The w-iLab.t Testbed

Stefan Bouckaert, Wim Vandenberghe, Bart Jooris, Ingrid Moerman, and Piet Demeester

Ghent University - IBBT, Department of Information Technology (INTEC), Gaston Crommenlaan 8, Bus 201, 9050 Ghent, Belgium
Stefan.Bouckaert@intec.ugent.be
http://www.ibcn.intec.ugent.be

Abstract. In this paper, the W-iLab.t wireless testbed is presented. The testbed consists of nearly 200 sensor nodes and an equal amount of WiFi nodes, which are installed across three floors of an office building. The testbed supports wireless sensor experiments, WiFi based mesh and ad hoc experiments, and mixed sensor/WiFi experiments. It is explained how changes in the environment of the sensor nodes can be emulated and how experiments with heterogeneous wireless nodes are enabled. Additional features of the testbed are listed and lessons learned are presented that will help researchers to construct their own testbed infrastructure or add functionality to an existing testbed. Finally, it is argued that deep analysis of unexpected testbed behavior is key to understanding the dynamics of wireless network deployments.

Keywords: testbed, wireless sensor, wireless ad hoc, wireless mesh, emulation.

1 Introduction

As a research group, frequently involved in interdisciplinary projects with industrial partners, validation of developed algorithms and protocols for wireless ad hoc, sensor and mesh networks on actual (prototype) hardware has been an important way of proving validity of theoretical and simulated novel concepts, and demonstrating the feasibility of network architectures [1,2]. Very often, our wireless experiments revealed minor or major flaws in theoretical assumptions [3], requiring time intensive debugging sessions and algorithm modifications that would not have been required if simulation results were the final product of our research.

Over the years, multiple different small scale wireless sensor and wireless mesh testbeds were set up and torn down in the scope of various projects, master theses and doctoral theses. While a lot of lessons were learned from these experiments on diverse types of hardware, there are also several drawbacks associated with the deployment of multiple individual testbeds. *(i)* Buying new hardware setups for every project is costly, and therefore limits the deployment scale. *(ii)* Different hardware architectures require different development approaches. As an example, in the case of IEEE802.11 based mesh and ad hoc research, experiments have been performed using off-the-shelf WiFi routers with custom built firmware, custom built multi-interface mesh nodes, PDAs, tablets, laptops and desktop computers with various wireless NICs, and integrated system boards. While experience with diverse network platforms is gained, there is a substantial overhead associated with creating new development environments. *(iii)* Results obtained from different test set-ups cannot easily be compared. *(iv)* Rebuilding old test set-ups is time-consuming and has a negative impact on the reproducibility of test results.

In order to overcome the drawbacks of these individual test set-ups and to enable wireless tests on a larger scale, the w-iLab.t testbed was designed and installed at the buildings of the IBCN research group and IBBT research institute in Ghent, Belgium. The w-iLab.t inherited its name from the larger IBBT iLab.t [4] test infrastructure, where the testbed is a part from. The w-iLab.t testbed consists of nearly 200 sensor nodes and an equal amount of WiFi nodes, which are mounted to the ceilings in the offices and hallways. Although the primary focus of the testbed is to support large scale wireless sensor and actuator network deployments, the testbed architecture supports WiFi mesh and ad hoc test, and mixed sensor/ad hoc experiments as well. In the remainder of this paper, the w-iLab.t testbed is presented. The design choices are motivated and the possibilities are demonstrated. Furthermore, we present lessons learned which can help testbed designers to analyze behavior of their own testbed set-up, inspire testbed administrators to add time saving functional blocks to their set-up, or act as a guideline during the initial design phase of a new testbed.

2 Goals and Requirements

One of the major drivers to perform real life experiments, is the fact that a purely mathematical or simulation based approach for designing wireless network solutions is not entirely representative for the real life performance of the same solutions when deployed in realistic environments. The reason for this discrepancy is a result of simplified traffic pattern and end user models, wrong assumptions about signal propagation and interference, interactions with other (wired or wireless) network devices and errors introduced by hardware and wireless drivers. While the latter errors should not be solved by upper layer network designers in theory, the success and applicability of a developed algorithm depends on the algorithm's ability to cope with unpredictable behavior introduced by any of the above elements.

