol (FSR) for Ad Hoc Networks*, Internet Draft, draft-ietf-manet-fsr-03.txt, *work in progress*, June 2002.
- [25] J. Gomez, A. Campbell, M. Naghshineh and C. Bisdikian, *Conserving transmission power in wireless ad hoc networks*, In Proceedings of IEEE Conference on Network Protocols (ICNP'01), September 2001.

- [26] J. Gomez, A. Campbell, M. Naghshineh, and C. Bisdikian *PARO: Supporting Dynamic Power Controlled Routing in Wireless Ad Hoc Networks*, ACM/Kluwer Journal on Wireless Networks (WINET), (to appear), 2003.
- [27] J. Haartsen, *Bluetooth - The Universal Radio Interface for Ad Hoc Wireless Connectivity*, Ericsson Review (3), 1998.
- [28] Z. Haas, M. Pearlman, P. Samar, *The Bordercast Resolution Protocol (BRP) for Ad Hoc Networks*, Internet Draft, draft-ietf-manet-zone-brp-02.txt, work in progress, July 2002.
- [29] Z. Haas, M. Pearlman, P. Samar, *The Intrazone Routing Protocol (IARP) for Ad Hoc Networks*, Internet Draft, draft-ietf-manet-zone-iarp-02.txt, work in progress, July 2002.
- [30] Z. Haas, M. Pearlman, P. Samar, *The Interzone Routing Protocol (IERP) for Ad Hoc Networks*, Internet Draft, draft-ietf-manet-zone-ierp-02.txt, work in progress, July 2002.
- [31] Z. Haas, M. Pearlman, P. Samar, *The Zone Routing Protocol (ZRP) for Ad Hoc Networks*, Internet Draft, draft-ietf-manet-zone-zrp-04.txt, work in progress, July 2002.
- [32] Y-C. Hu, A. Perrig and D. Johnson, *Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks*, In Proceedings of ACM International Conference on Mobile Computing and Networking (MobiCom 2002), September 2002.
- [33] Y-C.Hu, D. Johnson and A. Perrig, *SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks*, In Proceedings of Fourth IEEE Workshop on Mobile Computing Systems and Applications, June 2002.
- [34] J-P. Hubaux, L. Buttyan and S. Capkun, *The Quest for Security in Mobile Ad Hoc Networks*, In Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'01), 2001.
- [35] A. Iwata, C.-C. Chiang, G. Pei, M. Gerla, and T.-W. Chen, *Scalable Routing Strategies for Ad Hoc Wireless Networks*, IEEE Journal on Selected Areas in Communications (JSAC), Special Issue on Ad-Hoc Networks, August 1999.
- [36] P. Jacquet, P. Muhlethaler, A. Qayyum, A. Laouiti, L. Viennot, T. Clausen, *Optimized Link State Routing Protocol*, Internet Draft, draft-ietf-manet-olsr-07.txt, work in progress, November 2002.
- [37] P. Johanson, T. Larsson, N. Hedman, B. Mielczarek, and M. Degermark, *Scenario-based Performance Analysis of Routing Protocols for Mobile Adhoc Networks*, In Proceedings of ACM International Conference on Mobile Computing and Networking 1999 (MobiCom'99), August 1999.
- [38] D. Johnson, D. Maltz, Y-C. Hu and J. Jetcheva, *The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)* Internet Draft, draft-ietf-manet-dsr-07.txt, work in progress, February 2002.

- [39] D. Johnson, D. Maltz, J. Broch, *DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks*. In Ad Hoc Networking, edited by C. Perkins, pp. 139-172, Addison-Wesley, December 2000.
- [40] Q. Ke, D. Maltz and D. Johnson, *Emulation of Multi-hop Wireless Ad Hoc Networks* In Proceedings of the 7th International Workshop on Mobile Multimedia Communications (MOMUC2000), October 2000.
- [41] J. Kaba and D. Raichle, *Testbed on a Desktop: Strategies and Techniques to Support Multi-hop MANET Routing Protocol Development*, In Proceedings of 2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'01), October 2001.
- [42] Y. Ko and N. H. Vaidya, *Location-Aided Routing (LAR) Mobile Ad Hoc Networks*, In Proceedings of ACM International Conference on Mobile Computing and Networking (MobiCom'98), October 1998.
- [43] S.-J. Lee and M. Gerla, *AODV-BR: Backup Routing in Ad hoc Networks*, In Proceedings of IEEE Wireless Communications and Networking Conference 2000 (WCNC'00), September 2000.
- [44] S.-J. Lee, M. Gerla and C-K. Toh *Simulation Study of Table-Driven and On-Demand Routing Protocols for Mobile Ad Hoc Networks*, IEEE Network, vol. 13, no. 4, July/August 1999.
- [45] *LUNAR webpage*
<http://www.docs.uu.se/selnet/lunar/>
- [46] D. Lundberg, *Ad Hoc Protocol Evaluation and Experiences of Real World Ad Hoc Networking*, To appear in Technical Report Series, Department of Information Technology, Uppsala University, 2002.
- [47] H. Lundgren, D. Lundberg, J. Nielsen, E. Nordström, C. Tschudin, *A Large-scale Testbed for Reproducible Ad hoc Protocol Evaluations*. In Proceedings of IEEE Wireless Communications and Networking Conference 2002 (WCNC'02), March 2002.
- [48] *Mad-hoc AODV technical documentation*,
<http://mad-hoc.flyinglinux.net/techdoc.ps>
- [49] G. Malkin, *RIP version 2*, IETF RFC 2453, November 1998.
- [50] D. Maltz, J. Broch and D.B. Johnson, *Quantitative Lessons from a Full-Scale Multi-Hop Ad Hoc Network Testbed*, In Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC'2000) September 2000.
- [51] D. Maltz, J. Broch, and D.B. Johnson, *Experiences Designing and Building a Multi-Hop Wireless Ad Hoc Network Testbed*, CMU School of Computer Science Technical Report CMU-CS-99-116, March 1999.
- [52] *The Official IETF working group MANET webpage*, <http://www.ietf.org/html.charters/manet-charter.html>

- [53] M. Marina and S. Das, *On-demand Multipath Distance Vector Routing for Ad Hoc Networks*, In Proceedings of IEEE Conference on Network Protocols (ICNP'01), November. 2001.
- [54] S. Marti, T. Giuli, K. Lai, and M. Baker. *Mitigating routing misbehavior in mobile ad hoc networks*, In Proceedings of the Sixth annual ACM International Conference on Mobile Computing and Networking (MobiCom'00), August 2000.
- [55] J. Moy, *OSPF version 2*, IETF RFC 2328, April 1998.
- [56] S. Murphy and J.J. garcia-Luna-Aceves, *A Routing Protocol for Packet Radio Networks*, In Proceedings of ACM First International Conference on Mobile and Networking (MobiCom'95), November 1995.
- [57] B. Noble, M. Satyanarayanan, G. Nguyen, R. Katz, *Trace-Based Mobile Network Emulation*, In Proceedings of ACM SIGCOMM'97, September 1997.
- [58] *The Network Simulator - ns-2 webpage*
<http://www.isi.edu/nsnam/ns/>
- [59] R. Ogier, M. Lewis, F. Tempelin, B. Bellur, *Topology Broadcast based on Reverse-Path Forwarding (TBRPF)*, Internet Draft, draft-ietf-manet-tbrpf-0.6.txt, *work in progress*, November 2002.
- [60] *OLSR implementation (Inria)*
<http://menetou.inria.fr/olsr/>
- [61] *OPNET Technologies, Inc*
<http://www.opnet.com/>.
- [62] V. Park and S. Corson, *A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks*, In Proceedings of IEEE INFOCOM'97, April 1997.
- [63] M. Pearlman, Z. Haas, *Determining the optimal configuration for the zone routing protocol*, IEEE Journal on Selected Areas in Communications (JSAC), Special Issue on Ad-Hoc Networks, August. 1999.
- [64] G. Pei, M. Gerla, and T.-W. Chen, *Fisheye State Routing: A Routing Scheme for Ad Hoc Wireless Networks*, In Proceedings of ICC 2000, June 2000.
- [65] G. Pei, M. Gerla and X. Hong, *LANMAR: Landmark Routing for Large Scale Wireless Ad hoc Networks with Group Mobility*, In Proceedings of ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2000), August 2000.
- [66] C. Perkins, Ed. *Ad Hoc Networking*, Addison-Wesley, December 2000.
- [67] C. Perkins and P. Bhagwat, *Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers*, Computer Communications Review 24(4), October 1994.

- [68] C. Perkins and E. Royer, *Ad hoc On-Demand Distance Vector Routing*. In Proceedings of 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99), February 1999.
- [69] C. Perkins, E. Royer and S. Das, *Ad hoc On-demand Distance Vector (AODV) Routing*, Internet Draft, draft-ietf-manet-aodv-12.txt, *work in progress*, November 2002.
- [70] C. Perkins, E. Royer, S. Das, and M. Marina, *Performance Comparison of Two On-Demand Routing Protocols for Ad hoc Networks*, IEEE Personal Communications Magazine, Feb. 2001.
- [71] B. Plattner and J. Sterbenz, *Mobile Wireless Active Networking: Issues and Research Agenda*, IEICE Workshop on Active Network Technology and Applications (ANTA) 2002, March 2002.
- [72] R. Ramanathan and R. Hain, *An ad hoc wireless testbed for scalable, adaptive QoS support*. In Proceedings of IEEE Wireless Communications and Networking Conference (WCNC 2000), September 2000.
- [73] R. Ramanathan and M. Steenstrup, *A Survey of Routing Techniques for Mobile Communication Networks*, ACM/Baltzer Mobile Networks and Applications, special issue on Routing in Mobile Communications Networks, vol.1, no.2, October 1996.
- [74] E. Royer and C-K. Toh, *A Review of Current Routing Protocols for Ad-Hoc Mobile Networks*, IEEE Personal Communications, vol.6 no.2, April 1999.
- [75] K. Sanzgiri, B. Dahill, B. Levine, C. Shields, and E. Belding-Royer, *A Secure Routing Protocol for Ad hoc Networks*, In Proceedings of the International Conference on Network Protocols (ICNP), Paris, France, November 2002.
- [76] S. Singh and C. Raghavendra, *PAMAS - Power Aware Multi-access Protocol with Signalling for Ad Hoc Networks*, ACM Computer Communications Review, July 1998.
- [77] S. Singh, M. Woo and C.S. Raghavendra, *Power-Aware Routing in Mobile Ad Hoc Networks*, In Proceedings of ACM International Conference on Mobile Computing and Networking (MobiCom'98), October 1998.
- [78] M. Takai, J. Martin, and R. Bagrodia, *Effects of Wireless Physical Layer Modeling in Mobile Ad Hoc Networks*, In Proceedings of the 2001 ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2001), October 2001.
- [79] D. Tennenhouse, J. Smith, W. Sincoskie, D. Wetherall and G. Minden, *A survey of active network research*, IEEE Communications Magazine, vol 35, January 1997.
- [80] D. Tennenhouse and D. Wetherall, *Towards an active network architecture*, Computer Communication Review, vol 26, April 1996.

- [81] C-K. Toh *Ad Hoc Mobile Wireless Networks: Protocols and Systems*, Prentice Hall, 2002.
- [82] C.-K. Toh, *Associativity-Based Routing for Ad-Hoc Mobile Networks*, Wireless Personal Communications Journal, vol. 4, no. 2, March 1997.
- [83] C-K. Toh, Minar Delwar, and Donald Allen, *Evaluating the Communication Performance of an Ad Hoc Wireless Network*, IEEE Transaction on Wireless Communications, July 2002.
- [84] C. Tschudin, *On the Structuring of Computer Communications*, PhD thesis, Geneva University, 1993.
- [85] *UMD TORA/IMEP implementation*,
<http://www.cshcn.umd.edu/tora.shtml>
- [86] B. Wiberg, *Porting AODV-UU Implementation to ns-2 and Enabling Trace-based Simulation*, To appear in Technical Report Series, Department of Information Technology, Uppsala University, 2002
- [87] Wikipedia - The Free Encyclopedia, *Ad Hoc Protocol List, initiated by David Lundberg*,
http://www.wikipedia.org/wiki/Ad_hoc_protocol_list
- [88] J. Zander and R. Forchheimer, *SOFTNET - an approach to high-level packet communication*, In Proceedings of 2nd ARRL Amateur Radio Computer Networking Conference, 1983.
- [89] M. Zapata and N. Asokan, *Securing Ad hoc Routing Protocols*, In Proceedings of Workshop of Wireless Security (WiSe), September 2002.
- [90] Y. Zhang and W. Lee, *Intrusion detection in wireless ad hoc networks*, In Proceedings of the Sixth annual ACM International Conference on Mobile Computing and Networking (MobiCom'00), August 2000.
- [91] Y. Zhang and W. Li, *An Integrated Environment for Testing Mobile Ad-Hoc Networks*, In Proceedings of 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'02), June 2002.
- [92] L. Zhou and Z.J. Haas, *Securing Ad Hoc Networks*, IEEE Network Magazine, vol. 13, no.6, November/December 1999.

Paper A

Christian Tschudin, Henrik Gulbrandsen and Henrik Lundgren, *Active Routing for Ad-hoc Networks*, Published in IEEE Communications Magazine, Special issue on Active and Programmable Networks, April 2000.

© 2000 IEEE. Reprinted with permission.

