# A Testbed and Methodology for Experimental Evaluation of Wireless Mobile Ad hoc Networks

Erik Nordström Per Gunningberg Henrik Lundgren

Department of Information Technology
Uppsala University
{*erikn,perg,henrikl*}*@it.uu.se*

## Abstract

*Wireless mobile ad hoc network experimentation is subjected to stochastic factors from the radio environment and node mobility. To achieve test repeatability and result reproducibility such stochastic factors need to be controlled or assessed in order to obtain conclusive results. This has implications on the design of testbeds. We present a methodology that addresses repeatability and describe how it has guided us in the design of our Ad hoc Protocol Evaluation (APE) testbed. Finally, by using APE, we present side-by-side routing protocol comparison results and show a radio phenomena that is not visible in simulations.*

## 1 Introduction

Wireless mobile ad hoc networks [1] are infrastructure-less and can be deployed instantly and for temporary purposes. Ad hoc networks promises to be a useful technology in disaster areas, for military units in combat zones, for spontaneous meetings and in other scenarios where communication infrastructure is not readily available. In ad hoc networks there is no distinction between hosts and routers – all nodes forward packets. Nodes typically use radio to communicate, they may be mobile and they move relative to each other. This causes rapid topology changes that we do not see in the fixed, wired Internet. The range of radio communication limits the number of neighbors that a node can communicate with. Packets therefore, in general, need to traverse several hops to reach their destination. The chosen path changes with the mobility of nodes, thus requiring frequent routing updates. The current Internet routing protocols are inefficient in this highly dynamic topology. A research goal is to study how existing Internet protocols can cope with the dynamics and to propose new protocols. Much of the work take place in the IETF MANET group [7]. Most of the research so far is based on theoretical models, simulations and emulations. There is, however, an increasing need for experimental testbeds that can provide all the complex interactions between the environment and the subject under test that models, simulations and emulations fall short of. Experiences from real experiments are also necessary for improving the models. In this paper we present such a testbed for experimental evaluation of ad hoc routing protocols and a methodology for conducting conclusive experiments.

Compared to testbeds with static wired nodes, ad hoc testbeds present new challenges due to the radio communication and the inherent mobility of the nodes. First, links are intermittently connected and have varying quality that depends on topographical factors, such as building structures and slow time varying factors such as weather and temperature. Radio communication is also affected by relative fast time varying interference, such as hidden terminals, multi-path effects [9], and gray zones [17] (i.e., zones where nodes can detect radio signals from neighbors but packet transmission is poor). This time varying environment is not completely controllable and will introduce a stochastic factor from one experiment to another. Second, node mobility is often unpredictable and hard to repeat. Ideally, the movement pattern of all nodes from one experiment to another should be identical. In real world testbeds there are always some stochastic differences in the movement which have impact on the measurements. To conclude, these differences in combination with the varying radio environment will cause variations in the topology, i.e., the radio connectivity between nodes. Obviously, these small variations in the topology may have a significant effect on the routing protocol and network performance. In an ad hoc network testbed we need to ensure that this ”topology jitter” over time is controlled so that results from several experiments

[1] For the sake of simplicity and readability, we will use the term “ad hoc” instead of “wireless mobile ad hoc” throughout this document.

are comparable.

Another difference to wired testbeds, related to the node mobility, is a need for mobility scenarios, i.e., how the nodes should move during an experiment and how many nodes that should participate. It should be possible to mimic for example how people would be organized and move in a Search and Rescue operation. This example raises the scalability issue of a testbed with respect to the number of nodes, the duration of an experiment and the scaling of the geographical movement, both in terms of speed and displacement.

We present our ad hoc testbed APE and our methodology that we use for controlled and repeatable experiments. APE is freely available and has been running for three years. It is used by numerous organizations around the world. The testbed is based on laptops that are carried around according to a scenario experiment. Different ad hoc routing protocols can then be tested together with higher layer protocols and applications with respect to the same scenario. The protocol and application environment is Linux, but the testbed itself also runs on Windows. Currently we use IEEE 802.11b cards in ad hoc mode for radio communication. To ensure repeatable experiments each laptop provide instructions in real time, to the person who carries the laptop, where and when to go to specific locations known ahead of time. These instructions comes from a scenario description. Measurements are collected during the experiments and are analyzed afterwards. We collect information about the topology seen from all nodes at all times, which makes it possible for us to compare the topology jitter from one experiment to another.

The type of questions our APE testbed intends to answer include: ”Which of the MANET protocol implementations has the highest overall throughput?” (under a specific scenario). ”How well will a routing protocol react to topology changes and will it always take the best path?”. ”What is the performance impact of a complex indoor environment, real operating system, implementation and a mobility pattern and how does it compare to simulation results?”. When designing experiments for these types of questions it is important to use a systematic methodology and to understand experiment limitations and to what extent results could be generalized.

The main contribution of this paper is a formulation of an ad hoc experimentation and testbed methodology, building upon our design of APE and several years of experiments. APE has enabled us to perform large-scale evaluations of ad hoc routing protocols where experiments are repeatable. Two important objectives for the design has been the management of the users and a non-intrusive measurement system that is fully integrated into the testbed environment. We describe APE and some results that show the importance of the methodology.

