Thoughts on Mobile Ad-hoc Network Testbeds

Wolfgang Kieß, Stephan Zalewski, Andreas Tarp, Martin Mauve
Heinrich-Heine University Düsseldorf
{kiess,mauve}@cs.uni-duesseldorf.de
{stephan.zalewski, andreas.tarp}@uni-duesseldorf.de

Abstract

Currently, several research groups seek to develop appropriate tools and methodologies for real-world experiments on mobile ad-hoc networks. We argue that this should be done as a community effort rather than as independent projects. Furthermore we present our view on the functionality a good testbed should provide based on reports from other research groups as well as on our own experiences. This paper aims at stimulating a discussion, it is not meant to be a comprehensive specification of requirements or a detailed workplan.

1 Introduction

The two most common instruments used to evaluate algorithms for mobile ad-hoc networks (MANETs) are theoretical analysis and network simulation. Theoretical analysis provides fundamental insights into the characteristics of the investigated approaches, simulation enables their exploration in a dynamic environment. Both methods require a significant level of abstraction to reduce the real-world complexity of mobility, radio propagation, and hardware. As it has been discussed in [13], the direct transfer of findings from simulations to real-world systems is thus not advisable. As a consequence, theoretical analysis and simulation should be complemented by real-world experiments.

A surprisingly large number of these real-world experiments have already been conducted, leading to results and insights that were not foreseen in simulation or theoretical analysis. Most of the experimenters also had to face unforeseen difficulties. Nodes failed during experiments which was only discovered after the experiment, results were difficult to reproduce and therefore work was unnecessarily duplicated by multiple work groups or the network showed unexplainable behavior. We believe that these problems can be alleviated if future experiments satisfy three key requirements of scientific experimentation:

- **Reproducibility.** Independent research groups must be able to reproduce the results of an experiment. For mobile ad-hoc networks, reproducibility is a significant challenge due to the complex impact of radio propagation and node mobility on the results of an experiment.

- **Comprehension.** A scientist conducting an experiment must be able to access all relevant information to comprehend and explain the results of the experiment. There is a need for tools that collect information on different layers and combine this information to allow a detailed analysis.

- **Correctness.** Any experiment may suffer from broken tools, errors with the setup and problems when conducting the experiment. While reproducibility and comprehension will most likely reveal these problems, it is vital to the efficiency of a researcher to be able to verify whether any given experiment has produced valid results. This can be supported by an established methodology and a selection of appropriate tools.

To address these issues, there are efforts underway that aim at the development of methodology and testbeds to support systematic experiments in mobile ad-hoc networks. One early example is the APE

project [14], introducing, among other things, virtual metrics for node distance in order to increase reproducibility. Several other testbeds are currently being worked on [19, 8]. All these efforts provide very valuable individual contributions. However, it is our belief that a community effort is necessary to establish real-world experiments as the third standard instrument to evaluate algorithms and protocols for MANETs. This effort must include an analysis of the functionality expected from a "perfect" testbed as well as a discussion of the methodology for conducting experiments. Ideally this will lead to the design of a common architecture where the individual research groups can contribute reusable building blocks for a testbed that is supported by the ad-hoc networking community.

This paper is meant as our initial contribution to this effort. It describes work in progress not final results. In Section Two we present our view on a perfect testbed for mobile ad-hoc networks. Section Three describes some of our experiences with experimenting and Section Four concludes the paper.

2 The Perfect Testbed

In the following we assume that a testbed consists of two key elements: a number of physical devices (nodes) which may be moved around individually and the software to support and conduct the experiments.

In general an experiment for MANETs can be divided in several phases: implementation, experiment specification, node configuration, setup verification, execution, and analysis. For each of these phases we discuss how an ideal testbed could support them and present the current state of the art in this area.

2.1 Implementation

The first phase of an experiment is the implementation of the algorithm to be tested. A good testbed will support this phase in three ways: it will 1) help to minimize the work required for the implementation; 2) seek to reduce implementation errors; 3) encourage interoperability between algorithms implemented and evaluated by distinct research groups.

As a lot of algorithms will be initially analyzed by means of simulation, reusing the simulation code instead of reimplementing it eases the workload and reduces the potential for errors. Thus tools allowing to use the same code both for simulation and the real experiment are a vital component of a MANET testbed. Additionally, the testbed should support the implementation by providing well tested libraries for common MANET functionalities such as beaconing or for tasks such as tracing.