Through careful simulations and well designed small scale testbeds, networking algorithms and protocols can efficiently be debugged to a certain extent. Multi-hop environments can be emulated on a desktop by interconnecting a small number of wireless nodes through coaxial connections, RF splitters and RF attenuators [5], without the need for a large test infrastructure. However, even with the most advanced simulation models or desktop testbeds it is hard to represent a real networking environment, especially when it comes to simulating interaction with user-level programs and operating systems, evaluating network scanning techniques and channel selection mechanisms, or when modeling dynamic network environments with moving users and external interference. Additionally, measuring user satisfaction and quality of experience is only possible with large scale testbeds deployed in a realistic environment. Therefore, similar to [6] and [7], it was chosen to install the testbed in an office environment across three $18m$ by $90m$ office floors and thus create a natural network topology. On top of this default topology, additional topology control measures (cf. Section 4.2) can be taken to vary the perceived node density in the testbed.

In addition to allowing experiments in a realistic office setting, multiple technical and practical requirements were set before designing the testbed:

- Future network environments are expected to be increasingly heterogeneous. Therefore, the testbed should support tests with wireless sensor and actuator nodes, WiFi based mesh and ad hoc nodes, and mixed scenarios. Since sensor nodes are continuously evolving, it should be possible to easily replace or install additional sensor nodes at the test locations.
- It should be possible to install new software to any sensor or mesh/ad hoc node from a remote location, and to reboot the nodes remotely in case of node failure. The nodes are preferably powered by external power sources, as to avoid the frequent replacement of batteries.
- Sensor nodes react to environmental changes. Testing protocols that depend on environmental changes is not easily done with current testbeds, as, for example, it is not very convenient to test the reaction of a protocol detecting fire through a fast rise in temperature by holding a flame close to a temperature sensor. Therefore, the testbed infrastructure should be able to emulate environmental changes instead of relying on manual interventions, without necessarily requiring specific simulation protocols to be compiled with the software under test.
- Researchers should be able to use the testbed from any location. Personalized log in is needed to provide access control and to guarantee a fair share of access to the testbed for each user.
- Advanced logging functionalities are needed, both for wireless sensor network experiments and wireless mesh and ad hoc experiments.
- Deploying the network devices at the test locations must be as fast and simple as possible, requiring the least possible number of cables to be installed in the offices, reducing the installation cost and minimizing damage to the building.

In the next section, it is explained how the w-iLab.t architecture is able to fulfill all of the above requirements.

3 Testbed Architecture

3.1 Node Location

The testbed node locations are distributed across thee similar floors of office space. Figure 1 shows the location of the nodes on the third floor. Nodes are

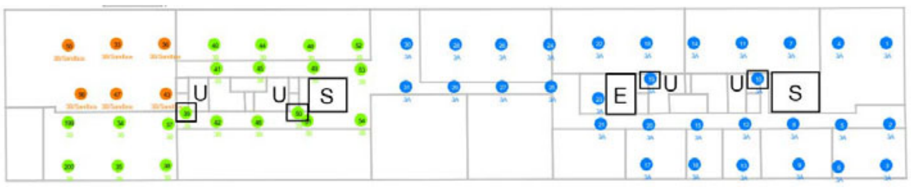

Fig. 1. Third floor of the testbed location. Office area is approximately 90m x 18m. S: staircase. E: elevator. U: utility shaft.

mounted near the ceiling of both hallways and individual offices which are separated by thermal insulated wooden walls causing little RF attenuation. Several other interesting construction elements are indicated on the floor plan: the elevator and elevator shaft are indicated by E and cause severe RF attenuation. Staircases enclosed in concrete walls (S) and concrete utility shafts (U) which run across the different floors cause an increased RF loss as well. Since the office ceiling is made of metal rasters and the floors of aluminum tiles, there is a large inter-floor signal attenuation inside the building. Therefore, it was chosen to deploy nodes in the utility shafts at every floor, thus constructing inter-floor paths with low signal attenuation.