The following section discusses the trade-offs between simulation, emulation and real world experimentation. Related work are thereafter introduced. Section 4 formulates a methodology for repeatability, followed by a description of the APE testbed in Section 5. In Section 6 it is shown how APE supports the methodology and Section 7 presents experimental results using APE. Finally, we conclude the paper in Section 8.

## 2 Simulation, Emulation and Real World

Network research has successfully been relying on a combination of modeling, simulation and emulation for performance prediction. All three of them complement each other. While simulation and modeling simplify some parts of a real environment in order to understand the impact of other factors, emulation and real world experiments aim at capturing the full interaction between all parts. This interaction is difficult to model or simulate as it requires such a detail that it is not feasible. Emulation go half way between simulation and real world testing, by modeling some parts and running live other parts.

Emulator testbeds have in common that they try to address the problems of scaling, management and test repeatability by emulating the wireless channel and the mobility. Emulator testbeds can not always be a substitute for real world experiments as we later point out in section 7. A strength is that emulators allow repeatable experiments where results are easily reproduced and provide an efficient way to evaluate a prototype, for example a routing protocol. A downside is that many emulator testbeds require hardware components that are modified, for example custom-built wireless channel emulators. Therefore emulators are seldom easily reused by other researchers to reproduce and verify results.

For ad hoc networks, simulation is attractive since it is easy to control mobility, wireless randomness and scalability. Hence it is easy to repeat experiments and reproduce results. Research has shown that simulation models are not accurate enough to truly model the unpredictable environments that ad hoc network protocols are subjected to in the real world [17]. As a result, there is an increasing demand to complement simulation with real world testbeds and experimental research to improve the models.

## 3 Related Work

The MobiEmu emulator [24] is an example of an emulator testbed. It uses packet filters to emulate the mobility of an ad hoc network, typically using an Ethernet LAN, or a well connected non-mobile IEEE 802.11 network. Our APE testbed has a similar scenario driven dynamic filtering capability but as an emulator MobiEmu is more sophisticated and uses a master controller that connects to other hosts over a dedicated control network.

The emulators in [14, 15, 21] require special hardware and testbed components are not reusable outside the specific environments. Kaba and Raichle's "testbed on a desktop" [15] artificially reduces the radio ranges of their network cards. A similar approach is taken by Sanghani et al. in [21], where attenuators reduce the radio range and a switch is used to emulate mobility. In [14] Judd et al. digitize the output signals of wireless network cards and process them through a digital signal processing engine to emulate physical signal propagation before feeding them into the receiver(s). [14, 21] also discuss the issues of repeatability and the problem of unrealistic simulations.

In [12], Desilva et al., experimentally evaluate the performance of the AODV routing protocol in a test network consisting of six nodes. The nodes are static for repeatability and management reasons. Route breaks are emulated by artificially purging routing table entries. Toh et al. report on a real world experimental setup consisting of five nodes where they remove network cards to mimic mobility [22] .

Ramanathan and Hain [20], describe another experimental setup where wireless routers are used for real world experiments. They report that setting up and managing a mobile multi-hop experiment was far more difficult than they expected. Experimental results are presented for static settings of up to five nodes. To scale experiments beyond five nodes they use a software emulation of the radio channel.

The CMU real world testbed [18] is built upon a FreeBSD implementation of the DSR protocol [13] and consists of six moving nodes and two stationary nodes. The moving nodes are transported in cars equipped with GPS units for the purpose of continuously recording positions in order to be able to replay test runs in a visualization tool. The authors do not address repeatability in the paper and the complicated hardware setup makes it hard for other researchers to reproduce results.

MIT's Grid Roofnet [10] testbed consists of static rooftop nodes forming a multi-hop ad hoc network spanning a campus area. There are 38 stationary computers equipped with wireless LAN cards and omnidirectional antennas. It has been used to study the causes of packet loss [9, 11]. It is not suited for mobile multi-hop ad hoc network experimentation.

## 4 Methodology

The goal of our methodology is to fulfill the following principle:

> *Experiments must be repeatable and the repeatability assessable to guarantee the reproducibility of the results.*

For the purpose of the following discussion we make a distinction between repeatability and reproducibility. Repeatability concerns the ability to *repeat test runs* in a consistent manner. Reproducibility concerns the ability for researchers to *reproduce results* derived from previous experiments, typically – but not necessarily, performed by other researchers.

Experimental research of ad hoc networks is to a larger extent than simulation and emulation affected by stochastic factors (e.g., radio environment and node mobility) that make repeatable experiments harder to achieve. The impact of stochastic factors on repeatability is closely bound to the variance. If the variance is naturally small or can be reduced to an acceptable level by controlling some parameters, the negative effect on the repeatability could be acceptable. An acceptable level clearly depends on the type of experiments and the hypothesis under test.

In our methodology we deal with stochastic factors in the following ways. First, the primary goal is to reduce the number of stochastic factors. The second goal is to reduce the variance of the stochastic factors. Finally, we need to be able to monitor and assess the variance of the stochastic factors and the impact on the conclusiveness of our results. By systematically dealing with all factors we believe that experiments can be repeated in a satisfactory way. Other researchers might not be able to *exactly* reproduce measurement data but rather to, for example, reproduce results in the form of individual rankings between protocols or general trends in network performance.

