Demo Abstract: Twonet - Large-Scale Wireless Sensor Network Testbed with Dual-Radio Nodes

Qiang Li, Dong Han, Omprakash Gnawali
Department of Computer Science
University of Houston
{qiangli,donny, gnawali}@cs.uh.edu

Philipp Sommer, Branislav Kusy
CSIRO Autonomous Systems
Brisbane, QLD, Australia
{philipp.sommer,brano.kusy}@csiro.au

ABSTRACT

We present Twonet, a large-scale sensor network testbed with dual-radio nodes. Twonet has 100 Opal nodes with low-power 32-bit ARM CPU and 2.4 GHz and 900 MHz radios. These nodes are managed by a network of 20 Raspberry Pi nodes at tier 2 and a PC server at tier 1. These nodes together provide a robust testbed for public access. Twonet represents a major addition to the collection of wireless sensor network testbeds that are publicly available. We hope Twonet's availability will foster sensor network research based on a modern 32-bit sensor node architecture and multi-channel wireless networking.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design

General Terms: Design, Measurement

Keywords: Sensor networks, Dual-Radio, Testbed

1 Introduction

Wireless sensor network testbeds traditionally facilitate testing of network protocols and system services in real-world environments at a scale of up to a few hundred sensor nodes. Researchers may initially evaluate their systems using theoretical analysis or simulations. The next step towards more realistic testing is to use a table-top experiment or a testbed to evaluate how algorithms perform on a hardware node platform. The testbeds are typically deployed in office environments and thus offer realistic radio propagation and wireless interference environment to exercise the various properties of network systems we wish to study. Most practical network protocols that enjoy high reputation in the research community perform at least a part of their evaluation on testbeds.

Due to the cost and logistical challenges, testbeds are often limited to a few tens or hundreds of nodes and a fixed network topology. Despite these limitations, testbeds are a popular tool for realistic, moderate-scale evaluation of sensor network algorithms, especially for researchers that would otherwise have limited access to sensor node hardware. Additionally, they provide the necessary repeatability that allows direct comparison of algorithms against related work.

Over the last decade, the wireless sensor network research community has built several testbeds and made them available to researchers from around the world. Motelab, now out of commission, is perhaps the most widely used publicly available wireless sensor network testbed with 200 TelosB motes [6]. Flocklab [4], Indriya [1], TWIST [3], and Kansei [2] are a few examples, of the many testbeds we have built in the last decade. However, most existing testbeds are limited to older 8- or 16-bit architectures with limited program flash and memory and to a single radio interface. Furthermore, existing testbeds provide only limited access to sensor nodes and a very basic support for collection, inspection, and debugging of experimental data.

We present *Twonet*, a publicly available large-scale sensor network testbed with dual-radio sensor nodes based on the modern Cortex-M3 architecture. Twonet's creation is motivated by the increasing interest in research on multi-channel wireless networking in sensor networks. Twonet, consists of 100 Opal nodes [5] with 2.4 GHz and 900 MHz radios deployed across four floors of an academic building. Similar to the way early sensor network testbeds helped advance research on wireless sensor networks, Twonet will enable new research on multi-band radio communications and will enable new class of sensor network applications that utilize the powerful yet energy-efficient Cortex-M3 CPU architecture.

2 System Architecture

Our design of Twonet is inspired by earlier sensor network testbeds and uses the time-tested three-tier architecture.

Tier 1: Controller. A single Linux server controls the operations and access to the testbed. The controller serves three main functions. It provides a web front-end for users to interact with the testbed. The user may specify experiment parameters, upload binaries, and download results. The controller distributes the programming and control job to the Proxies at tier 2. The controller also collects the logs generated by the sensor nodes and saves it to a database.

Tier 2: Proxy. Twenty Raspberry Pi nodes constitute tier 2 of Twonet. Raspberry Pi is a low-cost embedded Linux platform with sufficient memory and CPU cycles to program, control, and collect the logs from the sensor nodes. Each Raspberry Pi is powered using a Power-over-Ethernet

This work is supported in part by the Sensors and Sensor Networks TCP of CSIRO and a generous gift from Cisco.

SenSys'13, Nov 11-15 2013, Roma, Italy.
ACM 978-1-4503-2027-6/13/11
http://dx.doi.org/10.1145/2517351.2517440.

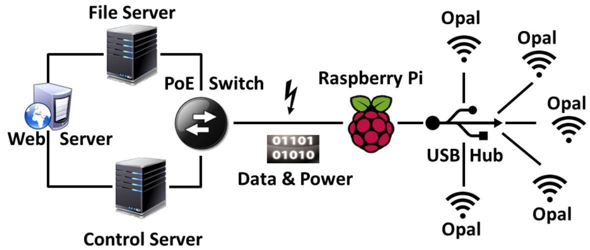

Figure 1: System architecture of the Twonet testbed: Up to five Opal sensor nodes are connected to each proxy (Raspberry Pi) via USB. Multiple proxies are connected to the testbed controller through a power-over-Ethernet backbone.