Active Routing for Ad Hoc Networks

Christian Tschudin and Henrik Lundgren, Uppsala University

Henrik Gulbrandsen, Ericsson Research

ABSTRACT

Ad hoc networks are wireless multihop networks whose highly volatile topology makes the design and operation of a standard routing protocol hard. With an active networking approach, one can define and deploy routing logic at runtime in order to adapt to special circumstances and requirements. We have implemented several active ad hoc routing protocols that configure the forwarding behavior of mobile nodes, allowing data packets to be efficiently routed between any two nodes of the wireless network. Isolating a simple forwarding layer in terms of both implementation and performance enables us to stream delay-sensitive audio data over the ad hoc network. In the control plane, active packets permanently monitor the connectivity and setup, and modify the routing state.

INTRODUCTION

Ad hoc networking is a very challenging and still open research field for the design of routing protocols. The connectivity among mobile devices is constantly changing and leads to permanent adaptation and reconfiguration of the routing state. The Mobile Ad Hoc Networking (MANet) group of the Internet Engineering Task Force (IETF) has generated and is reviewing several proposals for ad hoc routing protocols, but consensus is difficult to achieve [1]. Our basic hypothesis is that there is no best ad hoc routing protocol suitable for all circumstances. Too many constellations, link characteristics, quality of service (QoS) requirements, and security concerns would need to be considered. We therefore suggest using an active networking (AN) approach where a multitude of routing protocols can be chosen among and run in parallel. We observe that first, the AN approach has no software deployment problem and allows us to experiment with different active routing protocols in an easy way. Second, adaptive deployment of routing protocols enables us to choose the best protocol candidate depending on the network's current topology, traffic patterns, and so on. Third, AN supports customization of routing protocols: instead of relying on a common routing service, applications can run their own routing protocol, actually creating a private virtual (routing) network.

In this article we report on our testbed for active routing protocols and on a demo applica-

tion featuring the streaming of time-sensitive audio data. We discuss the network architecture for active (ad hoc) routing. We go into the details of an active routing protocol called *active cellular IP* and later report on the implementation and demo experiences.

This work was done in the Active Routing and Resource Control for Ad Hoc Networks (ARRCANE) [2] research project funded by NUTEK (project #P11766) and Ericsson Research, SwitchLab.

A FORWARDING ARCHITECTURE FOR ACTIVE ROUTING

Routing protocols in the Internet are part of the "hidden" activities going on in the network. Their task is to collect connectivity information and to set the forwarding state in all Internet nodes in order to provide end-to-end connectivity. Routing is a common service available to and shared by all applications, wherefore access to routing state is critical: control over routing protocols, and their configuration and management, is the responsibility of the network managers.

Active networking proposes to open the interface to network nodes. In the capsule approach, each data packet is replaced by an active packet that contains instructions on how to process the packet's payload. Ideally, capsules can be self-routing, making them independent of shared routing state.

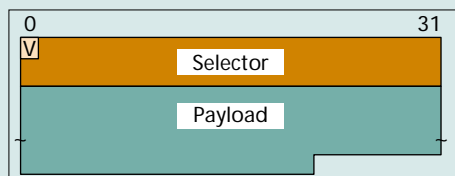
A middle ground between private routing at the capsule level and common forwarding via the route table consists of partitioning the routing state. Active packets can create private forwarding entries through which passive data packets are routed without any active packet processing overhead. Applications are then free to either choose "standard" forwarding services or the private virtual routing world controlled by active packets. In fact, what we need is a programmable infrastructure supporting different *network personalities* that share the route table resource.

THE SIMPLE ACTIVE PACKET FORMAT

In order to distinguish packets belonging to different private "forwarding circuitry," we need an additional packet header field. One possibility is to add a new protocol layer at the application level and rely on the router's forwarding engine to consider this field when forwarding the packet. However, we chose to go to the lowest level

SAPF

The simple active packet format defines a 64-bit header consisting of a single header field, selector, without any further options (the version bit *V* is an ultimate escape door just in case). Length information, for example, is assumed to be provided by the lower layer.



SAPF defines a forwarding abstraction that relies solely on the selector field, quite similar to tag switching where a packet's tag directly identifies (indexes) the packet's context and thus allows fast forwarding. Other benefits are the possibility to demultiplex packets directly to the application or using selectors as header compression identifiers, thus demultiplexing to the compression context. SAPF is an "active" packet format because it requires additional protocol functionality — assumed to be provided by active protocols — in order to negotiate SAPF selector values and install forwarding state in a remote node.

possible, and to add a single field between link and networking layer. According to this view, IP forwarding becomes just one network personality among others.

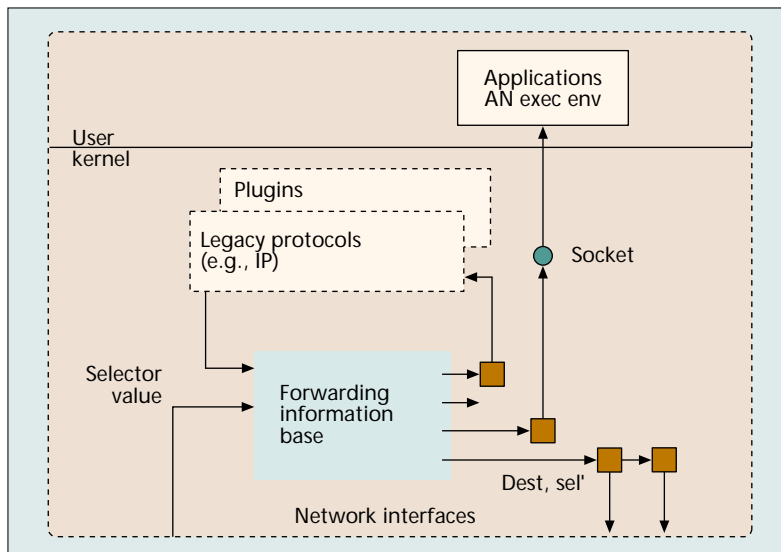
The header format used for demultiplexing different networking personalities is the simple active packet format (SAPF) [3]. It features a single *selector* field based on which the router can make fast forwarding decisions (see the text box on SAPF). IP traffic, for example, is tunneled inside SAPF packets; the corresponding SAPF selector merely points to the IP forwarding module (see Fig. 1 for a schematic block diagram of an SAPF-enabled router node). Active packets are also carried in SAPF packets, but their selectors point to their respective execution environment. Note that it is also possible to place SAPF headers at higher levels in the protocol hierarchy by using a tunneling approach, for example, in situations where there is no possibility of introducing a new protocol at the link layer.

The SAPF packet format fits the hybrid router approach well where a fast forwarding plane for passive packets is controlled by asynchronously processed active signaling packets [4]. In fact, SAPF aims to reduce the forwarding layer to a minimal set of functionality, leaving all configuration aspects to active packets.

Routing at Level 2.5 — Placing the SAPF demultiplexing layer below the IP stack makes sense for high-speed active networking because:

- Forwarding is streamlined [5].
- Bandwidth-consuming IP headers can be avoided for applications that use native SAPF packets.

IP tunnels can also be made bandwidth-efficient if header compression is used, in which case the SAPF selector plays the role of a compression context identifier. Surprisingly enough, locating



■ Figure 1. The SAPF node architecture is centered around the forwarding information base.

SAPF at a low level also makes sense for ad hoc networks. Bandwidth savings is an argument here too. The other reason is that IP routing protocols behave poorly in a highly dynamic ad hoc environment [6], so it is better to fool IP about the ad hoc situation and do routing at "level 2.5." This creates the illusion of a classical IP subnetwork for which IP routing need not become active.

SAPF Operation — The behavior of the SAPF forwarding layer is defined by the content of the nodes' forwarding information base (Fig. 1). The selector of an incoming packet is used as an index into the forwarding information base (FIB), where different entry types can be found. Plain forwarding (FWD) entries let a packet be copied to one or more remote nodes, possibly rewriting the SAPF selector if requested. Entries can also point to operating-system-dependent data structures like sockets, from which applications can read the packet's payload. A third possibility is to let FIB entries point to special packet processing software (e.g., an IP stack) or special router plugins that process a packet before it is handed back to the SAPF forwarding engine. Finally, FIB entries can also be "branch-and-forward" instructions which conditionally forward packets along two different paths. This FIB entry type will be discussed later when describing the routing procedure under control of the active ad hoc routing protocol.

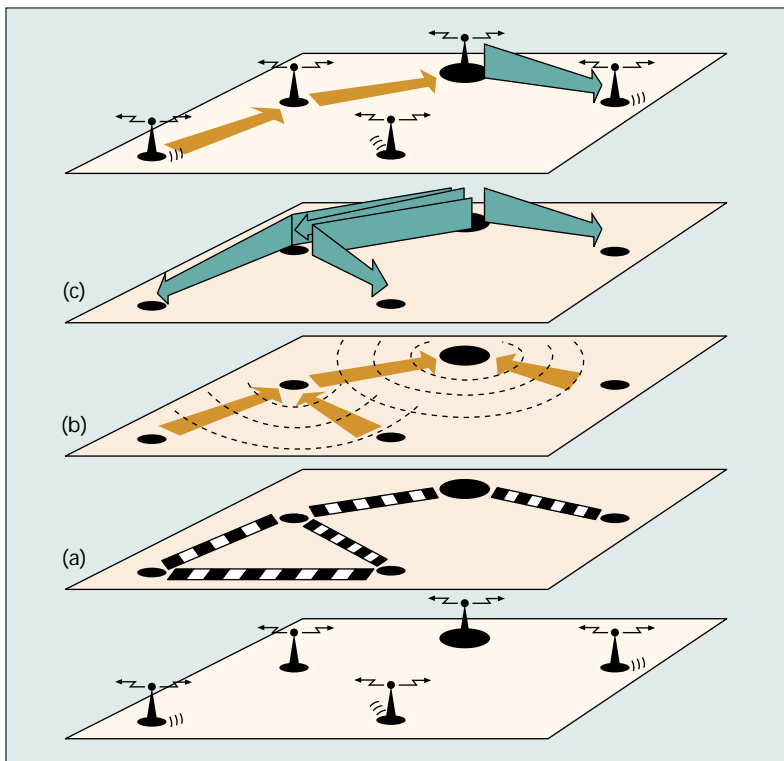
Entries in the FIB are linked to an expiration time, after which the entry will be removed by the system (soft state). An exception to this rule are "delivery sockets" which are only removed when the socket is destroyed.

The semantics of the SAPF layer is that of forwarding (and possibly duplicating) datagrams. This forwarding can be redefined and redirected in arbitrary ways. Thus, unlike UDP, where the datagram carries an end node address, an SAPF header only selects the entry in the next downstream node. Whether this entry contains a list of sockets or forwarding and header rewriting

CODE ON DEMAND IN AN AD HOC NETWORK?

Ad hoc networking challenges assumptions that are often implicitly made when routing is considered. This directly affects those active networking platforms that provide routing-dependent forwarding functionality.

Consider the “code-follows-capsule” approach, used, for example, in ANTS [10], where a node that does not know about a new capsule’s code has to request the code from the sending node. This scheme may fail in an ad hoc network because wireless links, unlike most links in common wireline networks, are not necessarily bidirectional. We are using active packets that have an explicit code portion (messengers). In the case of the three active subprotocols described in this article we made each active packet fully self-contained (i.e., it contains all instructions required for its processing). We can also use “code-precedes-capsule” (i.e., download and cache some code portion ahead) in order to minimize the code sent with every active packet.



■ Figure 2. Routing paths from one mobile device to another (see top plane) are the side effect of three different and concurrent active protocols that build on each other: a) neighbor discovery; b) minimum spanning tree; c) delivery paths.

data, or points to some other protocol stack is not specified. The SAPF header provides late-binding semantics to forwarding: forwarding behavior is defined at runtime by active packets that put state into the FIB.

ACTIVE ROUTING IN AN AD HOC NETWORK

The task of active routing packets for ad hoc networks is to set up virtual “paths” quite similar to a connection. These paths are not connections in the classical sense, which would behave like privately owned virtual links between two applications. Instead, paths in an ad hoc network are more like highways where cars can jump in and out as long as a path leads to the right destination. A major question, whose answer depends on the actual network topology, is how these paths should best be arranged.

For example, a path network can be organized in a star topology. From the center (imagine Paris) ray-like *delivery paths* exist toward each possible destination. There are also paths from each node toward the center. For routing, data packets are sent toward the center where they will be redirected to the desired delivery path. Alternatively, we can imagine a fully decentralized configuration where each node is the center of a separate star network from where delivery paths stretch out to each node in the network.

The highway analogy seems to imply that the delivery path infrastructure is permanently available, that is, active packets *proactively* maintain full routing connectivity all the time. However,

there are also reactive approaches to ad hoc routing where a path is only created when there is actual demand for data delivery (e.g., AODV [7]). We have implemented both types of active routing protocols, but concentrate in the following on a proactive variant called *active cellular IP* (ACIP). First, we briefly review the cellular IP protocol which inspired the design of ACIP.