The radio environment is an example of a stochastic factor that is hard or impossible to control and it affects network connectivity. We believe that this factor can only be assessed after the experiment. By letting a testbed monitor the *complete* connectivity at any time we can compare one connectivity map against a previous map and deduce whether they are similar enough. An unexpected disturbance can in that way be detected and analyzed before the experiment result is accepted.

The monitoring of the complete connectivity is a true distributed problem. No single node can alone infer from what it hear how the instant connectivity look like. Instead, all nodes need to collect their connectivity view and after the experiments this information is downloaded for analysis. There are several ways to address the mobility problem and in section 5 we describe in detail how we use movement choreography to achieve this.

The test platform software modules, operating system, and the applied configuration parameters may also interact in an unpredictable way, affecting the experiments. These parameters must be identified and controlled, preferably kept constant. This rises a management issue when scaling experiments. If easy management of these parameters is not part of the testbed design, then the scaling of experiments will introduce more complexity and more management. Increased complexity implies a higher risk of variance and loss of repeatability simply because there are more parameters to control. Therefore we decided that the number of parameters that relates to configuration and the interactions between software modules should be minimized.

## 5 The APE Testbed

In this section we introduce APE, the Ad hoc Protocol Evaluation testbed. We describe its components and how they have been designed with our methodology in mind. For a more detailed description of APE see [16, 19].

### 5.1 Software Overview

APE runs on laptops and is based on the Linux operating system. It was initially created to evaluate ad hoc network routing protocols. Since it is a Linux system it is possible to exploit the wide range of software available for this platform such as routing protocol implementations of AODV [1, 4, 5], DSR [2], OLSR [3, 8, 6] and LUNAR [23], but also tools such as traffic generators and logging software.

APE is distributed as a software package consisting of build scripts and source code. With them we build a stripped down and self-contained bootable Linux distribution which installs an *execution environment* on top of another operating system. The self-containment allows the execution environment to be tailor-made for a specific experiment and makes it easily distributed and installed on laptops at execution time. This also means that all nodes use the exact same configuration parameters. The execution environment is depicted in figure 1.

Within the execution environment we have built a scenario interpreter that reads commands and instructions from a scenario file and executes them at specified points in time. We have also added components for data logging and time synchronization between nodes. A modification to the wireless interface drivers, called *superspy*, allows logging of signal quality for all data packets received by the hardware. In addition, the APE software package comes with tools to perform post-experiment analysis of the collected data. This includes tools to perform time synchronization and merging of data from all nodes and to calculate connectivity, throughput, packet loss and hop counts.

### 5.2 Experiment Design

The first step is to design a movement scenario. As an example, the "Double Lost and Found" experiment consists of a movement scenario where three groups of nodes are initially positioned at the same location. At a given time two groups move away to a location outside the radio range

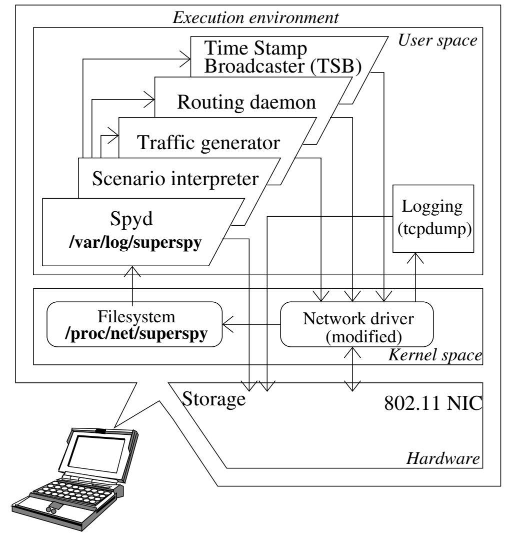


**Figure 1. Overview of the execution environment on APE nodes.**

of the other group. The two distant groups rejoin the third group at their initial position at *different* points in time. The data traffic scenario consists of when nodes should send packets to each other, which nodes that should communicate and how much data to send.

The next step is to transform movement and data transmission events into a scenario file. The designer specifies instructions to the test participants, that carry the laptops, to control when and where they shall move during the experiment. We call this the test *choreography*. The data traffic generated during the experiment is specified by startup times and parameters to traffic generators. Figure 2 shows an extract from a scenario file for one of the nodes in the Double Lost and Found experiment.

```
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=my_iperf –c 2 –t 330
node.0.action.1.msg=Stay at this location. You are sending data to node 2.
node.0.action.1.duration=30
node.0.action.2.msg=Start moving! Go to the end of House 1.
node.0.action.2.duration=75
node.0.action.3.msg=You should have arrived at the end of House 1. Please stay.
node.0.action.3.duration=30
node.0.action.4.msg=Ok, time to return to House 4!
node.0.action.4.duration=75
node.0.action.5.msg=You should be back now. Please stay and wait for the other group.
node.0.action.5.duration=90
node.0.action.6.msg=The other group should have arrived. Stay together for the last 30 sec.
node.0.action.6.duration=30
node.0.action.7.msg=The test is over. Thank you!
```

**Figure 2. Example scenario file for the Double Lost and Found experiment.**

The scenario file is divided into sequential *actions* that can contain up to three types of instructions to the scenario interpreter; a message to display to the test participant, a command to run on the test machine and a duration until the next action. In figure 2, action 0 starts the “superspy” daemon and the second action initiates a TCP connection to another machine. The rest of the actions are movement instructions to the test participant.