(PoE) switch and connects to sensor nodes through a USB hub. Raspberry Pi nodes run embedded Linux and use standard Linux toolchain to program the Opal nodes and to collect the debug logs generated by the nodes. Each raspberry Pi node is connected to 5-7 Opal nodes.

Tier 3: Sensor Node. The Opal nodes constitute the leaves of the network. Each Opal node is connected to a custom-built debug board. The debug board and Opal node each connect to Raspberry Pi using separate USB cables. The debug board is required to reset Opal during programming and also provides additional mechanisms to debug programs running on Opal. Each Opal node has a custom plastic enclosure with two antenna mounted for 2.4 GHz and 900 MHz radios.

Fig. 1 shows how these three tiers are organized in Twonet to achieve robustness, efficiency, and flexibility in the testbed design. Deploying a set of Proxy nodes at tier 2 improves the robustness of the testbed as the testbed can restart USB sub-system to restore individual Opal nodes. Being able to restart a single Proxy node is preferable to restarting a tier 1 controller. The Proxy nodes also help speed up testbed programming: all the Proxies can simultaneously program several nodes in parallel. Finally, having multiple Proxy nodes per floor allows for shorter distances between Opals and a Raspberry Pi's. The USB cables connecting Opals to Proxies are kept to within 25 feet, as we observed increased programming errors otherwise.

One of the common challenges in most testbed designs is powering the nodes. Our nodes are deployed above false ceilings and in places where there might not be abundant power outlets. The safety code prohibits power lines above the ceiling. As a workaround, we use PoE switch to power the nodes. Raspberry Pi is not designed to be directly powered using PoE. We use a PoE splitter to tap into power on the Ethernet ports to power Raspberry Pi. Each Raspberry Pi needs 10-14 USB connections (each Opal+debug board combination requires two USB cables) and hence requires an active USB hub with a large number of ports. We install two Ethernet ports at each Proxy location: one for networking and power for Raspberry Pi and another to power the active USB hub using a PoE splitter.

3 Debugging Modalities

In this section, we discuss different modalities for debugging sensor network applications that Twonet provides.

Serial output. Code instrumentation using `printf` commands is a common method to observe changes in the node's internal state. Thereby, the bandwidth of the serial port limits the update rate. The Cortex-M3 microcontroller on the Opal sensor node provides two separate serial ports for debugging: (1) a UART port running at 115200 baud, and (2) a high-speed USB 2.0 port operating at 480 Mbits. Data written to both serial ports are timestamped by the Raspberry Pi and forwarded to the file server for persistent storage.

Memory Tracing. The Opal's Cortex-M3 microcontroller debug port implements the JTAG protocol to provide read/write access to the system memory and peripherals. We employ a FTDI 2232H chip to convert USB commands from the Raspberry Pi into JTAG command sequences for the Cortex-M3. Therefore, we are able to periodically read the content of specific memory locations on the Opal node and report changes to the control server. This allows to collect memory traces without affecting the timings of the application under test and at a higher rate than what would be feasible by logging serial output from the application.

Global Breakpoints. The remote JTAG access to the Opal nodes allows for sending simultaneous start/stop requests to all nodes within the network. Thus, it is possible to stop the application running on the nodes and take a snapshot of the network state for further analysis. For example, the control server can decide to stop the network when memory tracing has revealed an anomaly in the application or network protocol which needs further investigation.

4 User Interface

These features are available on the testbed's web interface:

- Schedule the tasks by specifying the start time and duration of their experiments;
- Inspect and change the real-time status of the nodes: program, stop, or restart the node;
- Download data logged through the serial interface;
- The administrative interface allows approval of users, setting user quota, updating node metadata, and other housekeeping functions.

5 Conclusions

The three-tier organization of nodes in Twonet makes it robust, efficient, and cost-effective for deployment. The debug board allows advanced debugging functionality, typically only available in a small-scale testbed. We hope Twonet will contribute to advancing the state of research in multichannel wireless networking.

References

[1] M. Doddavenkatappa, M. C. Chan, and A. L. Ananda. Indriya: A low-cost, 3d wireless sensor network testbed. In *TRIDENTCOM*, pages 302–316. Springer, 2012.

[2] E. Ertin, A. Arora, R. Ramnath, M. Nesterenko, V. Naik, S. Bapat, V. Kulathumani, M. Sridharan, H. Zhang, and H. Cao. Kansei: a testbed for sensing at scale. In *IPSN '06*.

[3] V. Handziski, A. Köpke, A. Willig, and A. Wolisz. Twist: a scalable and reconfigurable testbed for wireless indoor experiments with sensor networks. In *REALMAN '06*.

[4] R. Lim, F. Ferrari, M. Zimmerling, C. Walser, P. Sommer, and J. Beutel. FlockLab : A Testbed for Distributed , Synchronized Tracing and Profiling of Wireless Embedded Systems. In *IPSN '13*, 2013.

[5] P. Sommer and B. Kusy. Demo Abstract: Distributed Debugging Architecture for Wireless Sensor Networks. In *IPSN '13*.

[6] G. Werner-Allen, P. Swieskowski, and M. Welsh. MoteLab: A Wireless Sensor Network Testbed. In *IPSN '05*.