CELLULAR IP

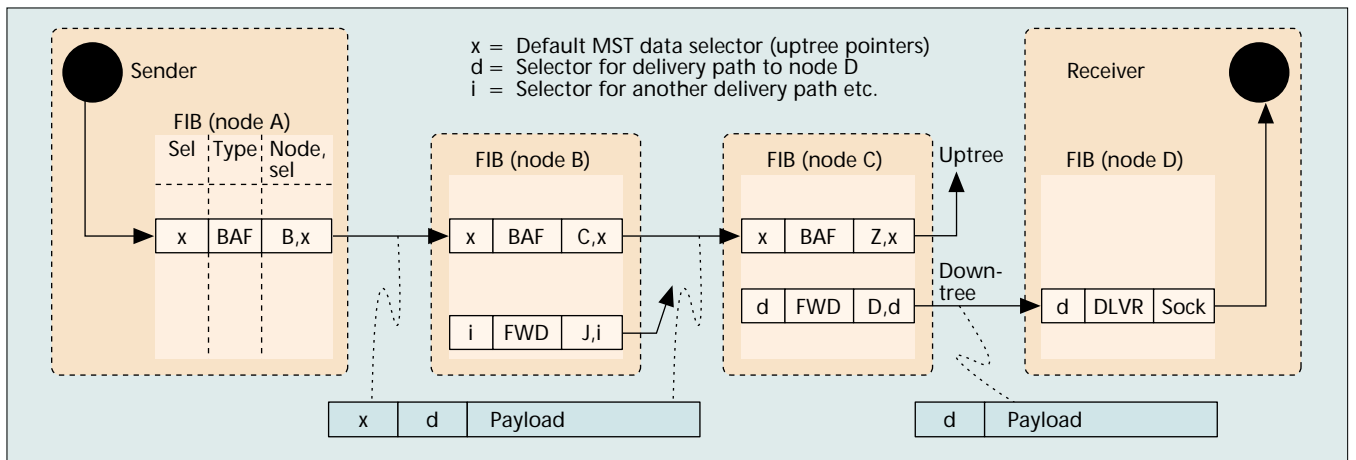
Cellular IP (cIP) is not an active protocol, nor is it a routing protocol for ad hoc networks; but it has some novel elements that make it attractive and easy to translate into an active version.

cIP is an access protocol that allows wireless IP devices to do fast handoffs from one base station to another [8]. Base stations are statically arranged as a logical minimum spanning tree (MST): a mobile device has to periodically send location information to the base station to which it wishes to attach. This information is propagated up the tree until it reaches the MST root node. During this back propagation, device-specific path information for delivery is put into each routing table. Handoff is simply achieved by letting the mobile device send its location information to another base station.

For ACIP we use a similar division of labor. An MST provides a default core of routing paths; mobile nodes are responsible for keeping delivery paths alive. Note, however, that we have no base stations in the ad hoc network. Mobile devices do not attach to leaf nodes; instead, they are part of the MST and can find themselves in any position in this tree. The layout of the MST depends on the reachability among nodes of the ad hoc network: it has to be dynamically recomputed all the time.

ACTIVE CELLULAR IP

ACIP is an activated version of cIP; the main difference is that all routing state is managed by active packets. Active packets are used to create the MST, and a different set of active packets is



■ **Figure 3.** Routing via branch-and-forward and forward entries in the SAPF FIBs: node C is the first node where we cross the destination's delivery path; hence, packets will travel downtree from this node on.

responsible for creating the delivery paths from the MST root to each possible destination (Fig. 2b and c). One addition to cIP is a neighbor discovery service, which adds a third active packet type to the protocol. In fact, ACIP is a combination of three different subprotocols that we describe in turn.

Neighbor Discovery — Any routing protocol ultimately depends on connectivity information. The active discovery packets collect this information and store it on neighbor nodes. In fact, our neighbor discovery packets are more complex, since they also announce interface names and link status information.

Each node in the ad hoc network periodically broadcasts an active *beacon* packet whose task is to announce the node's presence. This is done by putting information in the persistent data storage area of all neighbor nodes. The most important information is the sending node's and outgoing interface's names. At arrival, the beacon packet will also add a timestamp (expressed in local time), making it possible to reason about the freshness of the stored data.

Besides basic originator-related data, each active beacon packet also copies the accumulated connectivity information that was imported by other beacon packets. The net effect is that a node learns its full two-hop neighborhood for free. Most important, we learn which interface and address have to be used on a remote node in order to create a simple round-trip active packet. All data copied to a remote place include timestamp information; in order to avoid clock synchronization problems, time values are converted to a relative format (i.e., age) and, at the remote node, converted back to an absolute time value according to the local clock.

Minimum Spanning Tree — ACIP's MST is constructed with a wave algorithm. The designated root node periodically broadcasts an active MST packet which, on each node, leaves a footprint and rebroadcasts itself. Before rebroadcasting, however, it checks for existing footprints meaning that another MST active packet already arrived at this node, in which case the packet

terminates. As the wave spreads and advances, two SAPF back pointers are installed on each node in the form of entries in the FIB. One back pointer points to the uptree active execution environment, creating an MST *control* path toward the root node. The active delivery path packets described below make use of this path in order to find their way to the next node toward the MST root. The second back pointer points to the same uptree node, but uses a special branch-and-forward (BAF) entry. This creates an MST *data* path toward the root node which will be used for passive data packets. How routing is actually done is described further below.

The root of the MST is statically configured and, in our case, is also the gateway to the fixed network. It is possible to write active MST packets that decide at runtime which is the optimal node to assume this role. Decisions could be made to, for example, minimize the tree depth, or favor special nodes as in our case, where we assume that quite some traffic will go through the gateway anyway. It is also conceivable to add robustness to the MST subprotocol such that two MSTs would form in case of a network partition, and two MSTs would gracefully merge in case of partition healing [9]. Such changes to the MST subprotocol can be introduced without changing or restarting the two other active packet populations. We repeatedly exploited this protocol composition feature when we replaced the MST implementation after bug fixes, for example.

Most operations of the MST packets do not need connectivity information since the wave automatically gathers this information during its expansion phase. However, the MST packets can make use of the information collected by the neighbor discovery packets in special circumstances. More specifically, when existing back pointers should be changed, we check connectivity information. Only if the signal strength of the potential new link is substantially bigger than the signal quality of the existing link will we accept the establishment of new MST pointers. This avoids changing the MST layout because of a spuriously lost MST packet and also provides an elegant handover functionality.

MP3 STREAMING AND SAPF

MP3 is the layer 3 audio encoding scheme of the MPEG-1 and 2 digital media stream standards. MP3 streams have a constant bit rate requirement; 128 kb/s is most commonly used for stereo hifi quality. The constant bit rate makes it easy to compute the send deadline of an MP3 packet. Our sender-paced MP3 streamer looks at the bit rate defined in the MP3 audio file and combines it with each audio frame size in order to determine the frame's send time. MP3 frames are put directly into an SAPF packet. The SAPF header specifies which forwarding entry applies to this packet and, at the destination node, to which application the packet has to be delivered.

The Delivery Path — End nodes must establish a data delivery path from the root node to themselves. To this end they periodically send out active delivery path (DP) packets along the MST control pointers so that they reach the active execution environment of the uptree node. At each node they install forwarding entries pointing to where the DP packet came from. See plane c of Fig. 2 for the resulting paths.

Routing of Passive Packets — Routing of passive data packets in ACIP is a two-step procedure. First, the sender has to prepare a data packet with two SAPF headers back to back (Fig. 3). The outermost header selects the MST data path. The second SAPF header contains the destination-specific address. Since SAPF only defines selectors, not addresses, we use the selector of the delivery path that leads to the desired destination as an address.¹

The two-header SAPF packet is injected into the MST data path. On each node, the special BAF entry in the routing table will examine the second header. If the FIB contains an entry that matches the packet's second header, we will drop the first header whose function was to select the BAF entry and reforward the trimmed packet according to the found table entry. This means that the two-header data packet will travel uptree until the first time it hits the destination's dntree delivery path (see also the example in the top plane of Fig. 2) from where it continues with a single SAPF header.

IMPLEMENTATION

We built an active routing testbed which consists of Wavelan-equipped PC notebooks running Linux. SAPF was implemented as a kernel module and provides the CPU isolation essential for the forwarding of delay-sensitive audio streams. The test application consisted of running one or more MP3 streamers for the generation of audio streams, as well as MP3 players that listen on their specific SAPF selector for incoming MP3 frames (see the text box). The three ACIP sub-protocols run in the background and establish the forwarding paths between sender and receiver nodes.

MOBILE BASH

As the programming language for our active signaling packets we chose a rather crude approach which consists of shipping standard UNIX bash scripts (bash: "Bourne again shell"). A simple execution environment was written that waits on an SAPF socket with a well-known selector value

for the incoming bash script. Each received "mbash" script is executed by a separate UNIX process, and the UNIX file system is used as the persistent data storage area. The data storage area is subject to a softstate policy wherein a background task periodically scans the data area and removes files that have not been touched for a fixed time period. Mbash packets are transported in compressed form and decompressed before being started at a remote node.

MBASH PERFORMANCE

In order to test the overhead of shipping, receiving, and interpreting bash scripts, we wrote a simple active packet for measuring the round-trip time. The following list explains the resulting sequence of actions:

- At the sender site we fetch the local time. An active packet is composed that contains the local time, the return address, and a return script.
- The packet is compressed and sent out.
- The packet is received at the remote site and decompressed.
- A UNIX shell is started that executes the bash script. This script composes a return packet which is compressed and sent back.
- Back at the originating site, the return packet fetches the local time and computes the difference, storing the result in a file.

Using 366 MHz Pentium II laptops equipped with 2 Mb/s Wavelan cards, we obtained round-trip times between 40 and 60 ms. This is sufficiently fast to track topology changes and do active routing in an ad hoc environment.

EXPERIENCES AND DISCUSSION

The streaming of MP3 works very well. Because active mbash packets run in user space, SAPF forwarding at the kernel level does not suffer from competition for active packet processing time. The reaction timescale of 20–80 ms gives satisfactory results. Reducing it further would not change things because the forwarding gaps during ad hoc network reconfigurations are mainly determined by the frequency of MST and delivery path packets (which were in the 1–3 s range). Fading problems can easily be produced by walking around, resulting in packet loss although no routing configuration change is happening. Moving to smaller audio packet sizes may help reduce the packet loss rate due to fading.

The active packets used for ACIP are rather small: the neighbor discovery, MST, and delivery path packets are all in the range of 50–100 lines each, which includes the instructions to periodically send these packets out. On the wire, these compressed mbash scripts translate to a few hundred bytes.

A hard problem is *old* active packets, which can easily occur if a node's OS thrashes. It is rather difficult to make active packets aware of their own age and the rate of their progress, since these activities also use CPU time and cannot be guaranteed to be carried out in some fixed amount of time. The symptoms of this are that old active packets apply changes to the routing state based on old network conditions. This makes the design and implementation of self-stabilizing routing algorithms a difficult task.

¹ We assume that there is a mapping service that translates destination names to selector values. Our current approach is to apply a cryptographic hash function to the node's name and take the lower 63 bits of the resulting hash value as this node's "address."

We will need explicit support from the active execution environment:

- For dropping packets that become too old
- For providing execution quanta of guaranteed CPU time

Because of the inherently parallel execution of active protocols, it is easy to extend a running system. For example, we wrote a simple topology discovery tool (JARRCVIEW) based on a wave algorithm: Mbash scripts collect all reachability information available in the ad hoc network as well as delivery path data. The gathered information is fed to a Java application which continuously maps this data in graphical form.

CONCLUSIONS

Separating active signaling from the task of forwarding leads to novel network architectures that are well suited to high-speed *and* ad hoc networks. Because complexity in this hybrid network architecture is moved to the active packet layer, we can radically simplify the forwarding layer.

In ad hoc routing each node becomes a router. The challenge is to write active routing packets that consistently set the state of the mobile SAPF forwarding engines. Our active cellular IP (ACIP) routing protocol consists of three different active subprotocols of rather small size: with a total of approximately 200 lines of mobile bash scripts we were able to implement an ad hoc routing service that allows streaming of delay-sensitive audio data. The routing world created by these active packets can be called a "network personality" analogous to the term used in the operating system world. Other network personalities, such as the IP protocol suite or customized virtual private networks, can also be mapped to this hybrid router architecture.

An important implementation aspect of the architecture induced by the SAPF packet format is the ability to isolate the network personalities: for this, packet queues and scheduler configuration must be made accessible to active packets. Work in this direction is underway and will allow us to experiment with the important class of QoS-aware routing protocols for ad hoc networks.

ACKNOWLEDGMENTS

We would like to thank P. Gunningberg, Uppsala University, and P. Johansson, T. Larsson, and C. G. Perntz from Ericsson Research, Switchlab, for their active support and advice for this project.

REFERENCES

- [1] IETF MANet Working group: <http://www.ietf.org/html.charters/manet-charter.html>
- [2] ARRCANE Research Project, Uppsala Univ. and Ericsson Research Switchlab: <http://www.docs.uu.se/arrcane>
- [3] D. Decaspar and C. Tschudin, "Simple Active Packet Format (SAPF)," Active Networks Group RFC draft, Aug. 1998: <http://www.docs.uu.se/~tschudin/pub/cft-1998-sapf.txt>
- [4] B. Braden, "Active Signaling Protocols," Dec 1997: ftp://ftp.isi.edu/rsvp/active_signaling/ASP_overview.ps
- [5] T. Wolf, D. Decasper, and C. Tschudin, "Tags for High Performance Active Networks," *OpenArch 2000*, Mar. 2000.
- [6] P. Johansson *et al.*, "Scenario-based Performance Analysis of Routing Protocols for Mobile Ad Hoc Networks," *Proc. Mobicom '99*, Seattle, WA, Aug. 1999.
- [7] C. Perkins and E. Royer, "Ad Hoc On Demand DistanceVector (AODV) Routing," Internet draft, Oct. 1999: draft-ietf-manet-aodv-04.txt

- [8] A. Campbell *et al.*, "Cellular IP," Internet draft, draft-valko-cellularip-01.txt, Oct. 1999.
- [9] C. Tschudin, "A Self-Deploying Election Service for Active Networks," *Proc. 3rd Int'l. Conf. Coordination Models and Languages*, Amsterdam, the Netherlands, LNCS 1594, Apr. 1999.