As the next step, the designer compiles an APE distribution package. After verifying that the scenario file works as expected the test participants consisting of, for example, students are assembled. They install the APE package on their laptops, reboot into the execution environment and are thereafter ready to run the experiment. The test administrator starts the test by sending a start packet to all nodes. This will trigger the scenario interpreter to step through the instructions and the test participants follow them when they pop up on their screens. At the end of the experiment the log files are uploaded to a central computer for post-experiment analysis.

Although participants have never used APE, the interactions necessary for participating in tests are so simple and few that persons with only basic computer skills can participate.

### 5.3 Measurements and Data Gathering

In wired networks, *passive* nodes with the sole purpose of snooping on the network can accurately capture network state and deduce connectivity as perceived by *active* nodes participating in the network. In wireless networks connectivity is more subjective. Two closely located nodes may not have the same perception of the network connectivity at all times due to the radio environment and potentially segmented topology. Therefore, measurements (e.g., throughput), conducted through passive listening might not accurately capture all the packets sent and received at the source and the sink. A more accurate view is achieved by letting the active nodes perform measurements themselves. A trade-off is that active logging may affect the system under test.

APE relies extensively on active logging on participating nodes. The network cards are put in promiscuous mode so that the network driver receives all packets that a network card can listen to. Signal level and signal noise along with the time of reception of each data packet is recorded. This information is later used to assess the connectivity in the network. To be able to time synchronize the gathered data all APE nodes run a Time Stamp Broadcaster (TSB) program. It broadcasts the local time once every 10 seconds. The neighboring nodes that receive this broadcast records the enclosed timestamp.

Logging of test specific data is handled independently and depends on the application, for example ping RTTs or FTP file transfer throughputs.

## 6 Assessing Repeatability

By using the link layer information on received data packets along with signal strength, APE provides a complete map of the link connectivity and all the link changes between all nodes that took place during a test run. At any instant it is possible to get a picture of how the nodes are connected. This means that we can compare two test runs by checking that the topologies are similar enough at the same time instant. Since we are dealing with stochastic radio phenomena, people that move independently and drifting clocks, we can not expect a complete match between two test runs. Therefore it is impractical to check each map instance against each other even if it is doable. Instead we do a statistical assessment to ensure the repeatability at a higher layer of abstraction.

### 6.1 Topological Replay

We have developed a tool, APE-view, that visualizes the topology changes. Figure 3 shows snapshots from a topological replay of a test run with the Double Lost and Found scenario. APE-view takes the complete topology map as input and replays the link quality changes as an animation on a display. The link quality is used to calculate a virtual distance between nodes. This information is used to display the logical topology of the network during an experiment. Each node participating in the test is drawn as a square labeled with its node number. Node connectivity (i.e., a packet was sent between a node pair during a time interval), is visualized as a line between the nodes. The length of the line represents the virtual distance. APE-view plays the scenario in discrete time and the resulting animation can be used to qualitatively assess that the scenario played out according to its intent.

### 6.2 Assessment Metrics

APE currently provides two ways to statistically assess the repeatability of two experiments in terms of link layer connectivity; the Link Change Metric and the Virtual Mobility Metric. Both produce a diagram showing a statistical metric of topology changes as a function of time. We call these diagrams *fingerprints* since we compare fingerprints from several test runs to ensure that they are similar enough for a comparison of the test results. What ”similar enough” means, is highly dependent on the protocol under test and the metric used. For example, it is possible to say that the statistical topology metric of two test runs do not deviate more than 10 percentage of each other.

