Evaluating Wireless Multi-hop Networks Using a Combination of Simulation, Emulation and Real World Experiments

Erik Nordström, Per Gunningberg, Christian Rohner and Oskar Wibling

Department of Information Technology, Uppsala University

Abstract—Mobile ad hoc networks suffer from complex interactions between protocols on different levels. To identify and explain anomalies found in real world experiments we propose to combine real world testing with simulation and emulation. By inspecting the differences in the results obtained using the three evaluation methods we can efficiently identify when anomalies occur and get a hint about where to further investigate. Our approach has proved to be valuable in an extensive comparison of the routing protocols AODV, DSR, and OLSR. The study resulted in two gigabytes of log data that required an efficient and structured method to isolate important time regions for further manual analysis. We discuss our approach and present some key findings.

I. INTRODUCTION

Evaluation of mobile ad hoc routing protocols is often done in either simulation, emulation, or real world testing. All of these methods have their specific advantages and disadvantages. Simulations usually assume simplified models with idealized conditions. Emulations consider real network stacks but still allow for controlled mobility and a semi real radio environment. These elements are included in real world tests leading to a complex environment and usually quite some effort to perform the tests.

In this paper we propose to combine these three methods of evaluation to identify instances for closer evaluation, by looking for discrepancies in the behavior of the evaluated protocols. Our approach proved to be useful in a study of the protocols AODV, DSR, and OLSR in different scenarios and with different traffic types.

Performance discrepancies are often not obvious when only looking to one of the evaluation methods. By comparing the results of the three evaluation methods we can in particular identify the first time of occurrence of an incident, and also get a hint about the cause of that incident by evaluating between what evaluation method the discrepancy occurs.

The target protocols chosen to evaluate our methodology are well known wireless multi-hop routing protocols. They have all been part of the IETF standards track and represent very different design choices. Our study is motivated by the fact that these routing protocols, despite their maturity, are only recently being used in the real world, for example in the increasingly popular area of mesh networks. Although they have been experimented with in such scenarios, there are no side-by-side comparisons and few studies include mobility.

Real world side-by-side comparisons are particularly demanding, because they require that experiments are repeatable. Repeatable experiments ensure that inherent differences in the protocols are exposed instead of effects from random factors and that the results are conclusive. Furthermore, current studies [5], [9] do not expose the dynamic properties of the protocols, either because the scenarios are static or the protocols only implement a limited set of features that limit their efficiency in a mobile environment.

We use a scenario based approach for our cross-environment study. A set of carefully designed and controlled scenarios have been reconstructed in each environment. Previous work that couples real world and simulation [12], [14], compare protocols in simulation and real world by feeding (e.g., GPS) traces into a simulator. Also distinguishing from other work is the extent of our measurement study. There are 27 combinations of three scenarios, using UDP, Ping and TCP traffic and AODV, DSR and OLSR routing. Each experiment combination was repeated 10 times, resulting in 270 independent experiments. Our structured and tool-supported approach has enabled us to perform an experiment of this magnitude. The complete traces comprise around two Gbytes of data and will be made available for other researchers along with all software.

II. EXPERIMENTAL SETUP AND METHODOLOGY

A. Coupling the Real World, Emulation and Simulation

We use the same protocol implementations that run natively in real world, for all three environments. This contrasts to previous studies which have relied on implementations that use emulation or translation layers to be able to run simulator code in the real world [20], [17], [10]. These approaches suffer from considerable overhead and sometimes require specific scheduling between real time and simulator time, which increases the uncertainty in the results and the conclusions.

Table I lists a number of factors in which the real world, emulation and simulation differ for our experiments. We know from previous work that the radio is an important factor for explaining performance discrepancies between the platforms [14]. In order to study its impact we try to control the other factors. Mobility is handled with choreography and scenarios. Hardware and software are identical while protocol logic is varied with the routing protocols, but otherwise the same between the platforms. Through this harmonization we can
use simulation and emulation as a baseline for the protocols and gradually expose the radio factors of the real world.

<table>
<thead>
<tr>
<th>Environment</th>
<th>Routing Logic</th>
<th>HW</th>
<th>Stack</th>
<th>Mobility</th>
<th>Radio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real World</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Emulation</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>Partial</td>
<td>Partial</td>
</tr>
<tr>
<td>Simulation</td>
<td>✓</td>
<td>x</td>
<td>Model</td>
<td>Model</td>
<td>Model</td>
</tr>
</tbody>
</table>

TABLE I
RELATIONSHIP BETWEEN THE REAL WORLD, EMULATION AND SIMULATION IN TERMS OF ROUTING LOGIC, HARDWARE, NETWORKING STACK, MOBILITY AND RADIO ENVIRONMENT. A ✓ MEANS THAT THE ELEMENT IS GENUINE AND NOT A MODEL OR, BY RESTRICTIONS, PARTIAL. A x INDICATES ABSENCE OF AN ELEMENT.