ADDITIONAL READING

- [1] D. Wetheral, G. Gutttag, and D. Tennenhouse, "ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols," *OpenArch '98*, Apr. 1998.

BIOGRAPHIES

CHRISTIAN TSCHUDIN (tschudin@docs.uu.se) is an associate professor at Uppsala University and has worked in Geneva and Zurich, Switzerland, and Berkeley, California, on active networking since the early 1990s.

HENRIK LUNDGREN (henrik@docs.uu.se) is a Ph.D. student at Uppsala University and works on the ARRCANE project. More specifically he is looking into the problems of porting existing ad hoc routing protocols to their active form.

HENRIK GULBRANDSEN (m93hgu@student.tdb.uu.se) works for Ericsson Research on the QoS support for the ARRCANE infrastructure and is at the same time a Master's thesis student at Uppsala University.

An important implementation aspect of the architecture induced by the SAPF packet format is the ability to isolate the network personalities: For this, packet queues and scheduler configuration must be made accessible to active packets.

Paper B

Henrik Lundgren, David Lundberg, Johan Nielsen, Erik Nordström and Christian Tschudin, *A Large-scale Testbed for Reproducible Ad hoc Protocol Evaluations*, In Proceedings of The Third Annual IEEE Wireless Communications and Networking Conference 2002 (WCNC 2002), March 2002.

A Large-scale Testbed for Reproducible Ad hoc Protocol Evaluations

Henrik Lundgren, David Lundberg, Johan Nielsen, Erik Nordström, Christian Tschudin

Abstract—We have built an Ad hoc Protocol Evaluation testbed (APE) in order to perform large-scale, reproducible experiments. APE aims at assessing several different routing protocols in a *real-world* environment instead of by simulation. We present the APE testbed architecture and report on initial experiments with up to 37 physical nodes that show the reproducibility and scalability of our approach. Several scenario scripts have been written that include strict choreographic instructions to the testers who walk around with ORiNOCO equipped laptops. We introduce a mobility metric called *Virtual Mobility* that we use to compare different testruns. This metric is based on the *measured* signal quality instead of the geometric distance between nodes, hence it reflects how a routing protocol actually *perceives* the network’s dynamics.

Index Terms — Ad hoc networking, routing protocols, protocol evaluation, routing testbed, implementation benchmarking.

I. INTRODUCTION

AD HOC networks consist of a number of wireless mobile nodes that dynamically form a network without any pre-existing communication infrastructure. As the nodes may move arbitrarily and unpredictably, the network’s topology will be subject to constant changes. Special routing protocols are required in order to coordinate the forwarding of data packets sent by neighboring nodes. Conventional routing protocols used in the Internet today perform poorly under such circumstances.

There exist several ad hoc routing protocol proposals both inside and outside the IETF WG MANET [1]. Some of these proposals are evaluated through simulations like in [2], [3] but only a few of them through (small-scale) tests [4], [5], [6], [7]. We believe that after so many years of study it is time to expose these routing protocols to real environments. Our Ad hoc Protocol Evaluation testbed (APE) specifically aims at experiments at larger scale and enables a direct comparison of routing protocols by ways of the controlled reproducibility of testruns.

Previous testbed projects report on experiments with 10 nodes or less. In [5], [6], [7] they examine performance using only stationary settings (mobility is emulated in [6] by removing the wireless adapter). In [5], [6] they use off-the-shelf hardware while specific hardware is used in [7]. In [4] they use a mix of both stationary and mobile nodes and point out the problems of real-world, outdoor radio propagation. None of these testbeds report on the possibility to run different protocols or reproducing mobile testruns.

Scaling to larger node numbers is achieved in the APE testbed by making the installation as easy as possible, even for settings with 50 mobile nodes and persons; per-node choreography instructions and internal scripts enable to specify arbitrary

mobility patterns and traffic conditions. With the APE’s modular architecture we can alter kernel versions, routing protocols, and traffic generators.

Our approach with a strict choreography, combined with a special “virtual Mobility metric” which is derived from measured signal quality, allows us to create verifiably reproducible experiment settings. Our long term goal is assess statements like “under some network condition characterized by Virtual Mobility v , protocol A behaves better than protocol B”. However, today such a comparison is beyond the scope of this paper due to the quite limited availability of working implementations of ad hoc routing protocols. Nevertheless, we report in this paper on our experiences and insights from first experiments with up to 37 nodes.

The rest of the paper is outlined as follows. The architecture and implementation of the testbed is presented in Section II. Section III describes the computation of the virtual mobility metric. In Section IV the physical environment is described together with three tested scenarios. Section V discusses the analysis of the experiments. Finally, we conclude and outline future work in Section VI.

II. APE ARCHITECTURE AND IMPLEMENTATION

The APE testbed comprises tools for choreography configuration, network interface data collection, uploading of all measurement results to a central machine as well as post-run analysis and chart generation. The testbed is based on Linux; any routing protocol implemented under Linux can be inserted into the testbed. We have successfully run the APE testbed with three different routing protocols; AODV [8], [9], OLSR [10], [11] and TORA [12], [13].

Figure 1 shows an overview of the APE software that runs on each node during a testrun. The choreography script interpreter parses scenario files (see Figure 2) and controls the startup of traffic generators and the time synchronization application (TSB). A modification to Lucent Technologies’ ORiNOCO driver [14] enables to log signal qualities from *all* packets that a node can detect.

A. Choreography and Data Traffic Generation

The scenarios are strictly choreographed; after the initial “ready-set-go” testbed participants only need to follow the instructions that appear on the screen on when and where to move.

Different traffic generators can be inserted independently and would then simply be launched from within the choreography script. Normal Ping, for example, generates useful traffic as we can record packet routes, end-to-end delay, and packet loss. Other types of traffic in the network are beaconing traffic from the time synchronization applications and from the routing daemons.

H. Lundgren, D. Lundberg, E. Nordström and C. Tschudin are with the IT Department, Uppsala University, Sweden. E-mail: henrikl@docs.uu.se
J. Nielsen is with Ericsson Research/Switchlab, Stockholm, Sweden.

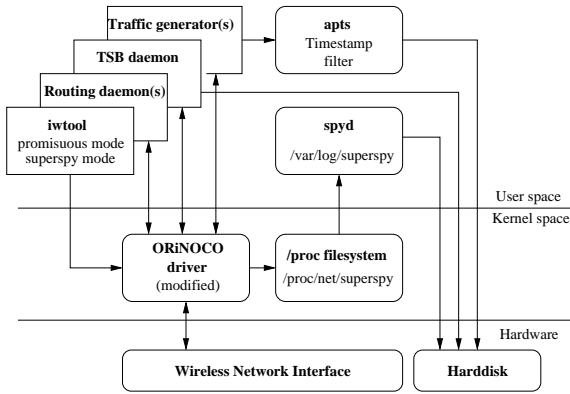


Fig. 1. Overview of software components running on each node in the APE testbed.

```

node.0.action.0.msg=Test is starting...
node.0.action.0.command=start_spyd
node.0.action.0.duration=1
node.0.action.1.command=ping_node 1 660
node.0.action.1.msg=Stay at this location.
    You are pinging node 1. (30sec).
node.0.action.1.duration=30
node.0.action.2.msg=Start moving! Go to the
    end of House 1 (DoCS). (75 sec)
node.0.action.2.duration=75

```

Fig. 2. Excerpt from the scenario file for a Double Lost'n'Found scenario. (see Section IV-B for scenario description)

B. Data Gathering Tools

Data is time stamped and logged both at the Ethernet and the IP layer. Using a user-space utility, the ORiNOCO driver is set in promiscuous mode to enable our spy routine. We get the MAC address from the Ethernet frame, and signal and noise levels from ORiNOCO driver. Each node logs the result of its own IP traffic. After an experiment the log files are directly uploaded by the participants to a mobile “collect node”.

C. Data Analysis Tools

Sorting and synchronization scripts are used to merge all log files and to adjust them for clock skews. This results in single log files for the Ethernet as well as for the IP packet level where all entries are globally ordered in time. Analysis scripts enable to extract a testrun’s salient features as for example the virtual mobility metric, connectivity in terms of number of neighbors, link changes per second, packet loss and hop count.

A log-driven animation tool called APE-view was written that allows to replay scenarios by positioning the nodes on the screen based on the logged signal quality between node-pairs (see also Section V-A.2 and the figure at the end of this paper).

D. Testbed Distribution

Due to the large number of nodes and persons involved, the configuration of each mobile node must be very simple. We assume that participants download on an experiment basis a pre-made software package. The package to install is identical for everybody and can be used on a host computer running either the Windows98 or Linux operating systems. No harddisk

re-partitioning is required: under Windows it suffices to execute a self-extracting file which makes the computers reboot to Linux, reading and writing to a filesystem image included in the distribution. Under Linux, participants only need to run a single script that adds a boot option in the Linux boot loader and makes the computers reboot to the testbed environment. A set of special “software package fabrication programs” allows us to easily generate packages for different routing protocols, Linux kernels and choreography scripts.

III. CAPTURING MOBILITY IN AD HOC NETWORKS

A mobility metric is useful for characterizing the state of the network and its dynamics. Geometric mobility metrics like those described in [3], [15] require that the nodes’ physical positions are known at all times. Real-world settings require GPS or triangulation techniques to achieve this in outdoor or indoor environments, respectively. Looking at route changes due to link breakages is the basis of another mobility metric proposed in [15]. More (and less binary) information can be obtained by monitoring the link quality, as we will show below.

A. A “Virtual Mobility” Metric

Background noise, obstacles and even small movement can influence the signal quality. Instead of using the real distance between two nodes, we propose to compute a “virtual distance” from the perceived signal quality. This metric captures the real world dynamics as perceived by the nodes. In this section we define this new mobility metric; in Section V we show that this metric gives a “fingerprint” of the link quality changes that characterizes a testrun.

1) *Path Loss Model*: The definition of our mobility metric relies on “virtual” distances which we base on the *measured* signal quality. The path loss model mentioned in [16] is used to relate signal quality and distance: for the far field case (indoor) it proposes a path loss coefficient of 3.3:

$$Q \text{ in dB} = \alpha - 33 * \log(\text{dist}/\beta) \quad (1)$$

which, after calibration with the signal quality range of the ORiNOCO card and our measurements, results in the following inverse path loss formula:

$$D_j(\text{node}_i) = 4 * 10^{\frac{40 - 0.9 * Q_j(\text{node}_i)}{33}} \quad (2)$$

where Q is the signal quality (0...75) for a packet received from node j at node i . D is in the range of 0.5 to 65 m. We only considered the far field model as our experiments focus on large movements and long distances with nodes going out of reception range.

2) *Calculating vM*: Virtual Mobility between node_i and node_j is calculated for node_i as follows. For a given time interval t_k we average the virtual distances obtained from all packets heard from a specific node_j . We define D_j^k , the mean virtual distance to node_j for time slot t_k , as

$$D_j^k(\text{node}_i) = \frac{1}{P_j^k} \sum_{a=1}^{N_j^k} D_j^a \quad (3)$$

where P_j^k is the number of packets received from $node_j$ during t_k and D_j^a is the virtual distance obtained from the signal quality of packet a that was received from $node_j$ during interval t_k .

The virtual mobility vM for $node_i$ with respect to $node_j$ for time interval t_{k+1} is simply the change in mean virtual distance, namely

$$vM_j^{k+1}(node_i) = |D_j^{k+1}(node_i) - D_j^k(node_i)| \quad (4)$$

and the average virtual mobility perceived by $node_i$ at time t_k is

$$vM_{avg}^k(node_i) = \frac{1}{S} \sum_{l=1}^S vM_l^k(node_i) \quad (5)$$

where S is the number of nodes vM is calculated over.

Finally, let *network* virtual mobility for time t_k be

$$vM^k = \frac{1}{N} \sum_{i=1}^N vM_{avg}^k(node_i) \quad (6)$$

where N is the number of nodes in the network.

This basic definition yields the *mean* network vM -value at every time interval. It represents how an average node moves during a test. The upper and lower quantiles of this mean value reflect the movement heterogeneity and can reveal different movement patterns within the network. Depending on the focus of interest, the basic vM definition can be altered in order to take multiple hops into account, to use active links only, or to weight links differently etc.

IV. EXPERIMENTAL SET-UP

A. Physical Environment

We have created scenarios for both outdoor and indoor environments. To stress the vM metric with as complex signal propagation patterns as possible we here choose to focus on the experiments performed indoors. The indoor experiments took place in three interconnected buildings on campus shown in Figure 3. The three buildings have a combination of offices and lecture halls. The white paths through the buildings are the corridors and bridges connecting the buildings. The letters [A...H] marked in the corridors refer to specific spots where nodes will be placed during different scenarios described below. The buildings stretches over totally 174 meters. The walls within the buildings are very thick and radio signals are effectively damped when loosing line-of-sight. The nodes move at normal walking speed, in this case meaning slightly above 1 m/s.

B. Scenario Descriptions

Three scenarios will be described briefly below for which we will show measurement results in Section V. Figure 4 in conjunction with Figure 3 give an overview of the exact movements and times within these three scenarios.

1) *Lost'n'Found*: The purpose of this scenario is to examine how different movements and link breakages can be captured by the vM metric. The scenario is choreographed in the following way. Assume a group of nodes that resides at location D. After a while the group splits into two clusters and one of the clusters moves away towards location A. At some point when the moving cluster is between location D and A the two clusters lose radio contact with each other. The remote cluster stays at location A for a while before returning to D. The data traffic in this scenario consists of all nodes sending broadcast Ping.