3.2 Hardware Components and Initial Testbed Installation

The TMote Sky sensor mote, which is used as the primary type of sensor device in our testbed, is programmed through a USB interface. Since USB technology is not designed to support cable lengths longer than 3 to 5 meters without intermediate USB hubs, a large sensor network cannot be deployed in an office environment using USB cables only. In contrast, Ethernet technology allows longer cable lengths, but is not commonly supported on sensor nodes.

The chosen solution for our testbed was to deploy cheap embedded Voyage Linux operated Alix 3c3 system boards [8] at all node locations. These embedded system boards, which we call *iNodes*, are equipped with an ethernet NIC, a serial port, VGA output, compact flash storage, onboard audio, two mini PCI slots and two USB ports. Using the iNodes as relay devices allows the sensor nodes to be programmed remotely.

Two additional advantages are associated with the use of these iNodes: *(i)* by installing two miniPCI 802.11a/b/g compatible atheros based wireless NICs and adding dual band antennas, the control hardware for the sensor tests can be used as test equipment for WiFi based mesh and ad hoc tests. *(ii)* The power consumption of the iNodes is low: each iNode consumes only 6.5W in idle state, rising to 7.8W if both WiFi interfaces are enabled and continuously transmitting with a processor load of 100%. The low power consumption allows the iNodes to be powered using only power over ethernet (PoE). As such, only a single ethernet cable and a PoE converter per node are needed to power the iNodes and connect them to a central administration server. This reduces installation

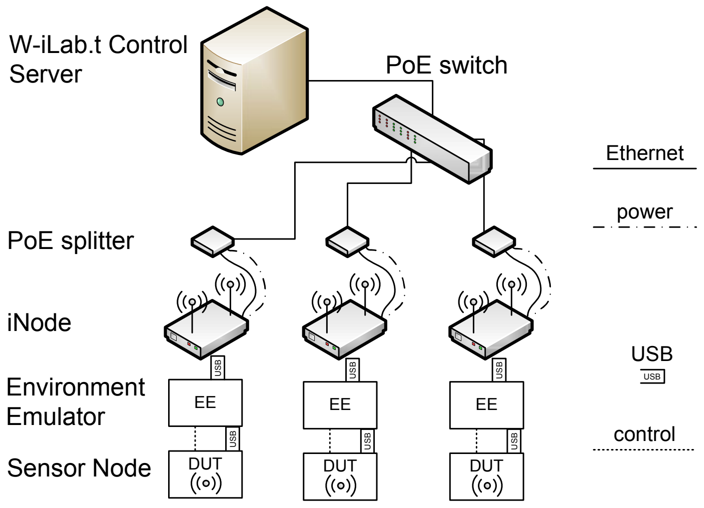

Fig. 2. Logic overview of the w-iLab.t architecture

complexity and cost, and allows for remote power switching of the iNodes, and by extension, sensor nodes.

In order to emulate changes to the physical environment of the node, an in-house designed circuit board called *environment emulator* (EE) is added between the USB port of the sensor device under test (DUT) and the iNode. The EE is built around a micro controller, a three port USB hub, and a voltage regulator/measurement chip. It is plugged into a USB port of the iNode, and is equipped with two additional USB ports. The most important goals of the EE are the following: first, one port is used to connect the DUT, the other port allows to connect additional EEs in cascade, thus allowing multiple (heterogeneous) sensor nodes to be tested using the same back-end testbed infrastructure. Second, the EE can replace the USB power from the DUT with its own internal power source. Thus, the EE is able to emulate depleting batteries, energy harvesting power sources and node failures. Third, the power that is consumed by the DUT is measured with a sample frequency of 4kHz, allowing to measure the exact power consumption of any sensor node while executing a certain protocol. Fourth, general purpose digital and analog IO pins are connected to the DUT, allowing to emulate real time digital and/or analog sensor input via programmable events. Fifth, a seven segment LED display and status LEDs provide additional feedback when e.g. flashing the sensor nodes, writing information to the logs, or during events occurring during normal node operation.