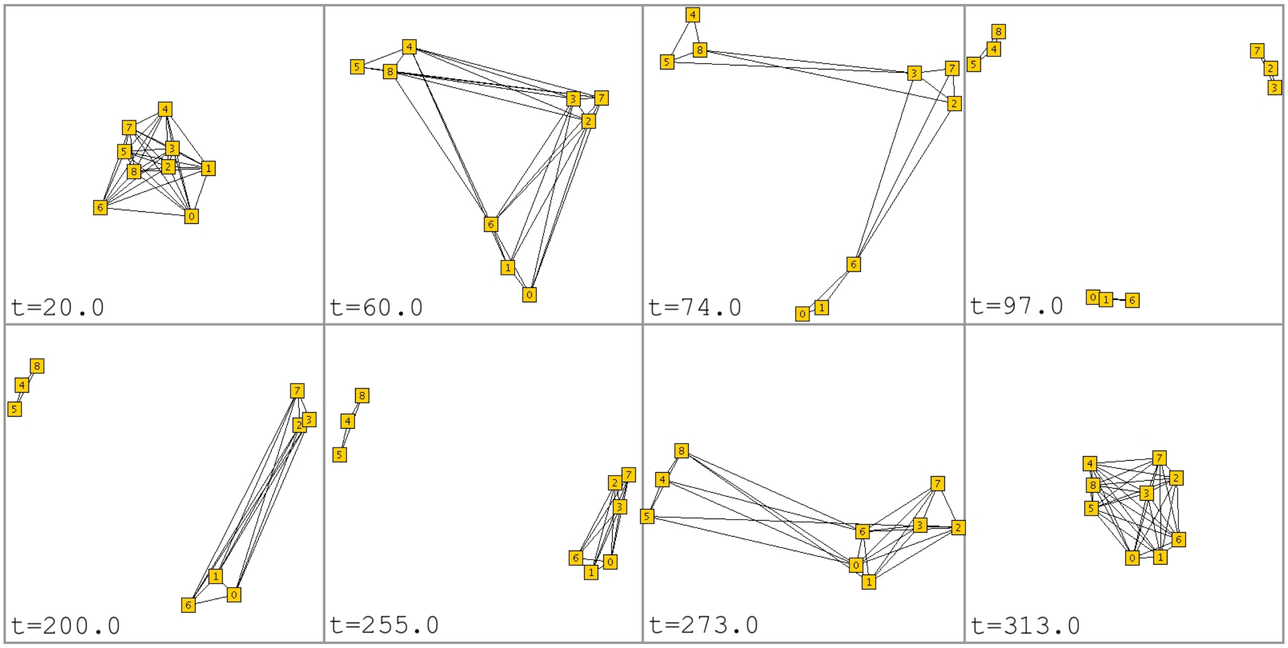


**Figure 3. Snapshots from an APE-view topological replay of a Double Lost and Found experiment.**

#### 6.2.1 Link Change Metric

The idea with this metric is to calculate the number of link changes per time unit. A lot of movement means a higher metric and a higher "temperature" in the network. The time interval for an experiment is $I$. It is divided into $n$ intervals of length $l$ where $n = \frac{I}{l}$. $l$ is chosen sufficiently small to notice frequent changes in connectivity but large enough to not cause erratic behavior. A typical value of $l$ is 1 or 2 seconds depending on how much mobility there is in the scenario. The number of packets received by $node_i$ from $node_j$ during time interval $t_k$, where $k = 0 \ldots n-1$, is $P_{ij}(t_k)$. Thus, a link has been established between $node_i$ and $node_j$ at time interval $t_k$ if $P_{ij}(t_k) > 0$ and $P_{ij}(t_{k-1}) = 0$, and a link has been lost when $P_{ij}(t_k) = 0$ and $P_{ij}(t_{k-1}) > 0$. The network wide occurrence of link changes (established and lost) $L(t_k)$ is then the sum of all nodes' link changes for time interval $t_k$. Figure 6 shows the frequency of link changes $L_{freq}(t_k) = \frac{L(t_k)}{l}$.

#### 6.2.2 Virtual Mobility Metric

While the Link Change Metric only captures the link changes, the Virtual Mobility metric also includes the signal quality as a measurement on the relative mobility. It uses a simple attenuation path loss model to map signal quality to a virtual distance. The Virtual mobility metric represents how on average a node moves (speed and how often) during a test. For the purpose of this discussion and for simplicity we use some empirical data in the loss model. Thus the distance $D$ in the range of 0.5 to 65 m is defined as

$$D_j(node_i) = 4 * 10^{\frac{40-0.9*Q_j(node_i)}{33}} \quad (1)$$

where $Q$ is the signal quality (0. . . 75) for a packet received from node $j$ at node $i$. Virtual Mobility between $node_i$ and $node_j$ is calculated for $node_i$ as follows. For a given time interval $t_k$, where $k = 0 \ldots n-1$ and $n = \frac{I}{l}$, we average the virtual distances obtained from all packets heard from a specific $node_j$. The virtual mobility $vM$ for $node_i$ with respect to $node_j$ for time interval $t_{k+1}$ is simply the change in the mean virtual distance between time interval $t_{k+1}$ and $t_k$. We calculate $vM$ at time $t_k$ for $node_i$ with respect to ***all*** other nodes in the network and average this sum to receive $node_i$'s average virtual mobility at time $t_k$. Finally, the *network* virtual mobility for time $t_k$ is the averaged sum of virtual mobilities calculated for all nodes in the network. The upper and lower quartiles of the mean value can also be calculated. This reflects the movement heterogeneity and can reveal different movement patterns within the network. See [16] for complete step-by-step formulas.