2) *Double Lost'n'Found*: This scenario focuses on how different movement patterns are visualized by the vM metric. The differences from the *Lost'n'Found* scenario are that here there are three clusters residing at E and two of them are “lost” simultaneously when moving towards A and H, respectively, but “found” at different times when moving back towards E. The traffic in this scenario consists of unicast Pings from each cluster to the other two clusters.

3) *Double Split*: The goal with this scenario is to create a multi-hop setting and to stress the routing daemon with weak and fluctuating link qualities. Assume two groups of nodes, one located at point C and the other at point F. There should be radio contact between point C and point F. After a while the two groups split into equally sized clusters. One of the clusters located at C starts moving towards point B, and one of the clusters located at point F starts moving towards point G. During the stay at the remote destinations these clusters will only have radio contact with the other neighbor cluster at their respective place of origin, i.e., each group has radio contact with only their adjacent group(s). After some time the remote clusters start moving back towards their respective place of origin and the other cluster. The traffic in this scenario consists of unicast Pings from each cluster to the other three clusters.

V. FIRST RESULTS

We will present results from repeated experiments using the AODV Mad-hoc implementation [9] and Inria’s OLSR implementation [11] with the number of participating nodes ranging from 9 to 37 nodes. The focus in section V-A will be on using the vM metric as a tool for capturing movements within the network and comparing repeated testruns. In section V-B we focus on the ping success ratio and IP packet hop count calculations. The size of some of the log files used to extract these results reached 70 MB and contained time stamped entries for more than 3 million packets.

A. vM Analysis

Figures 5, 6 and 7 in this section show the network virtual mobility (m/s) as a function of time (s). The solid line represents the mean value. The upper quantile and lower quantile are represented by dotted and dashed lines, respectively. These are here called vM -high and vM -low.

Although all nodes at some point are standing still according to the choreography, the vM value will indicate small movements. This is simply the normal fluctuation of the radio signal quality between nodes standing closely together. Inherent radio propagation properties like reflection as well as very small movements by participants could cause signal quality changes.

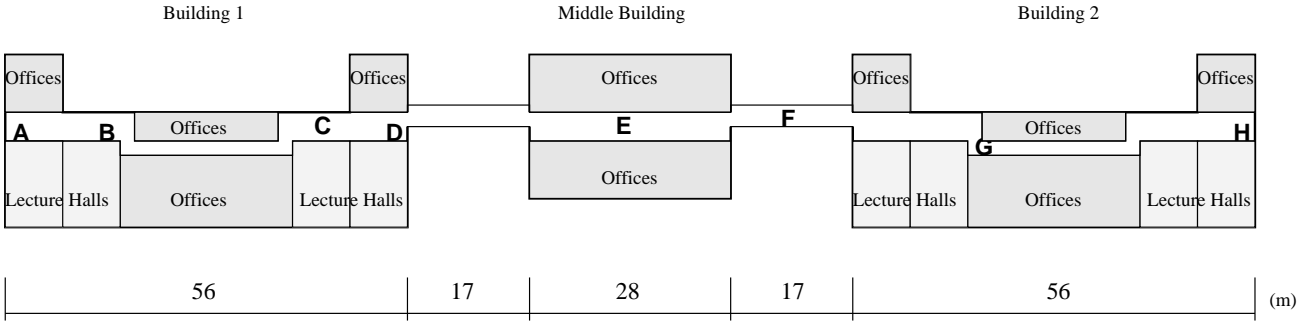


Fig. 3. The physical test environment.

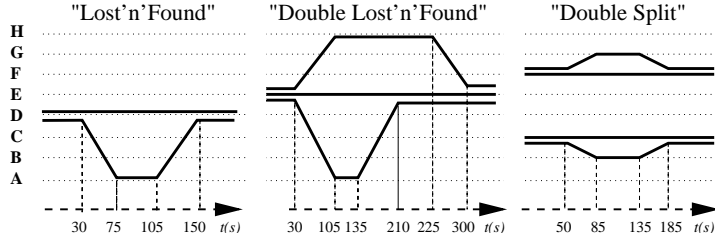


Fig. 4. Time-space diagrams for the three experimental scenarios discussed in Section IV-B.

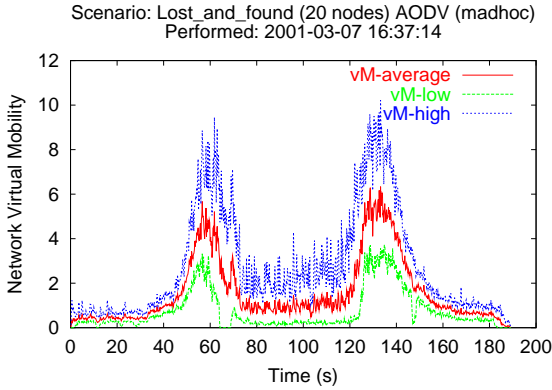


Fig. 5. The virtual mobility for scenario *lost'n'found* with 20 participating nodes.

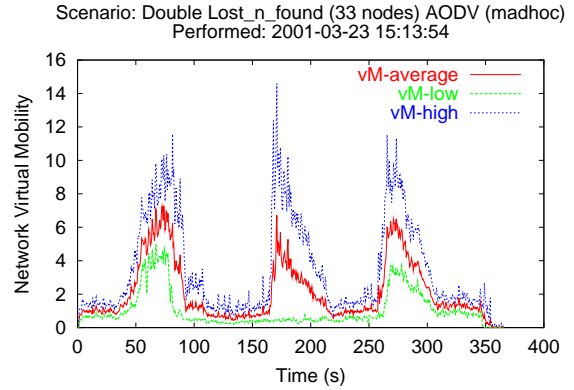


Fig. 6. The virtual mobility for scenario *double lost'n'found* with 33 participating nodes.

1) *Characterizing a Testrun with vM*: Figure 5 shows vM for a *Lost'n'Found* scenario with 20 nodes. The two peaks at time 60 and 125 correspond to when the two groups start losing and regaining radio contact with the other group. During the time when the two groups are out of radio range of each other the mean-value curve is back to approximately the same level as when the two groups were standing right next to each other. This vM graph together with the known choreography represents the character of this testrun.

2) *Deriving Topological Positions from Signal Quality Information*: It is possible to recreate the topological correct configuration of an experiment using the collected signal quality data. This allows us to visualize logical connectivity for each node and to provide an intuitive background for understanding metrics like virtual mobility, packet loss, and optimal route set-up. In Figure 12 we have taken a snapshot of APE-view every 25 seconds during a replay of the *Lost'n'Found* scenario.

3) *Discerning Different Movement Patterns Among Nodes*: Figure 6 shows the vM values for a *Double-Lost'n'Found* scenario run with 33 nodes. The three peaks represents the split (at time 80) and the two re-connections (at time 170 and 260). The split event gets the highest average mobility factor because the number of nodes that lost their contact with neighboring nodes during the split is higher than the number of nodes that did regain contact with other nodes during each of the two re-connections.

The vM-low curve in Figure 6 has no peak during the first re-connection around time 170, indicating low mobility among some of the nodes. This is perfectly reasonable because the third group, still standing still at the remote location at this time, did not have contact with any of the other two groups and should not yield any significant mobility. Hence, we can use the vM-low and vM-high values to test whether different groups have different mobility patterns during a testrun.

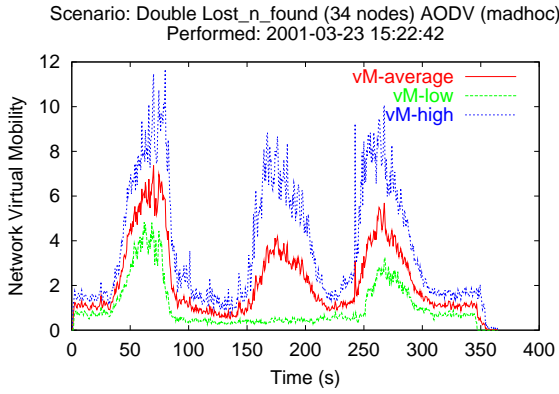


Fig. 7. The virtual mobility for scenario *double lost'n'found* with 34 participating nodes.

4) *Reproducing Testruns*: We repeated testruns several times in order to examine how well we can *reproduce* results. Figure 7 show the second run of the Double Lost'n'Found scenario (compare with Figure 6). The only difference to the first testrun is that the second testrun comprises 34 nodes instead of 33 (which was due to a computer crash). The participants get exactly the same instructions in both testruns and thus should move in the same way. Comparing our first testrun (Figure 6) with the second testrun (Figure 7) we can see that the overall shape of the curves match well and to some extent also the value of the vM. We conclude that our choreographed approach ensures that the participants move in the same fashion and that the vM can reveal whether the signal quality fluctuations in two testruns are similar enough to make results from further performance analysis comparable.

B. Packet Loss and Hop Count Analysis

Figure 8 shows the Ping success ratio for the *Double Lost'n'Found* scenario with 34 nodes, using the Mad-hoc AODV [9] implementation that is based on AODV draft version 5. The Ping success ratio starts to decrease right after the nodes start to move (compare with Figure 7). At time 65 the slope of the Ping curve becomes steeper which is right in the middle of the first peak in Figure 7. During time 90 and 160 no Ping packets get through between the groups. This is the time when the three groups have no radio contact. During the re-connection phase of the first groups the ping success ratio increases and stabilizes around 0.5 until time 250 when the second group enters into radio range and all nodes gradually regain full connectivity. Figure 9 shows the connectivity over time and one can clearly see the different phases of the Double Lost'n'Found scenario.

One would expect that – as the two clusters at the edges move apart from each other – they would re-route their traffic via the middle cluster. However, analysis of the hop count reveals that this is not the case for the Mad-hoc implementation (see Table I). In a Double Split scenario where the clusters retain contact with their adjacent groups, we even observed that Mad-hoc did not find ping paths longer than 2 hops at all.

Hop Count Analysis – Mad-hoc-AODV vs. OLSR-Inria: We repeated the Double Lost'n'Found and Double Split testruns,

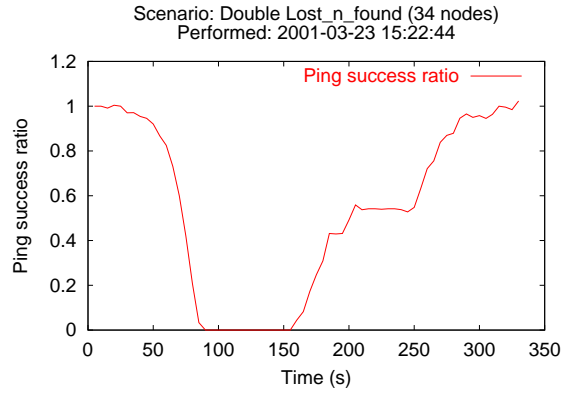


Fig. 8. Ping success ratio for the *Double Lost'n'Found* testrun with 34 nodes.

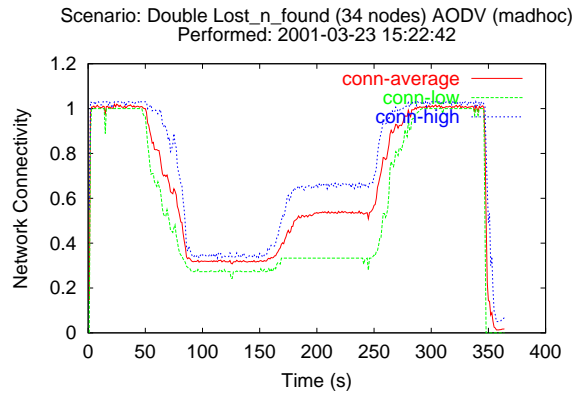


Fig. 9. Connectivity for the *Double Lost'n'Found* testrun with 34 nodes.

with 9 and 8 nodes, respectively, with the Mad-hoc-AODV software and the OLSR-Inria implementation [11] which is based on OLSR draft version 3. The Figures 10 and 11 show the link changes per second during the Double Split scenario for Mad-hoc and OLSR-Inria, respectively: The graphs indicate that the implementations have been exposed to similar degrees of fluctuating links.

In the tables II to V we compare the number of hops that pings were taking during the testruns. The results shown in these tables confirm our intuitive finding: The OLSR-Inria implementation runs smoothly and has no problem to set up multi-hop paths. For example, it managed to set up 4-hop paths (7 hops when counting forth and back path) in the Double Split scenario. Mad-hoc-AODV, on the other hand, seems to fail in detecting link breakages and setting up new multi-hop routes in

	1 hop	2 hops	3 hops
Request paths	1127	1	–
Reply paths	1127	1	–
Complete pings	n/a	1126	2

TABLE I
NUMBER OF SUCCESSFUL PINGS CLASSIFIED BY THE NUMBER OF HOPS,
FOR MAD-HOC IN *Double Lost'n'Found* WITH 34 NODES

	1 hop	2 hops	3 hops
Request paths	2940	1	–
Reply paths	2940	1	–
Complete pings	n/a	2939	2

TABLE II

NUMBER OF SUCCESSFUL PINGS CLASSIFIED BY THE NUMBER OF HOPS, FOR MAD-HOC IN *Double Lost'n'Found* WITH 9 NODES

	1 hop	2 hops	3 hops	4 hops
Request paths	3009	40	–	–
Reply paths	2974	71	4	–
Complete pings	n/a	2963	53	33

TABLE III

NUMBER OF SUCCESSFUL PINGS CLASSIFIED BY THE NUMBER OF HOPS, FOR OLSR-INRIA IN *Double Lost'n'Found* WITH 9 NODES

	1 hop	2 hops	3 hops
Request paths	2962	2	–
Reply paths	2959	5	–
Complete pings	n/a	2957	7

TABLE IV

NUMBER OF SUCCESSFUL PINGS CLASSIFIED BY THE NUMBER OF HOPS, FOR MAD-HOC IN *Double Split* WITH 8 NODES

	1 hop	2 hops	3 hops	4 hops
Request paths	2892	371	100	15
Reply paths	2894	380	98	6
Complete pings	n/a	2870	46	336

	5 hops	6 hops	7 hops	8 hops
Complete pings	26	86	14	–

TABLE V

NUMBER OF SUCCESSFUL PINGS CLASSIFIED BY THE NUMBER OF HOPS, FOR OLSR-INRIA IN *Double Split* WITH 8 NODES

mobile scenarios. Hence, although the OLSR-Inria implementation seems stable and capable of setting up multi-hop routes, we have to conclude for the moment that a direct comparison of the protocol performances is not yet possible.