The hardware components of the w-iLab.t testbed architecture are summarized in Figure 2: the iNodes are powered and connected to a control server through a gigabit PoE switch. Two WiFi cards are installed at the iNodes, allowing to perform WiFi mesh and ad hoc experiments, and the USB ports of the iNodes are used to connect the sensor node via an environment emulator, which allows advanced testbed manipulation and logging.

3.3 Using the Testbed

Wireless Sensor and Actuator Experiments. The w-iLab.t testbed is accessible by authorized users via a web based interface, which allows users to monitor the testbed status, to upload sensor firmware, to select which nodes will be running what type of firmware during a specific experiment, to schedule an experiment at a specific time for a specific duration, to get an overview of past, current and future tests, and to retrieve results and additional information on completed tests.

The testbed is organized in several geographical zones and sub-zones such as *'third floor'*, *'first half of the third floor'* or *'entire testbed'*. The user can schedule tests in one or multiple zones, or may deploy different code on each individual node. Zone reservations are non blocking, meaning that if one user is running a test on one zone, another user might run a simultaneous test in another, non-overlapping zone. To avoid interference from other experiments, a single user can reserve the whole testbed but only use part of it.

The W-iLab.t control server software is based on the MoteLab [6] software. The software was modified and expanded to support the use of the EE and to allow a more advanced collection and easy representation of test results. Modifications include *(i)* added support for EE scenarios. The user is able to configure events to be triggered at (a selection of) EEs at a user specified time. For example, the user might specify a scenario in which several buttons are pressed at some sensor nodes, while other sensor nodes observe an emulated rise in temperature or fail because of (emulated) battery depletion. The EEs are synchronized and execute the scenario with a maximum error of $100\mu s$. *(ii)* A result processing toolbox, comprising a *sniffer*, *visualizer* and *analyzer* module. Events and sensor node logging information are stored in an SQL database together with the precise timestamps and other test data such as the individual power consumption of the sensor nodes. If the sniffer is enabled, certain sensor nodes are configured as promiscuous nodes and keep a log of all captured frames on a user defined channel. The visualizer and analyzer are universal GUIs allowing both real-time and post-experiment visualization of e.g. packet flows, sensor values or other user measured data, either on a map of the sensor testbed, or by producing a scatter diagram of measured values.

As such, a user is able to easily define tests and emulated scenarios, schedule sensor experiments, and analyze and visualize test results in real-time or after the experiment.

Wireless ad hoc and Wireless Mesh Experiments. As previously stated in Section 3.2, two WiFi NICs are installed at every iNode. In order to enable mixed WiFi node / sensor node experiments and to keep a uniform interface, it was decided to integrate the support for the WiFi nodes into the same web interface as used for the sensor nodes. Moreover, this fully integrated approach assures that no scheduling conflicts can occur between wireless sensor and wireless mesh experiments. Additionally, when running WiFi experiments, the user should be allowed to operate the devices using a custom Linux kernel, custom drivers and custom application software.

Implementing the above flexibility for WiFi tests might endanger the operation of the sensor testbed: in the default testbed set-up, the iNodes execute a daemon which interprets management information from the central control server, controls the EE and installs the firmware to the DUT. Hence, there could be a certain risk involved in allowing the iNodes to be used for experiments: if a WiFi experiment goes wrong or a user deliberately or unwillingly removes or corrupts crucial files needed for booting the iNode or controlling the sensor nodes, the sensor testbed might become unstable or stop functioning.

These potential issues were avoided as follows. Three subcomponents are required to operate the WiFi testbed: the w.iLab-t central control server acting as a *Preboot Execution Environment* (PXE) server, a user defined *Network File System* (NFS) share, and the iNodes themselves. Two partitions are installed on the iNodes: a first partition holding the original iNode software for controlling sensor experiments, and a second partition used for WiFi experiments only, possibly in combination with a user specified kernel. Whenever an experiment is scheduled, the iNodes reboot using the management functionalities of the PoE switch and contact the PXE server to determine which partition to boot. In case of a WiFi or mixed experiment, the iNode is instructed to load the second partition. The user might specify the location of a custom image using specific kernel located on the NFS share, and also specifies the location of the libraries, binaries and other files or scripts needed to perform the experiment. As a new experiment starts, a user defined start script is executed that e.g. might copy the required files from the share to the iNodes, and/or execute a specific program. Not all nodes need to run the same code, allowing experiments with different node roles.