The real world experiments feature people that carry laptops and move according to a scenario choreography that is displayed on the screens. All experiments take place in our building, so the test environment consists of offices and corridors.

Both the emulated and the real world experiments use the Ad hoc Protocol Evaluation (APE) testbed [15], which is publicly available. APE is driven by scenario descriptions to ensure that the mobility pattern is repeatable from one experiment to another, modulo the natural variance caused by people moving around with laptops. We have run complementary experiments to quantify this variance, by varying the mobility artificially and by having people interfere. The results show that this variance is not significant for the experiments in this paper.

When an experiment is conducted, APE reads commands from a scenario file and executes them at the specified time. The scenarios schedule the traffic load and contain instructions to the persons carrying the laptops. As traffic we use either synchronous UDP or Ping traffic (CBR), or a TCP file transfer. APE logs all traffic seen by all nodes during the experiments. By matching the time stamps of all logged packets from all nodes after the experiment, we get a complete global view of the whole experiment. The bulk of our post experiment analysis is based on this information. The logging adds overhead which has some effect on TCP throughput. Our UDP and Ping measurements are not affected by this logging.

All APE computers are identically configured IBM Thinkpad X31 laptops. APE version 0.5, built with Linux kernel 2.6.9 is used for all experiments. The WiFi interfaces are PC-card based Lucent (Orinoco) silver cards supporting the IEEE 802.11b standard. The cards use the Agere Systems Linux driver version 7.18 (March 2004), which we have updated to support Linux kernel 2.6 and an updated wireless extension API. This driver comes with its own firmware that is dynamically loaded onto the card at initialization. All our experiments are run with the driver set to 11Mbps fixed rate with RTS/CTS turned off. See for example [21] for a motivation to this choice of RTS/CTS setting.

1) Emulation: The emulations use the same HW/SW platform as the real world experiments including the wireless cards. Nodes are stationary and in close proximity, e.g., in the same room and their radios will intentionally interfere with each other. This type of emulation is relative simple, but also quite common and allows comparisons to previous work [10]. The connectivity changes, due to mobility, are emulated using MAC filters by selectively filtering traffic between nodes. The times to enable and disable the filters are extracted from traces generated in the real world experiments. A connectivity change matches the time when the real world signal strengths causes a change of connectivity. We do not introduce any variance in this connectivity time.

This filtering schedule is added to the APE scenario schedule making the connectivity changes completely predictable. The channel quality is high and stable until a change. In addition to being predictable, the approach eliminates the impact of, for example, gray zones [16] when the signal strength is so weak that the connectivity fluctuates and there are other radio propagation phenomena that degrade the radio channel. The emulation results can therefore, when compared to the real world results, give an indication of the impact of these phenomena on the different routing protocols. Although nodes are stationary in the emulation, it is important to observe that there are still some internal and external radio interferences that impact the experiments. However, our measurements show that this variance is negligible in our context.

2) Simulation: For the simulation we use ns-2 version 2.29. We recreate mobility in ns-2 to match the scenarios from APE (by translation into a schedule). Nodes are configured in a chain topology with logical placement and distance matching real world measurements. The node movement speed is programmed to be between 1.33 ± 0.0125 m/s. This gives a variance in the times when each waypoint is passed of up to two seconds. In the real world experiments, each waypoint is reached within 1-2 seconds of the scripted time.

The choice of a radio propagation model to match the actual environment is delicate. We settled on using the standard ns-2 TwoRayGround model to be comparable to other simulation studies and to determine whether this commonly used model can be used to predict the real world performance of our routing protocols. However, to make this simple model match our experimental indoor set-up better, we tuned the WiFi transmission range to 45m. It is slightly longer than the measured average value. The real values vary, of course, much more unpredictably with the actual building layout. We believe that the simulations still provide a convincing reference to the emulation and real world experiments, because they let us isolate problems that are not radio specific. The chosen parameters for the radio model are listed in Table II.