VI. SUMMARY AND FUTURE WORK

After many years of simulations and in order to become accepted Internet protocols it is important to test ad hoc routing protocols with a large number of nodes in real world settings. In this paper we report on the Ad hoc Protocol Evaluation testbed (APE): It is based on choreographed testruns which enables to accurately repeat experiments. APE has a modular architecture allowing us to plug in different routing protocols and traffic generators. So far APE was used for testruns with 3 different protocols and up to 37 nodes. Our experiences are that quality and maturity of protocol implementations differs heavily and although we have AODV, TORA and OLSR working inside APE, we need more (and more stable) implementations to make full use of APE. However, our experiments show that APE can also be used for examining implementation behavior, as in the side-by-side comparison between Mad-hoc AODV and OLSR-Inria. We plan to make the APE testbed distribution available to other research institutions [17].

As a part of our APE efforts we have developed a new mobility metric called “virtual mobility” (vM); It is based on changes in the measured signal quality and characterizes a network’s mobility condition. This metric provides the basis for assessing the reproducibility of testruns in comparison experiments. Our choreography approach turned out to be successful under this metric; Virtual distances can also be used to reconstruct topological network configurations without additional positioning information. We expect the gathered signal strength data to be further useful for validating existing simulation models and for running trace-driven simulations.

ACKNOWLEDGEMENTS

We would like to thank Ericsson for donating the wireless ORiNOCO PCMCIA cards and Per Gunningberg for setting up this project’s framework. Many thanks to our colleagues and the students who participated in the experiments.

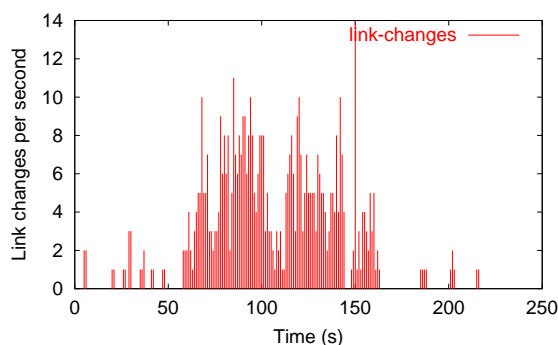


Fig. 10. Link changes per second for Mad-hoc in the *Double Split* testrun with 8 nodes.

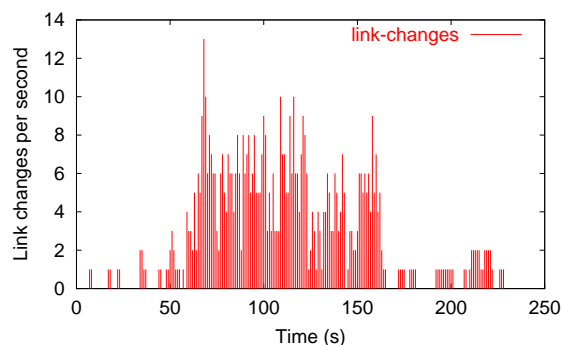


Fig. 11. Link changes per second for OLSR-Inria in the *Double Split* testrun with 8 nodes.

REFERENCES

- [1] *The Official IETF working group Manet webpage*, <http://www.ietf.org/html.charters/manet-charter.html>
- [2] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva. *A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols*. Proceedings of ACM/IEEE MobiCom'98, October 1998.
- [3] P. Johanson, T. Larsson, N. Hedman, B. Mielczarek, and M. Degermark. *Scenario-based Performance Analysis of Routing Protocols for Mobile Adhoc Networks*, Proceedings of ACM/IEEE MobiCom'99, August 1999.
- [4] D. Maltz, J. Broch and D.B. Johnson, *Quantitative Lessons from a Full-Scale Multi-Hop Ad Hoc Network Testbed*. Proceedings of WCNC 2000, September 2000.
- [5] C-K. Toh, Richard Chen, Minar Delwar, and Donald Allen. *Experimenting with an Ad Hoc Wireless Network on Campus: Insights and Experiences* ACM SIGMETRICS Performance Evaluation Review, Vol. 28, No. 3, pages 21-29, 2001.
- [6] C-K. Toh, Minar Delwar, and Donald Allen. *Evaluating the Communication Performance of an Ad Hoc Wireless Network* To Appear in IEEE Journal on Selected Areas in Communications (JSAC Wireless Series), 2001.
- [7] R. Ramanathan, R. Hain, *An ad hoc wireless testbed for scalable, adaptive QoS support*. Proceedings of WCNC 2000, September 2000.
- [8] C. PERKINS, E. ROYER AND S. DAS: *Ad hoc On-demand Distance Vector (AODV) Routing*, Internet Draft, draft-ietf-manet-aodv-09.txt, work in progress, November 2001.
- [9] *Mad-hoc AODV technical documentation*, <http://mad-hoc.flyinglinux.net/techdoc.ps>
- [10] P. JACQUET, P. MUHLEHALER, A. QAYYUM, A. LAOUITI, L. VIENNOT, T. CLAUSEN: *Optimized Link State Routing Protocol*, Internet Draft, draft-ietf-manet-olsr-05.txt, work in progress, October 2001.
- [11] *HIPERCOM OLSR implementation*, <http://menetou.inria.fr/olsr/>
- [12] V. PARK, S. CORSON: *Temporally-Ordered Routing Algorithm (TORA)*, Internet Draft, draft-ietf-manet-tora-spec-04.txt, work in progress, July 2001.
- [13] *UMD TORA/IMEP implementation*, <http://www.cshcn.umd.edu/tora.shtml>
- [14] *Linux Driver for ORiNOCO v6.06 webpage*, <http://www.wavelan.com/>
- [15] Y-C. HU AND D. JOHNSON: *Caching Strategies in On-Demand Routing Protocols for Wireless Ad Hoc Networks*, Proceedings of ACM/IEEE MobiCom'2000, August 2000.
- [16] A. Kamerman, *Coexistence between Bluetooth and IEEE 802.11 CCK Solutions to Avoid Mutual Interference*, Lucent Technologies Bell Laboratories, Jan. 1999, also available as IEEE 802.11-00/162, July 2000.
- [17] *APE testbed project page*, <http://apetestbed.sourceforge.net>

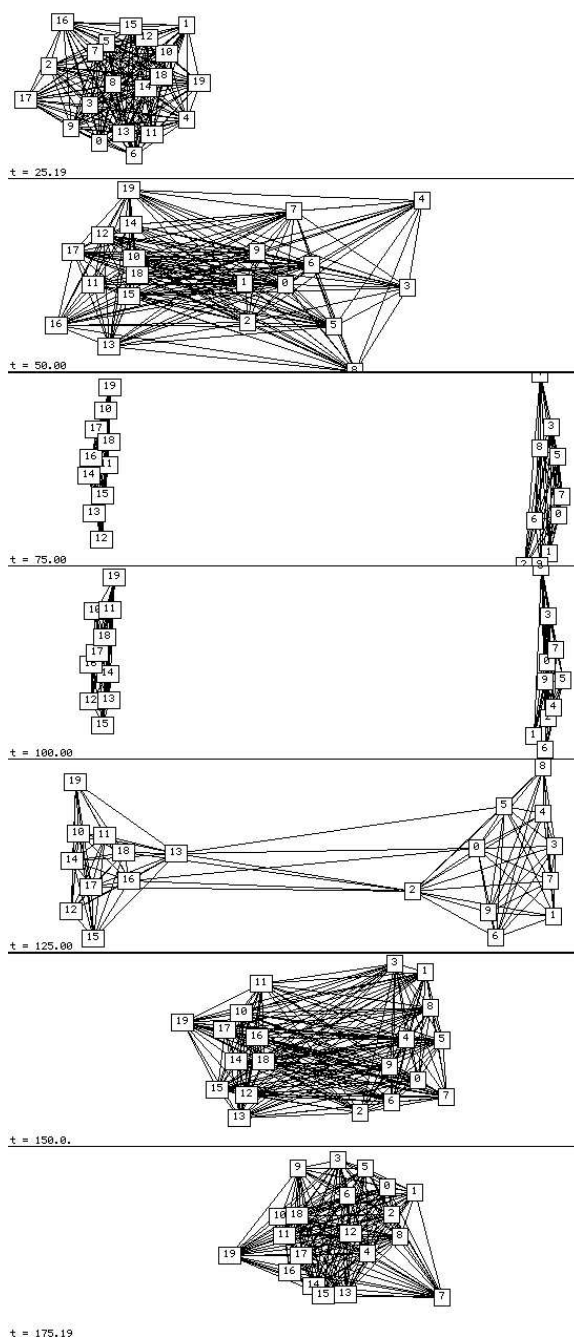


Fig. 12. Snapshots of virtual positioning from a replay of the “Lost’n’Found” scenario with 20 nodes (see Section V-A.2).

Paper C

Henrik Lundgren, Erik Nordström and Christian Tschudin, *Coping with Communication Gray Zones in IEEE 802.11b based Ad hoc Networks*, In Proceedings of The Fifth ACM International Workshop On Wireless Mobile Multimedia 2002 (WoWMoM 2002), September 2002.

Coping with Communication Gray Zones in IEEE 802.11b based Ad hoc Networks

Henrik Lundgren
henrikl@docs.uu.se

Erik Nordström
erik.nordstrom@it.uu.se

Christian Tschudin
tschudin@docs.uu.se

Department of Information Technology
Uppsala University
S-751 05 Uppsala, Sweden

ABSTRACT

Our experiments with IEEE 802.11b based wireless ad hoc networks show that neighbor sensing with broadcast messages introduces “communication gray zones”: in such zones data messages cannot be exchanged although the HELLO messages indicate neighbor reachability. This leads to a systematic mismatch between the route state and the real world connectivity, resulting in disruptive behavior for multi-media data transfer over ad hoc routing protocols. Concentrating on AODV we explore this issue and evaluate three different techniques to overcome the gray zone problem. We present quantitative measurements of these improvements and discuss the consequences for ad hoc routing protocols and their implementations.

Categories and Subject Descriptors

C.2.1 [Computer-Communication Networks]: Network Architecture and Design—*wireless communication*
; C.2.2 [Computer-Communication Networks]: Network Protocols—*routing protocols*

General Terms

Design, Experimentation, Measurement, Performance

Keywords

Communication gray zone, gray zone, IEEE 802.11b, MANET, mobile ad hoc networks, real-world experiment, routing protocols

1. INTRODUCTION

Wireless ad hoc networks consist of autonomous mobile nodes which provide a joint network service. The involved routing protocols must detect multihop paths and, in the range of a few seconds or below, react on changes in the topology. Such timing requirements and the characteristics

of wireless links make conventional Internet routing protocols inappropriate for ad hoc networks.

Several ad hoc routing protocols like DSR [6], AODV [13] or OLSR [5] have been proposed in the last 5 to 10 years. These protocols have been subject to intensive evaluations through simulation, but far less effort has been documented on the evaluation of the corresponding protocol implementations. When we measured the performance of our own fully conformant AODV implementation (called AODV-UU [2]), we observed an unexpected high amount of packet loss, especially during route changes. We found that the increased amount of packet loss coincided with specific geographic locations that we call *communication gray zones*. Inside gray zones the packet loss is severe and applications with continuous packet flow, like streaming multi-media and large file transfers, will suffer severe performance losses under such circumstances.

Reproducing our tests and comparing them with the behavior of the implementations of OLSR [12] and LUNAR [8] confirmed AODV-UU’s poor results. OLSR and LUNAR were chosen for comparison because of their availability and their different routing strategies. AODV is a reactive protocol which discovers and maintains routes on demand. When a route to an unknown node is needed AODV broadcasts a route request that is disseminated through the network. If the destination, or a fresh route to the destination, is found a route reply is unicast back to the source. During this process routes are set-up inside the traversed nodes’ routing tables. In addition, periodic broadcast HELLO beacons are used to sense neighboring nodes and based on this routes can be added, deleted or updated. OLSR is a proactive protocol which senses the network topology in the 10-second range using broadcast messages. LUNAR is a hybrid protocol as the on-demand discovery is combined with a proactive route re-discovery every third second. As with AODV, the route requests are broadcasted while the route replies are sent via unicast.

In this paper we show that gray zones are linked to the difference between messages that are broadcasted (e.g., AODV’s HELLO messages) and the other unicast data packets. Three different schemes counteracting the communication gray zones were added to AODV-UU, and their effectiveness was validated through controlled real world measurements with Ping, MP3 streaming and intermittent HTTP traffic. We discuss these findings and how they relate to other ad hoc routing protocols in general.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

WoWMoM’02, September 28, 2002, Atlanta, Georgia, USA.
Copyright 2002 ACM 1-58113-474-6/02/0009 ...\$5.00.

The rest of the paper is outlined as follows. In Section 2 we explain the gray zone problem and why it appears. In Section 3 we describe three mechanisms that reduce or eliminate the gray zone problem. Results from experiments with the enhanced AODV-UU implementation are reported in Section 4 before discussing and concluding our findings in Sections 5 and 6, respectively.