Encouraging interoperability is mainly a matter of interfaces and methodology. A good testbed will specify concise interfaces and best-practice methods for integrating new functionality such that it can be reused by other research groups. It will also support interoperability through a clean and simple architecture.

The concept of reusing simulation code for real-world experiments has already been utilized in some projects, namely in SURAN [1], WINGS [5] and the routing protocol evaluation presented in [23]. A particularly promising approach seems to be nsclick [16] which allows to run a click router [12] inside the ns-2 [17] network simulator.

There also exist libraries supporting the implementation of algorithms for ad-hoc networks. One class are libraries such as the PICA API [2], designed to provide platform-independency for the implementations using them. Other libraries offer additional functionality like neighbor discovery, flooding or packet buffering during route discovery. One example is the ad-hoc support library [11].

2.2 Experiment Specification

After the implementation is complete, the experimenter specifies the scenario used for the evaluation. In order to allow other research groups to verify the results, the specification should be a complete description of the experiment made available as a file in a standardized format.

There exist at least two variants of scenarios, *strict* and *loose scenarios*. In a strict scenario each node follows detailed instructions on when to perform which action. Although a rigid description of the experiment fosters repeatability, there are setups in which this is not suitable, e.g., if experiments are run as background tasks on devices primarily used for other purposes. In this case, a scenario with loose descriptions of the services and actions able to adapt to the current state of the node is better suited. In both cases a good testbed

will support the specification of an experiment through a set of tools. At the very least these tools will allow the planning of node mobility and the timing of events via a graphical user interface.

The only widely known tool for the specification of strict scenarios are the choreography descriptions for the APE testbed [14]. To our knowledge there are no tools that support the specification of loose scenarios.

2.3 Node Configuration

When the scenario is prepared, the nodes need to be configured with the information required to run the experiment. This includes the implementation of the investigated algorithms as well as the specification of the actions and the movements of each node. This step mainly comprises the distribution of files and the configuration of nodes (e.g., setting of addresses), thus it should be automated as much as possible.

The key to the autonomous configuration of the nodes is the experiment specification. Since this specification contains any relevant information on how each node should behave, a good testbed will be able to install the required software and perform the necessary configuration based solely on this information. This can either be done by directly distributing the specification to each node or it can require its "compilation" to gain configuration files that are specific for each node.

If the nodes are physically accessible to the experimenter, the automatic file distribution can be provided with simple means, e.g., through a one-hop download. However, in loose scenarios the devices are normally not available for a direct download and the required files therefore have to be distributed to nodes that are already in the field. One approach to do this is to let the nodes distribute the required files amongst themselves, i.e., whenever two nodes come in radio range, they will exchange information and files on the scheduled experiments.

Automatic configuration is a concept used in APE [14], the PRNET project [9] provided a mechanism for the remote configuration of nodes.

2.4 Verification of the Setup

The actual execution of an experiment that uses a strict scenario is extremely costly in terms of manpower and time. A verification of the test setup and the used hardware in the forefield of the experiment in a controlled laboratory environment is therefore vital and should be supported by the testbed.

The verification can be divided into tests involving one or multiple devices. Single device tests allow to avoid problems such as lack of memory, low battery power or physical damages. Tests with multiple devices can reveal problems only occuring due to the interaction between devices. An important multiple device test which should always precede an experiment is running the complete setup installed on real devices under laboratory conditions. Although these artificial conditions prevent the acquisition of quantitative results, the setup is not expensive, can be repeated easily, and allows the isolation of errors.

There exist a number of approaches on how to create a multi-hop network topology for the in-lab verification, e.g signal attenuation [10] or *MAC filters* that drop packets based on the MAC source address of packets and the virtual position of the sender [15, 22]. A good testbed can use the position information in the scenario file to compute the virtual distances and control the topology accordingly.

2.5 Support during the Experiment

The main phase of the experiment starts with the distribution of the devices. Each experiment will most likely consist of several runs in which the nodes move around. Finally, the devices need to be collected and the phase is concluded by collecting the tracefiles from the devices. The main phase has some properties which necessitate a dedicated support by the testbed: 1) The time in this phase is expensive, only an optimal usage of the experimental time makes experiments economically feasible; 2) Repeatability of this phase is crucial for a scientific evaluation; 3) All information available here is valuable; Therefore, the testbed should support the experiment by optimizing the usage of experimental time, by guaranteeing, or at least fostering, repeatability and by collecting detailed information on the nodes' actions.