After the WiFi experiment completes, the iNodes automatically reboot and are instructed to load the first partition. As the first partition is booted again, the second partition is restored to its default state, providing clean iNodes for the next test using WiFi nodes.

Each time a scheduled experiment runs, a logging directory is created on the user defined NFS share. For each iNode in the test, a subfolder is automatically created that uniquely identifies the iNode by its hostname. The respective directories are then mounted to a logging directory on the iNodes. All output that is redirected to this directory on the iNodes is stored on the NFS share. This results in a flexible, fully user specified logging system. Furthermore, as the clocks on iNodes are synchronized through the Precision Time Protocol, logging output can be correlated by adding timestamps to the log messages.

4 Additional Features and Lessons Learned

4.1 Defining New Experiments

The W-iLab.t infrastructure allows fast and easy deployment of newly developed code on a large number of devices. Therefore, it is tempting to not only use the testbed for large scale deployment of stable algorithms, but also during the development phase for testing incremental adjustments. This results in the testbed not being available for the tests for which it is actually meant, and causes the sensor nodes and/or flash cards of the WiFi node to undergo a large amount of program/erase cycles during a single day, shortening the lifetime of the flash chips in the testbed. Therefore, early development is still performed on isolated small scale set-ups. Additionally, one zone in the testbed is reserved as a *sandbox* area which is meant for functional testing of new code before switching to another testbed zone. While the use of the 'normal' testbed zones is limited by a user-based quota, the sandbox area is not, thus promoting its use.

As for WiFi experiments, it was learned that when no user specified kernel is used, particular care should be taken in keeping the software on the personal testbeds and large scale testbed synchronized. More specifically, different versions of wireless drivers have shown to cause significant changes to stability and throughput, and result in syntax changes, leading to unexpected results. While obvious, simple driver settings such as disabling antenna diversity when only a single antenna is connected to the wireless NIC are often forgotten but result in considerable stability increases.

Furthermore, it was found that when analyzing a protocol, a researcher often has to create a lot of similar tests, where only a few parameters are changed. For example, in a sensor experiment, one might want to re-run a test on a different transmission power, or change the transmission interval of a certain protocol. Therefore, the option to use parameters in test definitions was added to the testbed: a user might schedule a single test, but with different parameters which are determined at scheduling phase. The system will translate these parameters to individual tests and schedule all of them. This way, a very large amount of test data is collected through a single scheduling action.

4.2 Topology Control

As previously stated, the w-iLab.t testbed is not located in a separate room but deployed in an office environment. This way, the use of noise injection [9] topology control techniques or attenuators was hoped to be avoided. While this assumption proved to be correct for the sensor network experiments, it was found that it is hard to create topologies with a large number of hops using the WiFi nodes, as their transmission power cannot be set to a value below $0dBm$ due to driver restrictions. After determining the receive sensitivity of the WiFi cards through a measurement campaign using a variable attenuator and modeling the RF propagation characteristics of the office environment, it was decided to add fixed attenuators to all WiFi interfaces of the testbed on the second and third floor of the testbed, with attenuation values of $10dB$ and $20dB$ respectively. The result of this attenuation is a variation of perceived node density at the different floors. Note that the effect of the $10dB$ attenuators on sending and receiving interfaces may be canceled by changing the output transmission power from $0dBm$ to $20dBm$, and that variation of the transmission power of the attenuated nodes allows to emulate environments ranging from sparsely connected (only the direct neighbors are within transmission range) to very densely connected (over 60 nodes in transmission range).

4.3 Cautionary Perspective on Testbed Experiments

While new testbed experiments are often characterized by unexpected issues such as protocol failures, node failures or driver errors, it is important to realize that every error happens for a reason. Although this is an obvious observation, authors discussing testbed experiments all too often resort to educated guesses on why a certain error was observed, such as "*we believe that the errors are introduced by the wireless driver*". There are two reasons for these often vague descriptions: first, it takes a huge amount of time to debug all aspects of a testbed deployment, while theoretic calculations and simulations might already be available and are considered to provide adequate proof of an algorithm's or protocol's functionality. Second, the tools to analyze the complex behavior of the testbed might lack.