### 6.3 Assessment Example

Our scenario "Double Lost and Found" nicely demonstrates an assessment of repeatability with the virtual mobility and link changes metrics. We look at two test runs running different routing protocols (AODV and OLSR). Figure 4 shows the average network connectivity during the

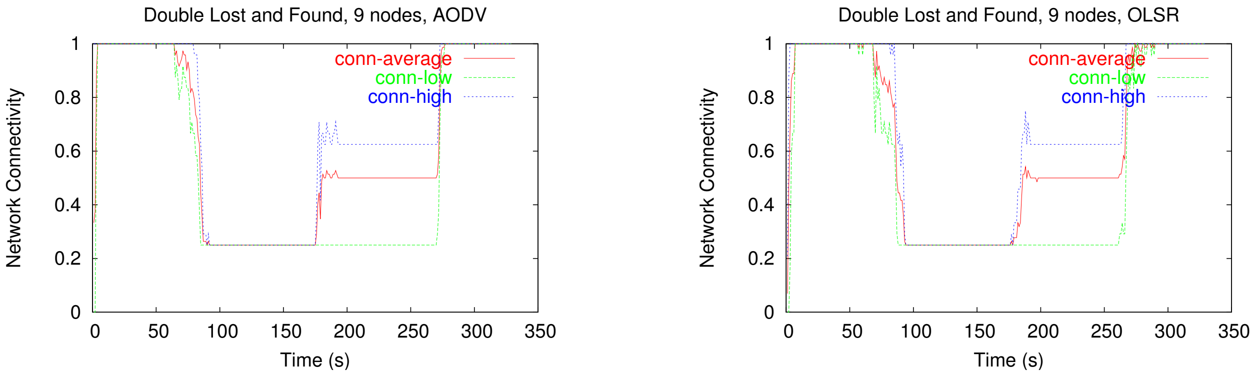


**Figure 4. Network connectivity during the “Double Lost and Found” scenario from two test runs (AODV and OLSR).**

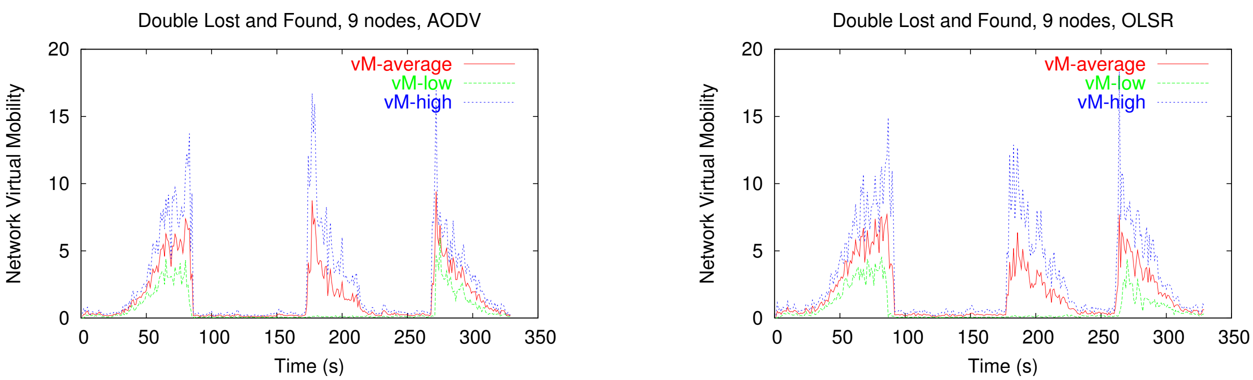


**Figure 5. Virtual mobility fingerprints where the three peaks correlates to node movement periods in the “Double Lost and Found” scenario.**

test runs and the upper and lower quartile (i.e., the average connectivity of the 25% most connected and 25% least connected nodes). A connectivity of 1 indicates full connectivity. Compare the connectivity with the snapshots in figure 3.

Figure 5 shows the average and the upper and lower quartile of the network virtual mobility from the two test runs with AODV and OLSR. We clearly see the moving periods being reflected as three peaks of virtual mobility. The observant reader note that the virtual mobility’s lower quartile of the middle peak is close to zero. This relates to when two groups re-join, while the third group is still isolated and have no mobility. The visual similarity between the two virtual mobility fingerprints indicate similar signal fluctuations.

Next, we examine the link change metric to ensure that the routing protocols were subjected to similar stress in terms of links established and lost. Figure 6 shows the corresponding fingerprints for AODV and OLSR. Note that some of the peaks are slightly shifted in time compared to the peaks in mobility. This shows that the link changes (established and lost) as expected occur first when nodes have moved far away enough or are getting close enough. Thus, the three peaks indicate high frequency of link changes at the times of separation and re-join of nodes. The spread of the link changes also shows how closely the group of nodes move. The timing and extent of the peaks correspond well in both virtual mobility and link change graphs.

We have argued that easy management of the testbed facilitates scalable experiments. This scalability was demonstrated when 33 students were drafted to participate in “Double Lost and Found” test runs. We experienced no major management difficulties. An interesting issue is how our virtual mobility and link change metrics perform with approximately four times the number of nodes. The larger the moving clusters become, the harder it becomes to control movement accurately simply because more people are involved which requires larger space when they are gathered