B. Scenario Descriptions

Our study comprises three mobility scenarios: End node swap, Relay swap and Roaming node. They are choreographed to test different aspects of the routing protocols. Their simplicity makes it feasible to recreate them in the simulation
and emulation environments. They are thus limited to four nodes and three hops, which is the minimum size multi-hop network for which interesting and repeatable real world mobility patterns can be achieved. The results show that routing protocols have performance problems with our small scale scenarios. Therefore, it does not make sense to scale the scenarios until the problems in the smaller scenarios are solved and the protocols achieve reasonable performance. Furthermore, a larger scenario would be considerably more complex to analyze in depth, more challenging for repeatability and extremely time consuming.

Figure 1 depicts logical overviews of the scenarios. Positions A, B, C and D correspond to physical locations in which nodes only have connectivity to their adjacent neighbors. In all experiments, there is only one traffic stream between nodes 3 and 0, consisting of either UDP packets, Ping messages or TCP segments. The scenarios are constructed so that there is always connectivity between the source node (3) and the destination node (0) over one or more hops. Nodes move at normal walking speed. We measured it to be about 1.3 m/s. Each scenario has a warm-up phase and a cool-down phase of at least 10 seconds, during which the routing protocols have time to converge (in the case of OLSR) or, in the case of cooldown, to deliver delayed data packets before the experiment ends. The traffic streams start some time after the warm-up phase, depending on scenario.

These scenarios are selected since they stress the ability of the routing protocols to adapt to different situations as discussed below.

The End node swap scenario (Figure 1 a) aims to test a routing protocol’s ability to adapt when both source and destination move and the shortest path changes from three hops, through two hops, to one hop and back. At time 31s, data transmission from node 3 to 0 starts. At time 51s, end nodes 0 and 3 start to move toward the other end node’s position (A and D) where they reach at time 113s. Nodes 1 and 2 are stationary during the course of the scenario.

The Relay node swap scenario (Figure 1 b) instead tests how a routing protocol handles mobility among intermediate nodes while the end nodes are stationary. The traffic is initiated between node 3 and node 0 at time 61s, the relay nodes 1 and 2 start to change positions at time 81s. When they meet in the middle, our node placement allows, depending on the current connectivity, a two hop route between end nodes 0 and 3 using either one of the relay nodes as an intermediary. The relay nodes reach their destinations at time 101s.

The Roaming node scenario (Figure 1 c) starts with one hop between node pairs 0 and 3, in contrast to the other scenarios. Also, there is no movement among potential relay nodes. Instead, node 3 “roams” the network, moving from position A to position D and back during the course of the scenario. All other nodes are stationary and only forward traffic. When initiating the traffic at time 26s, node 3 starts its movement from position A toward position D. At time 88s, node 3 has reached position D and heads back toward position A, which it reaches at time 150s. The Roaming node scenario aims to mimic, for example, a mesh network where a user (node 3) is mobile and communicates with a gateway (node 0). Here we study the effect of increasing path length as well as the route optimization behavior when node 3 moves back.

C. Routing Protocols

The MANET working group [1] intends to standardize one reactive and one proactive protocol based on AODV, DSR and OLSR. Current candidates are DYMO [7] and OLSRv2 [4]. There are two main reasons why these two protocols are not in our study. First, they are not yet as mature, e.g., in terms of implementations. Second, DYMO and OLSRv2 are evolutionary steps from AODV and OLSR, mainly differing in packet header format. Therefore, we anticipate that by closely examining AODV, DSR and OLSR\textsuperscript{1}, there could be valuable input for the design choices of both DYMO and OLSRv2. Furthermore, AODV, OLSR and DSR represent very different design choices and vary in how dynamic they are. Because of

\begin{table}[h]
\centering
\begin{tabular}{|c|c|}
\hline
Parameter & Value \\
\hline
\text{Pt} & 0.031622777 \\
\text{freq} & 2.472e9 \\
\text{CSSThresh} & 5.011872e-12 \\
\text{RXThresh} & 1.45647e-09 \\
\text{RTSThreshold} & 2000 \\
\hline
\end{tabular}
\caption*{Simulation parameters to mimic an 802.11b WiFi card with a transmit radius of 45 meters (indoors) and RTS/CTS turned off.}
\end{table}
E. Metrics

The following metrics are used in our analysis:

- **Delivery Ratio**: The number of packets received divided by the number of packets generated during an experiment.

- **Throughput**: The number of bytes useful data delivered divided by the time over which data is sent. This is also referred to as **Goodput**.

- **Latency standard deviation (σ)**: The variation in time for a packet traveling from the sender to the receiver. For Ping and TCP we calculate the round trip latency standard deviation. We use the standard deviation instead of the mean for two reasons: First, the standard deviation can measure the stability of routes. Second, in the case of UDP, accurate calculation of the mean is not possible due to the lack of good time synchronization between the nodes.

