NREL
NATIONAL RENEWABLE ENERGY LABORATORY

Field Trial of a Low-Cost, Distributed Plug Load Monitoring System

B. Auchter, D. Cautley, and D. Ahl
Energy Center of Wisconsin

L. Earle and X. Jin
National Renewable Energy Laboratory

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Technical Report
NREL/TP-5500-61257
March 2014

Contract No. DE-AC36-08GO28308

NREL
NATIONAL RENEWABLE ENERGY LABORATORY

Field Trial of a Low-Cost, Distributed Plug Load Monitoring System

B. Auchter, D. Cautley, and D. Ahl
Energy Center of Wisconsin

L. Earle and X. Jin
National Renewable Energy Laboratory

Prepared under Task No. BE13.0401

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

National Renewable Energy Laboratory
15013 Denver West Parkway
Golden, CO 80401
303-275-3000 • www.nrel.gov

Technical Report
NREL/TP-5500-61257
March 2014

Contract No. DE-AC36-08GO28308

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Available electronically at http://www.osti.gov/scitech

Available for a processing fee to U.S. Department of Energy and its contractors, in paper, from:

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831-0062
phone: 865.576.8401
fax: 865.576.5728
email: mailto:reports@adonis.osti.gov

Available for sale to the public, in paper, from:

U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161
phone: 800.553.6847
fax: 703.605.6900
email: orders@ntis.fedworld.gov
online ordering: http://www.ntis.gov/help/ordermethods.aspx

Cover Photos: (left to right) photo by Pat Corkery, NREL 16416, photo from SunEdison, NREL 17423, photo by Pat Corkery, NREL 16560, photo by Dennis Schroeder, NREL 17613, photo by Dean Armstrong, NREL 17436, photo by Pat Corkery, NREL 17721.

Printed on paper containing at least 50% wastepaper, including 10% post consumer waste.

Acknowledgments

The authors wish to acknowledge Ashleigh Keene of the Energy Center of Wisconsin for assistance with data analysis, and Smartenit team members Alfredo Choperena (president), Justin Samuel (customer relations manager), and Juan Eduardo Delgado Medina (software developer) for their support during the project.

We also thank Dane Christensen of the National Renewable Energy Laboratory for his help in selecting minimum specification requirements for the proof-of-concept monitoring system.

List of Acronyms

AHEM	Automated Home Energy Management
API	Application programming interface
FTP	File Transfer Protocol
LQI	Link Quality Indicator
MEL	Miscellaneous electric load
RF	Radio frequency
SBC	Single board computer
TCP/IP	Transmission Control Protocol/Internet Protocol
UPS	Uninterruptible power supply
USB	Universal serial bus
ZBRE	ZigBee repeater

Executive Summary

Detailed end-use monitoring at the plug load level remains a technical and logistical challenge. Miscellaneous electric loads (MELs) are the fastest growing energy end use, and represent the biggest category of energy use in buildings after space conditioning (U.S. Energy Information Administration 2013). Researchers have struggled to inventory and characterize the energy use profiles of the ever-growing category of MELs because plug-load monitoring is cost-prohibitive to the researcher and intrusive to the homeowner. These data represent a crucial missing link to our understanding of how homes use energy. Detailed energy use profiles would enable the nascent automated home energy management (AHEM) industry to develop effective control algorithms that target consumer electronics and other plug loads. If utility and other efficiency programs are to incent AHEM devices, they need large-scale datasets that provide statistically meaningful justification of their investments by quantifying the aggregate energy savings achievable.

To address this need, we have investigated a variety of plug-load measuring devices available commercially and tested them in the laboratory to identify the most promising candidates for field applications. We report the lessons learned from field testing of one proof-of-concept system, called Smartenit (formerly SimpleHomeNet), to which we added software to manage system configuration, interval-based data collection, and data transfer. The system was evaluated based on the rate of successful data queries, reliability over a period of days to weeks, and accuracy. It offers good overall performance when deployed with up to ten end nodes in a residential environment, although deployment with more nodes and in a commercial environment is less robust. We conclude that the current system is useful in selected field research projects, with the recommendation that system behavior is observed over time.

Table of Contents

List of Figures

List of Tables

1 Introduction

In the pursuit of energy-efficient residential technologies and building practices, measurement and verification play a crucial role in establishing the cost/benefit performance of any new system. If the efficiency measures implemented address the largest household electricity users and are expected to result in substantial energy savings at the whole-house level, then the collective impact of those measures may be adequately addressed by comparing the monthly utility bills before and after the upgrades. For more detailed evaluations of individual measures, the power consumption of each end use may need to be monitored separately. For major appliances and systems that have their own dedicated circuits, this is not an onerous task, and can be accomplished relatively inexpensively depending on the level of accuracy desired. In addition to the variety of research-grade energy meters available, recent years have seen significant advancements in the home energy monitoring space. Geared toward do-it-yourself consumers, a variety of energy monitors on the market today can be purchased for less than $400.

Detailed end-use monitoring at the plug load level remains a technical and logistical challenge. Miscellaneous electric loads (MELs) are the fastest growing energy end use, and represent the biggest category of energy use in buildings after space conditioning (U.S. Energy Information Administration 2013). Researchers have struggled to inventory and characterize the energy use profiles of MELs because plug load monitoring is cost prohibitive to the researcher and intrusive to the homeowner. MELs are diverse, ubiquitous, and constantly changing, so measuring their energy use is logistically complex and resource-intensive.