2. COMMUNICATION GRAY ZONES

Comparisons of MANET routing protocols based on simulations, which often present AODV in a favorable manner with good performance, are readily available, such as those in [4] and [7]. However, these findings could not be reproduced in real world: We observed strange performance problems of the AODV-UU implementation. In this section we explain how the problem of “communication gray zones” manifests itself and why AODV’s standard HELLO messages are inappropriate for neighbor sensing when using IEEE 802.11b. We also discuss why this problem is not evident in simulations using ns-2.

2.1 Performance Problems of Original AODV

We compared the performances of AODV-UU with those for OLSR [12] and LUNAR [8] in identical scenarios. In most cases AODV-UU performed better than OLSR, but that was what we expected because OLSR suffers from a 10 seconds re-route time. LUNAR and AODV-UU were approximately on par in most tests, but as the LUNAR implementation indicated some problems in stressed multi-hop configurations we had expected AODV-UU to win those contests. However, a more careful analysis of the AODV-UU results indicated that in some specific locations a node could have a valid route in its routing table, but no data got through to that next hop. We call the areas where we experienced this problem *communication gray zones*. In such gray zones, a node will experience considerable packet loss. The magnitude of the packet loss is larger than what can be explained by the re-routing that would occur when a node loses contact with its next hop.

Our measurements were made with a simple in-door mobility scenario that we call “*Roaming node*” (see Figure 1). The scenario is strictly choreographed and the experiments were performed using the APE testbed [3, 9]. The experimental setting consists of a total of four nodes where three of them are stationary (GW, C1 and C2), while a fourth mobile node (MN) is moving at a speed of approximately 1 m/s and “roams” the network. The MN is constantly communicating with the gateway node GW and will theoretically always have a possible route towards the GW. The scenario is run in a time period spanning 290 seconds: During this time traffic is routed over one, two and three hops via intermediate stationary nodes C1 and C2. This scenario lets us isolate and examine the route change phenomenas that we had experienced in testruns under various other conditions and configurations.

2.2 Conditions for the Forming of Communication Gray Zones

AODV relies on neighbor sensing to keep track of those nodes which are used as relay points for data transmissions. The neighbor sensing algorithm must therefore be able to detect when a link to a neighboring node can forward data. To this end, AODV uses periodic HELLO messages. These

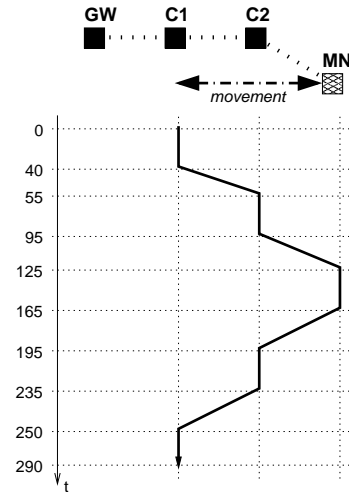


Figure 1: A simple “roaming node” scenario for a wireless multihop stub network.

HELLO messages have several salient properties that differentiate them from data packets and that contribute to the occurrence of “gray zones”: while HELLO messages can be heard, the same is not true for data packets to be exchanged between two “neighbors”.

- a. **Different Transmission Rate** – In IEEE 802.11b, broadcasting is always done at a basic bit rate while data transmissions normally are sent at higher rates (up to 11 Mbit/s in IEEE 802.11b). Transmissions at lower bit rates are more reliable and can reach further than at higher rates. As HELLO messages are broadcasted, this is the main cause to why gray zones appear.
- b. **No Acknowledgments** – Broadcast messages in IEEE 802.11b are transmitted without acknowledgments. HELLO messages are therefore not guaranteed to be sent over bidirectional links i.e., receiving a HELLO message is no indication that transmissions are possible in the opposite direction.
- c. **Small Packet Size** – The size of a HELLO message is small compared to a data packet. Small packets are less prone to bit errors since there are less bits to transfer than in large packets. Also, they have a smaller probability of colliding with the usually longer data packets. This makes it more likely for a HELLO message to reach a receiver than a data packet, especially over weak links.
- d. **Fluctuating Links** – At the transmission borderline, communication tends to be unreliable due to fluctuating quality of links. This leads to spurious HELLO messages which, once received, do not reflect correctly whether consistent communication between two nodes is possible or not. As a consequence this means that stable and longer routes can be replaced by shorter but unreliable ones.

All these elements together contribute, in various degrees, to the occurrence of communication gray zones.

2.3 The Shape of Communication Gray Zones

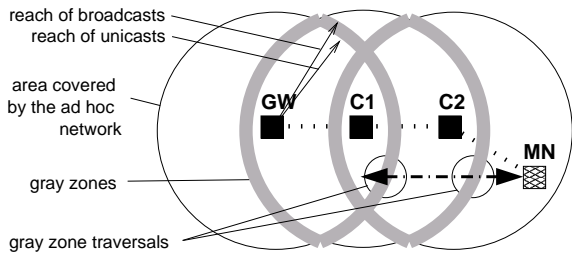


Figure 2: Communication gray zones for the “roaming node” scenario.

Figure 2 depicts in an idealized way where in the “roaming node” scenario communication gray zones can be experienced (the union of the three circles shows the area where the mobile node potentially can route to any of the three stationary nodes). In its journey from the place of C1 over C2 to its rightmost position, the mobile node will traverse two gray zones, namely when losing contact to the gateway node GW and when losing contact with the intermediate node C1. Similarly, two gray zone traversals are experienced when moving back, namely when regaining contact with C1 and GW, respectively.

These four traversals are easily found in the ping success charts shown in the appendix and discussed in Section 4.

2.4 Unrealistic ns-2 simulations

The implementation of IEEE 802.11 in ns-2 [11] does all transmissions at a bit rate of 2 Mbit/s, whether it be unicast or broadcast transmissions. Connectivity is also implemented as an on/off switch, where transmission suddenly breaks at a specified distance. In such a model, all properties mentioned above (except c) are not represented. This leads to uniform transmission ranges regardless of transmission rate, type and time, effectively preventing communication gray zones to emerge.

3. ELIMINATING GRAY ZONES

In this section we present three different modifications to AODV-UU that we experimented with to help neighbor sensing and reduce gray zones. The modifications are not in the AODV draft, but to some degree have all been proposed and discussed on various mailing lists, such as the AODV Implementors list or the IETF MANET mailing list.

We have added the suggested modifications to the AODV-UU implementation and then evaluated them in the APE testbed. In Section 4 we will then present how these modifications affect the performance of AODV.

3.1 Exchanging Neighbor Sets

To address the problem of unidirectional links when using broadcast HELLO messages, nodes exchange their neighbor set in an extension field of the HELLO messages. A node receiving such a HELLO message can then tell, by looking for its own address, if the link to the sender is bidirectional.

A new potential problem is introduced insofar as HELLO messages become variable in size, which in turn makes the success rate of HELLO messages depend on how many neigh-

bors a node detects. Furthermore, using a neighbor set extension will introduce a handshake-like procedure into the neighbor detection system of AODV. When two nodes discover each other as neighbors, they must acknowledge the other node’s HELLO message through the neighbor set extension before the detection process is complete. This will introduce a latency which may affect AODV’s ability to quickly adapt to changes in connectivity. Finally, this approach does not address the difference between unicast and broadcast transmissions.

3.2 N-Consecutive HELLOS

It has been proposed to request that a node should receive N consecutive HELLO messages from the same source before accepting it as a neighbor. N is typically set to 2 or 3. The idea is to bring stability into the changes in neighbor sets and ultimately the routes. On the other hand, this will introduce a latency which may hurt the on-demand properties of the protocol. This approach is again based on broadcasts only and is not sensitive to unicast/broadcast differences. However, it addresses the problem of fluctuating links.

3.3 SNR Threshold for Control Packets

The third way to improve neighbor sensing that we have evaluated is to use signal quality from the IEEE 802.11b driver as a criteria for “weak” control messages: control packets are discarded when they are received with a signal quality that is below some threshold¹. Intuitively this will counteract the gray zone problem, by forcing AODV to detect a longer route when link quality is so bad that data supposedly cannot get through while HELLO messages still can. Using a link quality criteria for broadcast messages will also minimize the probability of unidirectional links being present, although it cannot guarantee links to be bidirectional.

The down-side of this approach is that when cutting off AODV control traffic, AODV might not always be able to do a “best effort” attempt at delivering messages over a poor link, when no other alternative is available. Therefore one could expect that although the overall experience of the routing is smoother, there are times when AODV using link quality bias will not be able to deliver data while original AODV would.

4. RESULTS

We present results from experiments using the neighbor sensing extensions discussed in Section 3. For each protocol and each protocol variant dozens of test runs were performed and analyzed for establishing the qualitative basis of our findings. The actual quantitative numbers are extracted and averaged from two test series using the “Roaming node” scenario (see Figure 1).

The analysis focuses on gray zones and packet delivery success. First we analyze Ping delivery success in detail and compare the unmodified AODV-UU with the proposed neighbor sensing extensions. Next, we examine how gray

¹In our implementation we had to approximate this behavior: the 802.11b driver does not provide SNR values for individual packets wherefore we read the last value recorded for a given sender. Although we do this for each broadcast packet received, there is a slight chance that the SNR value belongs to some unicast packet that was received more recently.

zones affect two potential ad hoc networking applications: MP3 streaming and HTTP-based webpage requests. The active communication session in these experiments is always between the MN and the GW node.

4.1 Ping Delivery Success

Here we present detailed results of round trip packet delivery success when running a Ping application. Data traffic load consists of node MN sending 512-byte pings with the Record Route option which sums up to 580-byte IP packets (including IP header) to the GW node, at a rate of 10 packets/s. As this is a moderate traffic load and pings are sent at larger intervals than the round trip time, we know that packet losses are not due to the load put on the network. Ping delivery success ratio is calculated as the number of received ping replies divided by the number of sent ping requests during a one-second interval. The breadth and the depth of the dips in ping delivery success ratio indicates the range and severity of the gray zones. In the following discussion we present one representative testrun for each approach (see appendix for figures) –averaged numbers obtained from repeated testruns can be found in Table 1.

4.1.1 Original AODV-UU

Theoretically one would expect close to zero packet losses throughout the test because there is no self-colliding traffic and connectivity is always available. In order to verify this, we have modeled the “roaming node scenario” in ns-2 and have placed the AODV-UU code into the simulation environment. The (preliminary) results are that AODV-UU should successfully deliver more than 98% of the ping traffic. However, it is clear from inspecting figure 3 and the log files that AODV-UU running in the real world experiment did not live up to such expectations: The ping delivery success ratio for the original AODV-UU implementation documents severe packet losses during all moving periods.

Detailed inspection of the logs reveals the following. Between time 49 and time 56 it loses 10% to 100% of the packets. The second dip is between time 110 and time 124 where the ping delivery success varies from 12% and 100%. During time 172 to 193 the ping delivery success goes down to a minimum of 50%. During the short time period between 242 and 243 there is a complete loss of packets. The overall ping delivery success ratio is 91%. During three of the four gray zones we experience a complete drop out of packet delivery, while in one case it drops to 50%. Gray zones seem to stretch from approximately 5 to 20 seconds.

4.1.2 Exchanging Neighbor Sets

Including the neighbor set in HELLO messages should avoid uni-directional links as it requires the incoming HELLO messages to contain its own address, otherwise the sender is not considered to be a neighbor. In fact, we can see in Figure 4 that both the breadth and the depth of most dips in ping delivery success ratio are smaller than in the original version. The overall ping delivery success ratio raised from 91% to 97%.

4.1.3 3-Consecutive HELLOs

A visual comparison between the Figure 4 and Figure 5 clearly indicates less packet loss for the 3-Hello approach. Both the depth and breadth are significantly smaller. Specifically, one can see a reduction in packet loss during the pe-

riod when the mobile node MN is moving back and switches to a shorter route: during the gray zone traversals, packet loss is now below 10% on average. The original AODV-UU suffers from spurious HELLO messages in these cases because it directly installs new routes that not necessarily indicate stable and reliable transmission capability. The 3-consecutive HELLO approach successfully addresses this problem as it requires the new, shorter routes to become stable before replacing the existing ones. The overall ping delivery success ratio in this particular testrun was 99% but other repeated testruns have shown slightly less success.

4.1.4 SNR-Threshold for Control Packets

Setting the SNR threshold to 8 dBm and discarding control packets below this level produces the least packet loss of the three different approaches (see Figure 6 and Table 1). This can be explained by the fact that not only the problem of longer transmission range for broadcasted HELLO messages is addressed but also the problems of unidirectional links and spurious HELLOs: the probability of links being unidirectional decreases as we have logically shrunk the transmission border. Furthermore, spurious HELLO messages do not necessarily disrupt communication anymore. If logically we have an unstable link, indicated by the reception of spurious HELLOs, chances are still good to successfully transmit data packets to the next hop. Thus, this SNR threshold approach decreases the probability that installed routing entries do not reflect the true communication capabilities.

Detailed log inspection reveals the following. At the first dip we observe a ping delivery success ratio of 50%, but only during a one second interval. During time 101 to 118 there are several occurrences of minor packet loss but they are mostly around 10%. Although we are ignoring control messages below some signal quality threshold, we can see in our logs that the route change does not occur until time 113 which indicates that we could increase the threshold even more. However, further experiments with different threshold values have not produced significantly better performances. This indicates that it is hard to obtain completely smooth route changes when switching to longer routes. During route changes while moving back there are only singular packets lost at two occasions.