The usage of the experimental time can be optimized by automating tasks and by avoiding errors and therefore unnecessary repetitions. As device distribution and collection are physical tasks, the potential

for automation is small, here. This is different with tasks not requiring a direct (human) interaction like the tracefile collection or the movement of the nodes which can be automated by mounting the nodes on robots [3]. A large optimization potential also lies in the avoidance of the execution of erroneous experiments. By controlling that all nodes act within the parameters specified in the scenario, the testbed should assure that exactly the intended experiment is executed.

The repeatability of an experiment is provided if it is possible to rerun the same experiment such that the relevant parameters in both runs have sufficiently similar values. There are two ways to support repeatable experiments, either by comparing the parameters after an experiment to determine if it was a repetition of a prior experiment or by steering the experiments to ensure that these parameters lie within an acceptable threshold. Open issues in this context are the determination of the relevant parameters and the question if it is technically and economically possible to record these parameters.

For all aspects mentioned so far it is crucial to trace the data on the behavior of nodes and on external influences as completely as possible. This data can be used for a detailed post-run analysis as well as for the steering of the experiment. The data to be recorded involves packet-level traces, timing and positioning information, states of higher level protocols as well as physical and MAC layer logging.

To steer the experiment, the *experiment control* component of the testbed should continuously compare the actual values of the relevant parameters to those specified in the scenario. The testbed therefore should provide a method to specify and control boundaries for these parameters, soft boundaries like "position between x-5 and x+5" as well as hard boundaries like "GPS daemon running". In case some of these boundaries are violated, the testbed can adjust the behavior of the node during the run or mark the run as invalid. If the violation is severe, this can render the whole experiment unusable and should therefore be known right during the experiment. Thus, the testbed should support the transmission of status information to a central monitor station and it should also be possbile to remotely login to the affected node to correct errors or alter the configuration.

The experiment monitoring and the remote login necessitate the exchange of management messages between the nodes and the monitoring station, possibly during the experiment. This counteracts a primary design goal of the testbed, i.e., minimizing the interference of the test equipment (both hard- and software) with the experiment. A solution to this is the "out-of-band"-transmission of all management messages. This can be achieved either via a separate network interfaces during the experiment, as payload of experiment packets with a dummy payload or in the pauses between the single runs of an experiment using the tested network itself. Although the last method does not allow to stop erroneous runs directly, this is not problematic if runs are short. Furthermore, it is more practical than the other methods as the first method is more expensive and the second not always feasible.

The concept of monitoring the state of the network during the experiment has been used in the PRNET Network Monitoring [9], the SURAN Automated Network Manager [1], the CMU Position and Communication Tracking daemon [15] and ATMA [18]. For some of the tracing tasks, there exist standard tools like tcpdump [21] for packet level tracing or gpsd [6] which can help to record the node position. A means to synchronize the clocks of the nodes is the usage of a combination of NTP and GPS. The concept of determining if an experiment is a repetition of another one by hindsight has been used in the APE virtual mobility [14]. To our knowledge, there exists no experiment control that adapts the nodes' behavior dynamically to changes in the environment to foster repeatability.

2.6 Postprocessing of the Experiment

The postprocessing can be divided into organizing the raw data gathered during the experiment and analysing it. This phase should be governed by the principle that the raw data is a valuable resource. It needs to be documented, stored and published. Based on this data, independent researchers must be able to verify any conclusions that are drawn from the experiments. A good testbed will provide mechanisms to ease the documentation, storage and publication of the involved files. Furthermore it will provide or incorporate an extensible toolset for analysing the raw data.

The first postprocessing step is the structured, permanent storage of all raw data. As there are also a lot of other files involved in the postprocessing such as scenario files or tools, the testbed's automatic file handling should include these. One possibility is the implementation of a file management framework that defines interfaces to access, view, annotate, process, and store these files. The organization of the raw data is concluded by the documentation of the events and conditions not recorded in the traces but perceived by the human participants.