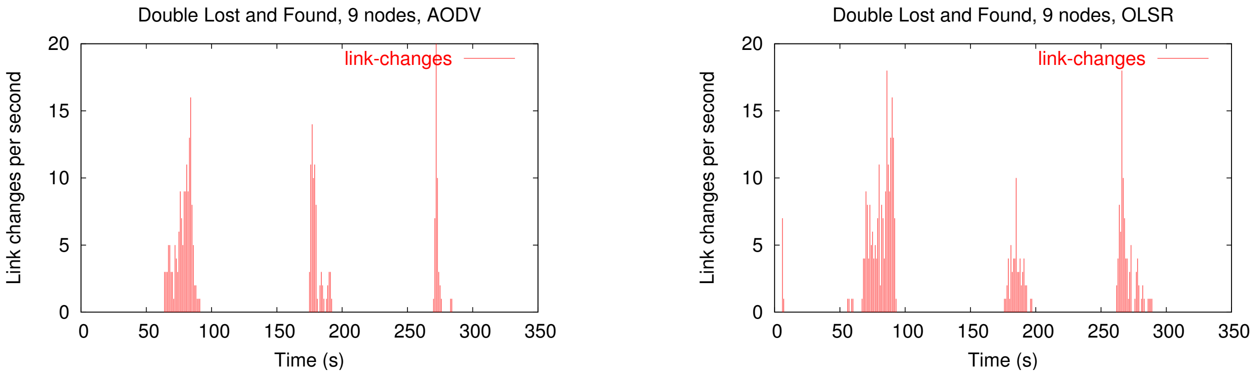


**Figure 6. Link change fingerprints for the "Double Lost and Found" scenario.**

together. In Figure 7, with 33 nodes, we easily recognize the overall characteristics for both virtual mobility and link changes. However, we note that the contours of the peaks have become a bit "blurred". The natural explanation is that when the participants move in quite narrow university corridors in groups of 11, they can not all move side by side so some of them have to move in a file.

## 7 Experiment Example: Routing Protocol Comparisons and the Gray Zone effect

The purpose with this example is twofold. First to show how side-by-side performance comparison of ad hoc routing protocols can be performed. Second, to highlight the importance of complementing simulation with real world experimental research by describing the "Gray zone" phenomena that does not show up in simulations.

We constructed a "roaming node" scenario for the purpose of conducting the side-by-side performance comparison. An overview of the scenario can be seen in figure 8. The scenario consisted of three static nodes positioned in a line such that they only had connectivity with the adjacent node(s). A forth node was mobile and moved alongside the static nodes to switch between a one, two and three hop configuration and back. It communicated with one of the end nodes acting as a fictitious gateway. This mobility pattern forced the protocol under test to deal with totally four route changes.

We repeated this experiment with OLSR [3], LUNAR [23] and AODV-UU [1] using Ping, MP3 streaming, and web request cycles. We ran several experiments with each protocol to achieve representative averages and the repeatability was assessed using the virtual mobility and link changes metrics. We ranked the protocols for each traffic type. We found that AODV did not perform as well as we had expected in comparison with the other protocols. After some investigation we found the main cause to be the difference in transmission rate between broadcast and unicast packets in 802.11b. While unicast packets may be sent with 11 Mbit/s, broadcast packets are always sent at the basic data rate of 2 Mbit/s, which translates to longer transmit ranges for broadcast packets. This caused AODV's Hello

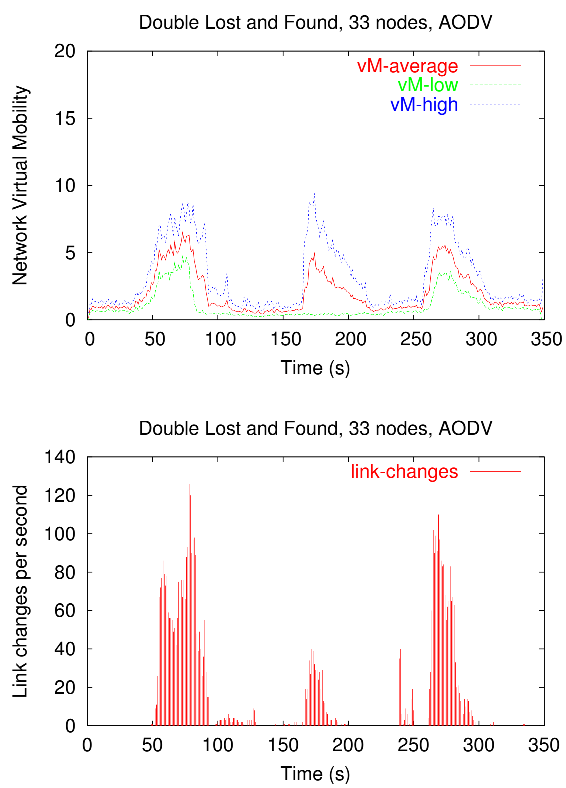


**Figure 7. Virtual Mobility and Link change fingerprints when scaling up the "Double Lost and Found" to 33 nodes.**

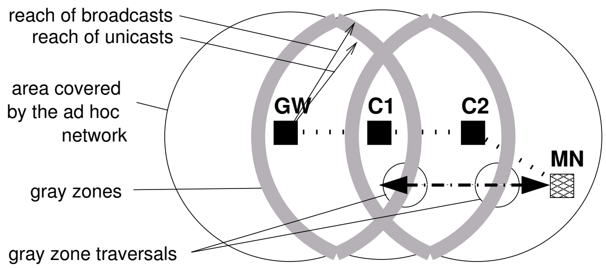


**Figure 8. The "roaming node" scenario and its communication gray zone areas.**

messages to falsely report nodes as neighbors when unicast transmissions were in fact not possible. Figure 8 illustrates how the gray zones were located in this particular scenario.