- **Average Hop Count**: For UDP, the average hop count is calculated from the source node to the destination node. For Ping, the hop counts for the Ping request and the Ping reply are added. Similarly, for TCP the sum of the hop counts for data packet and ACK is used.

D. Traffic Configuration

As mentioned above, all experiments have one data flow between a source node and a sink node. This flow consists of either synchronous UDP packets, Ping, or a TCP file transfer session. The CBR rate for UDP and Ping is 20 packets per second while TCP transmits with the highest achievable rate. UDP and Ping have no adaptive mechanisms such as congestion control. Therefore, UDP is used to sample the network connectivity and to measure the route latency. Ping requests are sent to the sink node, which then generates a reply packet for each received request. The request-reply mechanism is used to examine bidirectional connectivity and to measure the round trip times (RTT). TCP is used to study the effect of congestion control and reliable delivery.

We wanted to use the same packet size for all three traffic cases to minimize differences in size induced loss. We settled on 1378 bytes to allow for the DSR header in UDP and Ping. Similarly, for TCP the sum of the hop counts for data packet and ACK is used. Still, DSR pays a performance penalty for TCP due to this header overhead.

E. Metrics

The following metrics are used in our analysis:

- **Delivery Ratio**: The number of packets received divided by the number of packets generated during an experiment.

- **Throughput**: The number of bytes useful data delivered divided by the time over which data is sent. This is also referred to as **Goodput**.

- **Latency standard deviation (σ)**: The variation in time for a packet traveling from the sender to the receiver. For Ping and TCP we calculate the round trip latency standard deviation. We use the standard deviation instead of the mean for two reasons: First, the standard deviation can measure the stability of routes. Second, in the case of UDP, accurate calculation of the mean is not possible due to the lack of good time synchronization between the nodes.

- **Average Hop Count**: For UDP, the average hop count is calculated from the source node to the destination node. For Ping, the hop counts for the Ping request and the Ping reply are added. Similarly, for TCP the sum of the hop counts for data packet and ACK is used.

III. RESULTS AND DISCUSSION

A. Overview

Our result constitutes an efficient evaluation method for wireless multi-hop network protocol implementations, demonstrated by a real world performance study of AODV, OLSR, and DSR. A presentation of the full set of measurements is out of the scope of this paper. Here, we demonstrate how we take advantage of our evaluation method in the analysis of one of our experiments.

For each protocol, scenario, and traffic type we made 10 test runs which were averaged. The variance and min/max values were also calculated. For simulation, we generated the corresponding values. The emulation results, on the other hand, are from single experiments since the variance between tests using UDP and Ping was negligible with the deterministic connectivity used. Note that the emulation with TCP transfers suffers from contention limitations because nodes interfere. We chose to include those tests anyway for completeness. The real world experiments further suffer from the overhead of logging. Therefore, the TCP results could also only be compared over the routing protocols.

Because our scenarios are constructed with a potential path between source and destination at all times, the protocols should achieve high delivery ratios under ideal circumstances. AODV and DSR consistently achieve over 90% packet delivery ratio for all scenarios in simulation and emulation, when using UDP or Ping traffic. OLSR achieves slightly lower ratios at 80-90%, which is expected due to its slower convergence. Since the performance is good in both environments we can, with some confidence, exclude any serious routing logic problems as a source of errors.

B. Comparing simulation, emulation, and real world results

To illustrate the nature of our measurement data and also give an example of the use of our method, consider the graphs in Figure 2. These depict our UDP packet delivery measurements in the End node swap scenario using simulation, emulation and real world testing respectively.

We note that the simulation and emulation plots are very similar up to time 99. After that there are some differences due to the differing environments. In the emulation we have added a real networking stack, semi-real radio-environment and emulated node mobility. One example of a discrepancy to study more closely is why DSR achieves slightly higher total UDP throughput in emulation than in simulation. We see a couple of interesting time slots at approximately time 103-105 and 109-112 respectively. At those intervals there are “knees” in the simulation curve indicating that no UDP packets were delivered between those times. In the second case it is especially odd since the topology should not be fluctuating during this time. These places are thus good candidates to start investigating the discrepancy.