4.1.5 Ping Experiment Summary

Table 1 shows a summary of the total packet delivery ratio for the different approaches, averaged over several repeated testruns. We see that all three modifications to AODV-UU increase the delivery success ratio significantly. Most effective is the SNR threshold approach although it does not achieve lossless route changes.

Table 1: Overall Ping Delivery Success Ratio for the AODV-UU modifications (“Roaming node” scenario).

AODV-Original	91.9%
AODV-Neighb. set	97.7%
AODV-3-Hellos	98.0%
AODV-SNR thresh.	99.1%

4.2 Continuous MP3 Streaming

In this setting we continuously send MP3 music from the GW node to the MN node throughout the testrun. The MP3 music file used was encoded at 128 Kbit/s. We measured the percentage of successfully and in-order delivered MP3 packets. Apart from testing a real application, MP3 streaming differs from Ping in that the MP3 sessions are one-way only and do not accept out-of-order delivery of packets. There was not a single MP3 packet delivered out-of-order in any of the experiments. In comparison, the original AODV-UU had a few out-of-order deliveries during the Ping experiments.

Table 2: Overall MP3 packet delivery success ratio for AODV-UU modifications (“Roaming node” scenario).

AODV-Original	97.9%
AODV-Neighb. set	98.6%
AODV-3-Hellos	98.9%
AODV-SNR thresh.	99.7%

Table 2 shows the results from continuous MP3 streaming in the Roaming node experiment. The perceived playback quality during the testruns corresponds well to the results in Table 2: the SNR threshold and 3-HELLO approaches sounded very good with only minor “glitches” (<1s) in the music while the glitches in Neighbor Set Extension and Original AODV occurred more often and overall had longer duration (1-3s).

4.3 Intermittent HTTP Requests

Although not as conclusive as with Ping and MP3 experiments, we report on the AODV’s routing performance using HTTP traffic. Our simple web user model uses a static think time of 8 seconds and a data set size of 34 KB. These values are loosely based on [10] and [1] (10 to 15 seconds median user think time and 10 to 39 KB average data transfer). We counted the number of successful cycles and measured the request completion time. If the node is located in a gray zone at the moment it requests the webpage the TCP SYN packet will potentially not reach the destination. TCP will try to re-send the SYN packet after 3 seconds, 9 seconds, 21 seconds – after 30 seconds the request fails. This affects the number of successful requests as well as the average access time.

Table 3: Overall number of successful webpage accesses and average access time for AODV-UU modifications (“Roaming node” scenario).

<i>Protocol</i>	<i>HTTP cycles</i>	<i>avg. access time (sec)</i>
AODV-Original	33	0.84
AODV-Neighb. set	34.5	0.43
AODV-3-Hellos	33	0.90
AODV-SNR thresh	34	0.54

Table 3 shows a summary of the HTTP request experiments. The higher average access times for the Original

AODV-UU and the 3-HELLO extension is due to HTTP request failures. The Neighbor Set and SNR threshold extensions had a few TCP SYN retransmission but no failures. If we instead consider the median access time we can see in our log files that it only differed by 0.02 seconds among all the different approaches. We conclude that short intermittent traffic bursts are not as sensitive to gray zones as traffic with continuous data flows.

5. DISCUSSION

In this section we present results from comparisons with OLSR and LUNAR. Furthermore, we discuss the implications of gray zones for OLSR and LUNAR as well as for ad hoc protocols in general.

5.1 Comparison Against OLSR and LUNAR

We repeated all tests with the OLSR and LUNAR protocols, as a point of reference. Figure 7 and 8 show the Ping delivery success charts for OLSR and LUNAR, respectively. Table 4 summarizes the comparison results from all three experiments (Ping, MP3 and HTTP access).

Table 4: Comparison against OLSR and LUNAR for all three experiments (“Roaming node” scenario).

<i>Protocol</i>	<i>success ratio</i>		<i>HTTP cycles</i>
	<i>Ping</i>	<i>MP3</i>	
OLSR	89.0%	91.9%	32.5
LUNAR	96.5%	96.8%	31.5
AODV-UU	91.9%	97.9%	33
AODV-UU+SNR	99.1%	99.7%	34

5.2 Protocols without Gray Zone Problem

It seems that OLSR and LUNAR are less sensitive than AODV to communication gray zones for two different reasons.

Although OLSR uses broadcasts to disseminate its routing table data, it is possible that a mobile node does not stay sufficiently long in the gray zone for OLSR to be able to react (wrongly) on this. Thus, OLSR’s low overall Ping/MP3 success ratio is mainly due to the slow discovery of topology changes because of its proactive routing approach.

LUNAR does not rely on a broadcast neighbor sensing algorithm. Instead, it re-discovers delivery paths every third second. Thus, the creation of new routing table entries is *solely* based on *unicast* route replies, which mitigates the gray zone problem for LUNAR. Note that AODV too creates routing table entries based on unicast RREP messages. However, when using HELLO messages (instead of link layer notification), original AODV also adds routing table entries based on broadcasts. The lower overall performance of LUNAR can be explained by its 3-seconds “forget and re-learn” approach which potentially leads to longer packet loss periods.

An interesting future research topic is the problem of handling (intermediate) nodes which happen to be permanently located in a gray zone or which stay there for an extended period of time. In such cases it would be useful to make routing decisions based on the end-to-end quality of the routing path instead of local decisions only.

5.3 IEEE 802.11b is not Bidirectional

Successful reception of messages over IEEE 802.11b does not always imply that links are bidirectional. We have shown for AODV that such an assumption, currently built into simulation models, has adverse performance effects. Routing protocols that, without access to link level notifications, have to use HELLO like broadcast messages, need to be revisited and have to explicitly cope with communication gray zones. This can be in form of a mixed broadcast/unicast approach as in LUNAR, or signal quality based measures as we experimented with for AODV. One problem of the latter is that determining the exact cut-off level could be context specific. Ideally we would like the cut-off level to always logically reduce the range of broadcasts to match the range of unicasts. Reducing the range too much may prevent some otherwise acceptable communication, but will still make AODV resistant to communication gray zones.

6. CONCLUSIONS

In this paper we provide evidence for IEEE 802.11b based wireless ad hoc networks suffering from “communication gray zones”: In such zones it is possible to receive broadcasts but it is unlikely to successfully send or receive unicast messages. This leads to invalid routing table entries for protocols that establish their neighbor set using HELLO beacons, as e.g. AODV.

We have explored this problem and implemented three different gray zone work-arounds in the AODV-UU software. Their effect was evaluated in controlled real world experiments which showed that all three modifications substantially increase the packet delivery ratio. Only by enabling these modifications we were able to obtain real world performance figures that matched the simulation results. The approach that introduces a signal quality threshold for AODV control packet acceptance almost totally eliminates the effect of communication gray zones. In mobile scenarios, this gray zone elimination is most important for applications with continuous, real-time packet flows e.g., multi-media streaming.

In conclusion we state that ad hoc routing protocols which operate over IEEE 802.11b need to explicitly address communication gray zones. It could be by design of the protocol using broadcasts *and* unicasts in appropriate ways, by artificially limiting the range of broadcast messages or by basing routing decisions on the end-to-end quality instead of relying on local decisions.

7. REFERENCES

- [1] H. ABRAHAMSSON, B. AHLGREN: *Using empirical distributions to characterize web client traffic and to generate synthetic traffic*. Proc. IEEE Globecom: Global Internet, November 2000.
- [2] *AODV-UU webpage*
<http://www.docs.uu.se/scanet/aodv/>
- [3] *APE testbed project webpage*
<http://apetestbed.sourceforge.net>
- [4] J. BROCH, D. A. MALTZ, D. B. JOHNSON, Y. C. HU, J. JETCHEVA: *A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols*. Proc. ACM/IEEE International Conference on Mobile Computing and Networking 1998 (MobiCom'98), October 1998.
- [5] T. CLAUSEN, P. JACQUET, A. LAOUITI, P. MUHLETHALER, A. QAYYUM ET L. VIENNOT: *Optimized Link State Routing Protocol*. IEEE National Multi-Topic Conference (INMIC 2001), December 2001.
- [6] D. JOHNSON, D. MALTZ, J. BROCH: *DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks*. In Ad Hoc Networking, edited by C. Perkins, pp. 139-172, Addison-Wesley, 2001.
- [7] P. JOHANSON, T. LARSSON, N. HEDMAN, B. MIELCZAREK, AND M. DEGERMARK: *Scenario-based Performance Analysis of Routing Protocols for Mobile Adhoc Networks*, Proc. ACM/IEEE International Conference on Mobile Computing and Networking 1999 (MobiCom'99), August 1999.
- [8] *LUNAR webpage*
<http://www.docs.uu.se/selnet/lunar/>
- [9] H. LUNDGREN, D. LUNDBERG, J. NIELSEN, E. NORDSTRÖM, C. TSCHUDIN: *A Large-scale Testbed for Reproducible Ad hoc Protocol Evaluations*. Proc. IEEE Wireless Communications and Networking Conference 2002 (WCNC'02), March 2002.
- [10] B. MAH: *An Empirical Model of HTTP Network Traffic*. Proc. INFOCOM '97, April 1997.
- [11] *The Network Simulator - ns-2 webpage*
<http://www.isi.edu/nsnam/ns/>
- [12] *OLSR implementation (Inria)*
<http://menetou.inria.fr/olsr/>
- [13] C. PERKINS AND E. ROYER: *Ad hoc On-Demand Distance Vector Routing*. Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99), February 1999.

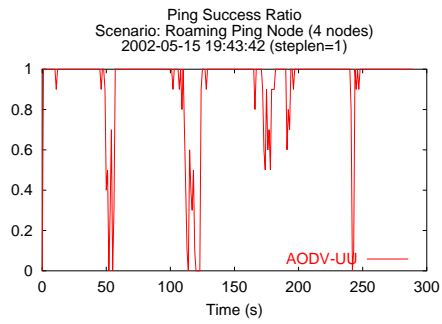


Figure 3: Original AODV-UU

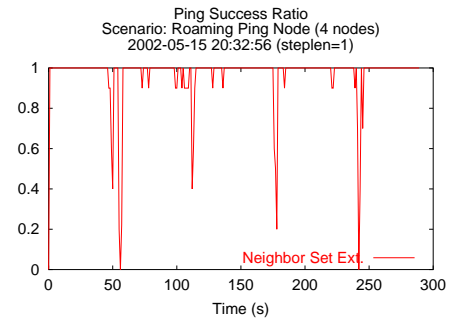


Figure 4: AODV-UU with exchanging neighbor sets

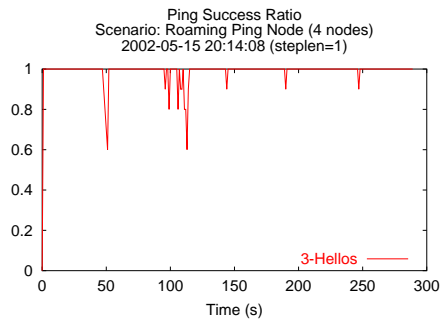


Figure 5: AODV-UU with 3-Hello extension

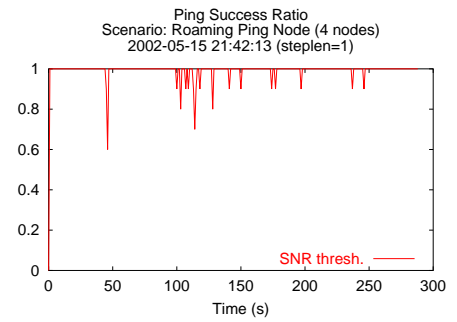


Figure 6: AODV-UU with SNR threshold for control packets=8 dBm

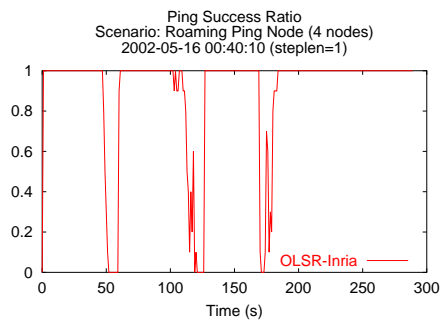


Figure 7: OLSR

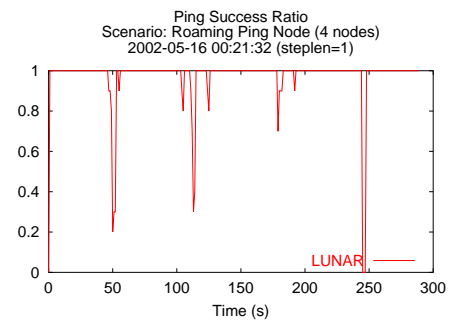


Figure 8: LUNAR

Recent licentiate theses from the Department of Information Technology

- 2002-001** Eva Mossberg: *Higher Order Finite Difference Methods for Wave Propagation Problems*
- 2002-002** Anders Berglund: *On the Understanding of Computer Network Protocols*
- 2002-003** Emad Abd-Elrady: *Harmonic Signal Modeling Based on the Wiener Model Structure*
- 2002-004** Martin Nilsson: *Iterative Solution of Maxwell's Equations in Frequency Domain*
- 2002-005** Kaushik Mahata: *Identification of Dynamic Errors-in-Variables Models*
- 2002-006** Kalyani Munasinghe: *On Using Mobile Agents for Load Balancing in High Performance Computing*
- 2002-007** Samuel Sundberg: *Semi-Toeplitz Preconditioning for Linearized Boundary Layer Problems*
- 2002-008** Henrik Lundgren: *Implementation and Real-world Evaluation of Routing Protocols for Wireless Ad hoc Networks*



UPPSALA
UNIVERSITET