With respect to the above, some recommendations are the following. *(i)* Test should preferably be run with some nodes acting as a sniffer, since the actual transmitted data is often key to solving problems and better understand the actions (not) taken by the protocol under test. *(ii)* Additionally, even when analyzing upper layer protocols, (basic) knowledge of RF propagation and interference is recommended. *(iii)* Finally, using open source software allows deep analysis of observed behavior.

It should never be forgotten that one of the reasons of using testbeds is to be able to study the behavior of a protocol in a realistic environment. If discovered issues are put aside because the are "*probably* due to X or Y", then the effort of implementing a fully working solution should probably not have been made to begin with.

5 Conclusion

The w-iLab.t testbed supports large scale sensor deployments, WiFi based mesh and ad hoc tests, and mixed sensor/WiFi experiments, and is therefore able to analyze the behavior of future heterogeneous network deployments. Nearly 200 node locations are available, situated across three floors of an office building. Through an easy-to-use web-based interface, researchers are able to control the deployment of the software to be tested based on network zones or may address individual nodes. Moreover, the environment emulator allows to emulate sensor network scenarios, provides advanced logging and control, and allows the modular addition of other type of sensor nodes. Test results can be visualized on a map or in graphs in real-time or after the test. The possibility to generate multiple tests based on the same code has proved to be a time-saving functionality, and attenuating WiFi signals is a feasible technique to create a sparser topology in the testbed.

The listed testbed experiences may inspire researchers to design a brand new testbed, or modify or expand their existing testbeds. In order to get a better understanding of the dynamics involved in a real-life deployment, it is necessary to try and explain all erratic behavior observed while conducting testbed experiments. This will eventually lead to the development of robust wireless deployments that are expected to be part of our lives tomorrow.

Acknowledgment

Stefan Bouckaert thanks the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen) for funding part of this research through a grant.

References

1. Bouckaert, S., Bergs, J., Naudts, D., De Kegel, E., Baekelmans, J., Van den Wijngaert, N., Blondia, C., Moerman, I., Demeester, P.: A mobile crisis management system for emergency services: from concept to field test. In: Proceedings of the WiMeshNets 2006 Workshop, part of the Third Intl. Conference on Quality of Service in Heterogeneous Wired/Wireless Networks, Waterloo, Canada (August 2006)
2. Vandenberghe, W., Lamont, K., Moerman, I., Demeester, P.: Connection management over an ethernet based wireless mesh network. In: WRECOM, Wireless Rural and Emergency Communications Conference, Rome, Italy (February 2007)
3. Bouckaert, S., Naudts, D., Moerman, I., Demeester, P.: Making ad hoc networking a reality: Problems and solutions. Journal of Telecommunications and Information Technology 1(1), 3–11 (2008)
4. iLab.t technology center, `http://ilabt.ibbt.be/`
5. Kaba, J.T., Raichle, D.R.: Testbed on a desktop: strategies and techniques to support multi-hop manet routing protocol development. In: MobiHoc 2001: Proceedings of the 2nd ACM International Symposium on Mobile ad hoc Networking & Computing, pp. 164–172. ACM, New York (2001)
6. Werner-Allen, G., Swieskowski, P., Welsh, M.: Motelab: a wireless sensor network testbed. In: Fourth International Symposium on Information Processing in Sensor Networks, IPSN 2005, pp. 483–488 (April 2005)
7. Handziski, V., Köpke, A., Willig, A., Wolisz, A.: Twist: a scalable and reconfigurable testbed for wireless indoor experiments with sensor networks. In: REALMAN 2006: Proceedings of the 2nd International Workshop on Multi-hop ad hoc Networks: From Theory to Reality, pp. 63–70. ACM, New York (2006)
8. PC Engines. Alix system board, `http://www.pcengines.ch/alix.htm`
9. Kaul, S.K., Gruteser, M., Seskar, I.: Creating wireless multi-hop topologies on space-constrained indoor testbeds through noise injection. In: 2nd International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities, TRIDENTCOM 2006 (2006)