The tools used to process the raw data provide functionality for consistency checks, data analysis or for enabling trace-based simulation. All these tools can be used for multiple experiments, thus the testbed should foster reusability to reduce work. If the tools are modular and reusable, this also increases reliability as results can be easily reproduced. Therefore the goal has to be the creation of a reusable standard toolset for the postprocessing of MANET experiments which should be publically available, extensible and well documented. The analysis tools should support the different input formats of real-world traces and simulator traces as this allows a direct comparison of results from both evaluation methods.

The file management as well as the toolset can be combined with a server that is publically available. If software tools and raw data are available in an open repository, other researchers can access and reuse the data and tools or verify the results. We are not aware of a toolset or testbed that supports the properties described above.

2.7 Orthogonal Concepts

The previous sections show how a testbed can support the single steps of an experiment. Besides that, there are general concepts such as error avoidance, reduction of workload, portability, modularity and benchmarking not limited to single steps.

Benchmarks provide references to improve comparability. One option to realize benchmarks for MANETs are *baseline protocols*. A baseline protocol must 1) have the best performance in its class, typically achieved through the use of "illegal" means such as global knowledge; 2) be easy to implement; 3) be easy to test; If the baseline protocol is included in the experiment, it builds an upper bound for the performance of the whole class of algorithms in this special setting. Instead of giving absolute numbers for the performance of an evaluated protocol, it can be expressed as percentage of the maximum. Candidates for baseline routing protocols are MERIT [4] or the "best-case" routing described in [22] which both use global knowledge to make routing decisions.

Another form of benchmarks are *standard tests* and *standard measure values*. Standard tests can be standard topologies such as chains or grids as well as tests on the maximum load the network can support or the load without data traffic. Standard measure values provide characteristics of a protocol. Examples are delay, packet order, route length or loss rate.

To reduce the workload and the errors arising due to reimplementing the same code multiple times, using the same source code for simulation, emulation, and real-world experiments is highly desirable. We call this SER integration. An additional advantage of this approach is the feedback that can be given between simulation and experiment. SER integration should be used for the evaluated algorithms as well as for scenario files and analysis tools. This means that the same scenario file should be able to drive a simulation as well as an experiment or that the output of both steps can be processed with the same analysis script.

Portability of the testbed software is very important since a test setup used for large MANET experiments will consist of heterogenous devices. As the full number of devices is only needed for a short time, sharing devices among workgroups and using devices like laptops or cell phones not especially bought for experimenting will be common in experiments. Furthermore not all workgroups will buy the same devices, leading to further heterogeneity. Apart from this, the product live cycle of mobile devices is short. If a device is bought today, it may be not possible to buy the same product in a year. Thus, the testbed software must be very portable to run on these different platforms.

Finally, it seems beneficial to employ a highly modular testbed architecture rather than a monolithic approach. This mainly is due to the large selection of tools allready available, including standard software such as tcpdump [21] or gpsd [6]. These tools should be used rather than reimplemented. In addition modularity will allow the easy exchange of components to

enable a competition on the level of individual components rather than complete testbeds. As a consequence we believe that the system should be only loosely coupled with some kind of "glue" combining these components to form the testbed. An example where this concept has been used to some extend are the scripts controlling APE [14].

3 Experiences

Since 2002 we have conducted experiments on real-world vehicular ad-hoc networks within the context of the FleetNet project [7]. The experiences gained through these experiments and the problems we encountered motivated us to investigate how to improve experimentation with real-world implementations of ad-hoc networks in general.

In a first step we tried to repeat experiments conducted by other research groups to confirm their results. This turned out to be extraordinarily difficult since there is almost always insufficient information available to reproduce the described results. Key issues were the lack of information on the environment (in particular regarding the setup of the experiment, the radio characteristics and the connectivity between nodes), no access to the raw data gathered during the experiment and a vast heterogeneity in the tools to setup, conduct and evaluate an experiment. These experiences led us to the first attributes of a good testbed: all information, code and tools need to be published, preferably in a standardized way.

We have conducted in- and outdoor experiments of a simple flooding algorithm for static ad-hoc networks with seven to ten nodes. The motivation for this very simple setup was to isolate problems that would lead us to general design criteria for experiments with mobile ad-hoc networks. In the following we sketch the main observations, more details can be found in [20].