To reduce the effect of gray zones we implemented and tested three workarounds. The most successful workaround was to filter and drop AODV control packets based on their SNR value upon reception. Using an SNR threshold at 8 dBm we managed to eliminate the gray zones almost completely. Table 1 summarizes the results for the different protocols (see [17] for a more complete discussion on gray zones and the related results).

**Table 1. Summary of results from protocol comparisons in a scenario with communication gray zones.**

| | *success ratio* | | *HTTP* |
|---|---|---|---|
| *Protocol* | *Ping* | *MP3* | *cycles* |
| 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 |

The conclusion from this experiment is that although ad hoc routing protocols can be designed and prototyped in simulation, conclusions about their performance in the real world can not be drawn until they have been subjected to their final operating environment.

## 8 Discussion and Conclusions

Experimental testbeds are needed to complement modeling, simulation and emulation. It provides the complex interactions between the environment and the subject under test. Testbeds also provide input to simulation models. Wireless mobile ad hoc experimental testbeds face new challenges as the radio environment and node mobility represent stochastic factors that can not be fully controlled. This makes it hard to achieve repeatability and result reproducibility. We have described our methodology that aims at ensuring that experiments are repeatable. We have presented how our design the of the APE testbed assess repeatability. To be able to assess the "topology jitter" induced by the radio environment and the node mobility, APE enables a post-experiment global view of the complete network connectivity at any instant. In scenarios where several nodes move during the same time period, this metric provide fingerprints that are easy to compare between test runs, for assessing repeatability. To control the node movement, APE makes use of strict movement choreography. We have without difficulties designed and performed repeatable experiments with up to 37 mobile nodes. Although APE experiments scale well through APE's easy management, the generation of choreography scenario files becomes complex with increasing number of nodes and might eventually be a limitation. In that case tools have to be developed to make scenario creation easier. We conclude that, although APE and its realization of our methodology might not be perfectly suitable for all kinds of experiments, it has enabled us to perform conclusive side-by-side protocol comparisons.

## References


[1] The AODV-UU implementation. http://www.docs.uu.se/scanet/aodv.

[2] The click DSR router project. http://pecolab.colorado.edu/.

[3] Inria OLSR implementation. http://menetou.inria.fr/olsr/.

[4] The Kernel AODV implementation. http://w3.antd.nist.gov/wctg/aodv_kernel/.

[5] MadHoc AODV technical documentation. http://madhoc.flyinglinux.net/techdoc.ps.

[6] Navy research lab NRL OLSR implementation. http://pf.itd.nrl.navy.mil/.

[7] The official IETF MANET working group webpage. http://www.ietf.org/html.charters/manet-charter.html.

[8] UniK OLSR implementation. http://olsr.org.

[9] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level measurements from an 802.11b mesh network. In *Proceedings of ACM SIGCOMM 2004*, pages 121–131, August 2004.

[10] B. A. Chambers. The grid roofnet: a rooftop ad hoc wireless network. Master's thesis, MIT, June 2002.

[11] D. S. D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of MobiCom'03*, September 2003.

[12] S. Desilva and S. R. Das. Experimental evaluation of a wireless ad hoc network. In *Proceedings of the 9th Int. Conf. on Computer Communications and Networks (IC3N)*, October 2000.

[13] D. Johnson, D. Maltz, and J. Broch. DSR: The dynamic source routing protocol for multi-hop wireless ad hoc networks.

[14] G. Judd and P. Steenkiste. Repeatable and realistic wireless experimentation through physical emulation. In *Second Workshop on Hot Topics in Networks (HotNets-II*, 2003.

[15] J. Kaba and D. Raichle. Testbed on a desktop: Strategies and techniques to support multi-hop manet routing protocol development. In *Proccedings of 2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, October 2001.

[16] H. Lundgren, D. Lundberg, J. Nielsen, E. Nordström, and 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.*

[17] H. Lundgren, E. Nordström, and C. 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 (WoWMoM)*, September 2002.

[18] D. Maltz, J. Broch, and D. Johnson. Quantitative lessons from a full-scale multi-hop ad hoc network testbed. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, September 2000.

[19] E. Nordström. A large scale ad hoc network testbed for reproducible performance tests. http://www.csd.uu.se/courses/course-material/xjobb/docs-reports/Nordstrom-2002.pdf.

[20] 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)*, September 2000.

[21] S. Sanghani, T. X. Brown, S. Bhandare, and S. Doshi. Ewant: The emulated wireless ad hoc network testbed. In *IEEE Wireless Communications and Networking Conference (WCNC)*, March 2003.

[22] C.-K. Toh, M. Delwar, and D. Allen. Evaluating the communication performance of an ad hoc wireless network. In *IEEE Transaction on Wireless Communications*, July 2002.

[23] C. Tschudin, R. Gold, and O. Rensfelt. LUNAR - a Lightweight Underlay Network Ad-hoc Routing protocol and implementation. In *Next Generation Teletraffic and Wired/Wireless Advanced Networking (NEW2AN)*, 2004.

[24] 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.