Similarly, with the real-world tests we further add the factors of real mobility and radio environment. Hence, before time 99, since we do not see a difference between simulation and emulation, but there is a significant difference to the real world situation we can conclude that a dynamically changing radio environment has had an important effect on protocol behavior. An example of an interesting time period to investigate closer is between approximately time 59-64 when there are “knees” in both the OLSR and AODV curves. This is a period when...
the topology changes rapidly, going from a three hop distance between the end nodes, to two hops, and then to one. We are thus expecting fluctuating connectivity when exposed to a real radio environment, causing problems for the routing protocols in setting up and maintaining routes.

The purpose of this section has been to show how we use our plots to assist us in selecting time periods for closer manual inspection. We have done this on the measurement data from our study and have been able to make several important discoveries about protocol behavior. To summarize, one important finding from using our method is that latency and timing issues play an important role for routing in highly dynamic environments. Protocols that are reactive (AODV and DSR) and at the same time rely on buffering during times of disconnection may cause queue build-up in the network that affects the timing and ordering of control messages, severely reducing the efficiency of the protocols. Because of the event-driven model and simplified queues in simulation, these timing issues are not apparent there. Emulation is not as affected either, because the ideal connectivity and swift changes in topology do not cause queue build-up to the extent that the protocols become affected.

IV. RELATED WORK

Grey et al. compare in [10] the routing protocols APRL, AODV, ODMRP, and STARA in a thirty-three node outdoor testbed. They use GPS to collect movement traces from experiments in an open field where 40 people walk around randomly with laptops. The traces are later fed into a “tabletop” emulation and a simulator. They use direct execution to allow protocols developed in simulation to run in the real world, similarly to the work of Saha et al. [20] as well as the nslick [17] project. In contrast to our code, packets are forwarded in user space and separate event-loops and scheduling increase the overhead. While our approach is scenario based, they instead use a larger scale network with random mobility and random traffic using only UDP. Each routing protocol is run separately and subjected to different mobility and traffic. Their focus is instead on validating different propagation models in simulation, which is the topic of a follow-up paper by Liu et al. [14].

Haq and Kunz [11] have evaluated OLSR using two different simulators as well as an emulated testbed. They study the total number of successfully transmitted packets using CBR traffic (UDP) at two different rates and two different packet sizes. The authors use a single scenario with five nodes and report that at low traffic rates, testbed results match closely with those from simulation. However, at higher rates they see very significant differences. Apart from providing a much more extensive study in terms of ad hoc routing protocols, scenarios and traffic types we combine simulation, emulation and real world testing. Haq and Kunz have further only studied the total number of packets received whereas we look at protocol behaviors during the whole scenario and also measure, e.g., latencies.

Johnson [12] recorded traffic traces from laptops, running DSR, mounted in cars whose positions were constantly logged using GPS. Several different traffic types were used and the collected data drove simulations as well as emulations. The author believes that simply comparing the average number of received packets from simulations and real experiments does not provide enough information to answer the question of how closely emulations come to reproducing simulation results. It can even produce an incorrect conclusion. He therefore suggests studying time-sequence number plots as well as other performance metrics over time. In our work we use different
performance metrics over time and compare simulations to emulations but also to the real world.

V. Conclusions

This paper makes two main contributions. First, we have developed a method for efficiently tracking protocol implementation problems. Second, as a proof-of-concept we describe how we implemented and took advantage of the method in a study of the three ad hoc routing protocols AODV, DSR, and OLSR.

By just looking at experimental results it is difficult to isolate the exact origins of a performance problem as, for example, software bugs cannot be excluded. We have illustrated how simulation, emulation and real-world testing can be combined into a method to efficiently expose the causes of protocol-specific behavior.

Simulation provides us with a baseline showing how each protocol can perform under ideal circumstances. We use packet delivery and time-sequence number plots to track problematic time periods. In the emulations we can thus study if the plots differ significantly from the corresponding simulations when adding a real networking stack, semi-real radio-environment and emulated node mobility. With the real-world tests we further add the factors of real mobility and radio environment. If we do not see much of a difference between simulations and emulations but the curves change significantly in the real world we know that the changing radio environment has had an important effect on the protocol. Our plots then enable us to track the first time slot where a discrepancy can be seen and carefully examine the logged data for what effect the radio environment had on the protocol at that particular time. So, then we know (a) the interaction with a real radio environment is the root of the protocol problems (b) the problem manifests itself first between times $t_1$ and $t_2$. This facilitates the manual tracking in log files. We know that we are looking to identify lost packets, e.g., HELLO messages, and we know between what time periods to search. By utilizing a combination of simulation, emulation, and real-world testing we are thus able to speed up the evaluation process considerably.

References