Inspired by [15], we started by measuring the radio ranges of our hardware (iPAQ5550 IEEE 802.11b) and discovered that the iPAQ radios were sometimes able to successfully deliver ping packets over more than 900 m while already a tree in the line-of-sight between two nodes can block a transmission. Thus, setting up a reliable, reconstructible 7-node/6-hop string topology for preparatory tests was only possible by carefully positioning each device around the edges of a building.

Our next step was to set up a multi-hop topology where every node had multiple neighbors. After several measurement sessions, a suitable experimental site seemed to be the university parking lot. Later on we discovered some undesirable properties of this location. As the library and other university buildings are close, there were other WLANs present requiring the careful selection of the radio channel (a problem also described in [18]). Another issue were moving cars possibly leading to frequent changes in the topology even though the nodes themselves did not move.

To determine the connectivity between the nodes at the start of an experiment, each node transmitted a number of beacons. While one node transmits its beacon at a time, all other nodes record the packet reception. The necessary exact coordination is difficult in a distributed ad-hoc network but can be achieved with a *domino effect*. Prior to the experiment a route is determined that includes all nodes of the network. The nodes use the sequence imposed by this route to coordinate their beaconing. The first node starts with its beaconing. Its successor will take over once this node has finished. This is repeated until the last node has transmitted its beacons. The domino approach worked well for our small networks, however it can be foreseen that a more sophisticated mechanism is required for larger and more dynamic networks.

Based on the experiences gathered during the FleetNet project, we were aware of the problem of having to reimplement the algorithms when switching between simulations and experiments. Therefore, we implemented flooding in click [12]. With the help of nsclick [16], the same code used for the experiment can be run in ns-2 [17]. We used the integration for two purposes: the first is to debug and gain first experiences with the implementation and the setup of the experiment in a controlled simulator environment. When experiments are conducted without proper tests, this can waste a huge amount of manpower: a bug in the configuration of one of our in-building tests rendered a whole series of test runs useless. A miscalculation of the pause time between sending packets resulted in intermixed flooding attempts. A simulation prior to the testing could have revealed this problem without incurring significant overhead. The second reason for an integrated simulation and experimentation approach is to use the data gathered during an experiment to im-

prove simulations with information about real-world radio characteristics. To model the real setup as exactly as possible, also the positions of the nodes in the real experiment must be known. We used GPS for this purpose and encountered the well known problems of position jitter and dependence on clear view to the sky.

During our experiments it often happened that either a link, the used software or a whole node failed. Every time this happened, we had to check each node manually. As a provisional solution, we implemented a simple in-band one-hop status check. Each node wrote its current status to a file accessible via HTTP. To control the nodes' status, it was sufficient to walk around and use a perl script to retrieve the status file from each node. Obviously, this approach has several limitations: it requires to walk in the radio range of each node to be checked, the transmissions "contaminate" the experimental data, and the correction of an error still requires physical access to the affected node. Because of these experiences we believe that node monitoring will be a very valuable element of a good testbed.

Another issue appeared during the postprocessing of the data from the experiments and the simulations. As the output format of the simulator (ns-2) differs from the trace format of the experiment (tcpdump), each tool for the analysis of the results had to be implemented twice. Furthermore there exists currently no good solution for commentation and documentation of the raw data such that special events during the experiments can later on be remembered and reconstructed.

One key question occurred during the postprocessing: how good was the performance of the simple flooding strategy? Of course there are absolute values on the number of transmitted packets and the rate of received messages. However, this does not provide any clue on how good our approach worked under the given circumstances in comparison to an (illegal) optimal solution. Thus benchmarks would be a great help in determining how well a given solution performed.

4 Conclusions and Outlook

We believe that a MANET testbed should be open source, not restricted to special hardware, customizable, and not bound to any specific location. It must support reproducible, comprehensive, and correct experiments. While there are promising individual contributions towards this goal, currently no approach fully satisfies these demands. For a multitude of reasons the best way to implement such a testbed seems to be a joint effort of the MANET community. The most important one may be a broad acceptance by the community.

A first step towards this goal should be a specification of the functionality that the testbed must provide. We expect that this will be a controversial discussion and hope that this paper may contribute to this effort. Once the specification has become stable, the key factor to a successfull community effort will be the design of a highly modular system where each research group can contribute individual parts. We expect that the specification of scenarios, the scripting for running experiments and the format of the raw experiment data will require immediate attention and provide the glue for the connection of the individual contributions. In order to stimulate these first steps we have set up a wiki at *http://wikicn.cs.uni-duesseldorf.de*.

References

[1] D. A. Beyer. Accomplishments of the DARPA SURAN program. In *Proceedings of MILCOM'90*. IEEE, Sept 1990.

[2] C. T. Calafate, R. G. Garcia, and P. Manzoni. Optimizing the implementation of a MANET routing protocol in a heterogeneous environment. In *The Eighth IEEE Symposium on Computers and Communications*, June 2003.

[3] Emulab - mobile wireless networking. http://-www.emulab.net/tutorial/mobilewireless.php3.

[4] A. Farago and V. R. Syrotiuk. Merit: A unified framework for routing protocol assessment in mobile ad hoc networks. In *Proceedings of MobiCom'01*, pages 53–60. ACM Press, 2001.

[5] J. Garcia-Luna-Aceves. Wireless internet gateways (WINGs) for the internet. Technical report, University of California, Santa Cruz, 2001.

[6] gpsd: a GPS service daemon. http://gpsd.-berlios.de/.

[7] H. Hartenstein, B. Bochow, A. Ebner, M. Lott, M. Radimirsch, and D. Vollmer. Position-aware ad hoc wireless networks for inter-vehicle communications: The FleetNet project. In *Proceedings of MobiHoc'01*, Long Beach, California, October 2001.

[8] S. Jadhav, T. Brown, S. Doshi, D. Henkel, and R. Thekkekunnel. Lessons learned constructing a wireless ad hoc network test bed. 1st Workshop on Wireless Network Measurements (WINMee 2005), April 2005.

[9] J. Jubin and J. D. Turnow. The DARPA packet radio network protocols. In *Proceedings of the IEEE*, volume 75, pages 21–32, January 1987.

[10] G. Judd and P. Steenkiste. Repeatable and realistic wireless experimentation through physical emulation. *ACM SIGCOMM Computer Communication Review (CCR)*, 34(1):63–68, 2004.

[11] V. Kawadia, Y. Zhang, and B. Gupta. System services for ad-hoc routing: Architecture, implementation and experiences. In *Proceedings of MobiSys'03*, San Francisco, California, May 2003.

[12] E. Kohler, R. Morris, B. Chen, and J. Jannotti. The Click Modular Router. *ACM Transactions on Computer Systems*, 18(3):263–297, August 2000.

[13] D. Kotz, C. Newport, R. S. Gray, J. Liu, Y. Yuan, and C. Elliott. Experimental evaluation of wireless simulation assumptions. In *Proceedings of MSWiM'04*, pages 78–82, October 2004.

[14] H. Lundgren, D. Lundberg, J. Nielsen, E. Nordström, and C. Tschudin. A large-scale testbed for reproducible Ad Hoc protocol evaluations. In *Proceedings of WCNC'02*, pages 337–343, Orlando, FL, March 2002.

[15] D. A. Maltz, J. Broch, and D. B. Johnson. Experiences designing and building a multi-hop wireless ad hoc network testbed. Technical Report CMU-CS-99-116, School of Computer Science, Carnegie Mellon University, 1999.

[16] M. Neufeld, A. Jain, and D. Grunwald. Nsclick: Bridging Network Simulation and Deployment. In *Proceedings of MSWiM'02*, pages 74–81, Atlanta, Georgia, September 2002.

[17] The ns-2 network simulator. http://www.isi.edu/-nsnam/ns/.

[18] K. Ramachandran, K. Almeroth, and E. Belding-Royer. A novel framework for the management of large-scale wireless network testbeds. 1st Workshop on Wireless Network Measurements (WINMee 2005), April 2005.

[19] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh. Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols. In *Proceedings of WCNC'05*, New Orleans, LA, March 2005. IEEE. (to appear).

[20] A. Tarp. Experimental evaluation of flooding in ad-hoc networks. Bachelor thesis, 2005. Department of Computer Science, University of Düsseldorf.

[21] tcpdump: a tool for network monitoring, protocol debugging and data acquisition. http://www.-tcpdump.org.

[22] Y. Zhang and W. Li. An integrated environment for testing mobile ad-hoc networks. In *Proceedings of MobiHoc'02*, pages 104–111. ACM Press, 2002.

[23] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic. Impact of radio irregularity on wireless sensor networks. In *Proceedings of MobiSys'04*, pages 125–138. ACM Press, May 2004.