MELs data represent a crucial missing link to our understanding of how homes use energy, and recent years have seen growing interest in quantifying plug load energy consumption via submetering. A few early studies focused largely on creating an inventory of typical household MELs and their associated power draw characteristics for on, off, and standby states to identify savings opportunities (Bensch et al. 2010; Roth et al. 2007; Hendron and Eastment 2006). Although these studies provide basic data on energy consumption, they do not provide the kind of real-time, detailed load characteristics data that could enable the nascent automated home energy management (AHEM) industry to develop effective control algorithms to target consumer electronics and other plug loads. If utility and other efficiency programs are to incent AHEM devices, they need large-scale datasets that provide statistically meaningful justification of their investments by quantifying the achievable aggregate energy savings.

To address this need, we have investigated a wide variety of plug load measuring devices available on the market today and tested them in the laboratory to identify the most promising candidates for field applications. The scope of this report centers around the lessons learned from a field validation of one proof-of-concept system, called Smartenit[1] (formerly SimpleHomeNet). Section 2 gives background information about the selection process and an overview of the laboratory testing results that led us to select this device for a field-based evaluation. Section 3 describes our trial field deployments of the Smartenit system. Section 4 discusses the system cost and performance tradeoffs. Section 5 concludes by summarizing our findings to date and outlining our plans for future work.

[1] www.simplehomenet.com/

2 Distributed Plug Load Monitoring Proof-of-Concept

2.1 System Requirements and Initial System Selection

We first established the desired high-level, qualitative attributes of a system suitable for residential field tests:

- System to monitor 120 VAC electrical plug loads via a wireless interface
- Low cost
- Commercially available and certified for residential and commercial uses (Underwriters Laboratories/Consumer Electronics/Canadian Standards Association safety certification, Federal Communications Commission approval)
- Ability to record detailed power attributes at a rate of multiple measurements/minute, including
    - Integrated energy consumption
    - Instantaneous true power
    - Voltage
    - Current
    - Power factor
- Reliable (able to deliver data with a low failure rate over weeks to months)
- Robust (ability to withstand physical, electrical, and communications stresses)
- Accurate
- Unobtrusive to occupant (small form factor, neutral appearance)
- Cross-manufacturer compatibility (not vendor locked)
- No monthly data subscription charges or cloud-hosted storage fees.

Based on these requirements, we initially identified 27 unique plug-in solutions[2] designed to measure electrical loads and transmit data through a wireless network. Most of the devices we found that satisfied our requirements appeared to be mainly sold within the AHEM market. Some are cross-listed as "demand-response controllers" and are marketed toward utilities. These devices were designed for load control, informal load measurement, and sometimes simple automation. In contrast, the desired research application centers around rapid, repeatable, and accurate load measurements. These AHEM-geared products are available at lower cost than similar equipment in the industrial automation market, with a probable tradeoff in reduced reliability and robustness. The AHEM systems are typically designed to measure and transmit readings no oftener than about once per minute.

[2] See Appendix A.

The consumer AHEM sector is changing very rapidly, with many products being adapted to new requirements as the industry evolves and utilities explore the uses of these types of technologies. As a result, the products under investigation changed significantly during the time that the proof-of-concept was being developed, tested, and evaluated. These changes made development of a proof-of-concept system more challenging, but ultimately resulted in improved functionality. We found that the manufacturer whose products we worked with most extensively was receptive and willing to work with us to meet some of our specific application needs within their existing product lines at no additional cost.

2.2 Smartenit Proof-of-Concept System Description and Initial Testing

The Smartenit (previously called Simple Home Net) system selected for development and testing includes plug load monitors ("end nodes") and a central gateway (the "Harmony Platinum gateway," which can be replaced by alternative products). ZigBee wireless communication occurs between the gateway and plug load devices. The central gateway acts as a connection between the ZigBee wireless network and an outward-facing Internet connection, allowing remote data access and control.

The Smartenit system was selected as a promising candidate for proof-of-concept development, based on the following characteristics:

- The ability to perform custom application development on the Smartenit gateway (which allows researchers and developers to define application-specific queries, query intervals, data storage options, etc.).
- A provided Web-based user interface for easy setup, joining, and tagging of the ZigBee wireless mesh network.
- The ability to use other original equipment manufacturer gateways with this vendor's plug load monitors (not vendor-locked).
- Hardware capable of an accuracy of about ± 2% error on energy measurement (kWh); the Analog Devices energy monitoring integrated circuit allows ± 0.5% error on energy measurement.
- The energy measurement integrated circuit used also provides measurement of instantaneous power, voltage, current, and power factor.
- Per the ZigBee communication protocol, the wireless network behaves as a self-healing mesh network.
- The product offering is actively under development and improvement, allowing for integration with emerging wireless standards (e.g., ZigBee SE 2.0), development of iOS/Android applications, and development of third-party interfaces.
- The design group, company president, and technical support staff were receptive to our needs and have incorporated critical product change requests in a timely manner.
- The manufacturer claimed to have tested the system with up to 30 end nodes.
- The unit cost of a plug load monitor (end node) is about $80 in small quantities.

- The plug load monitoring devices have been offered since at least 2009 and have a significant history of use in residential settings.

We initially procured a gateway and three end nodes (which grew to nearly 50 during the course of our work) for testing and development. We used the system outlined in Figure 1 to develop a software application to query each plug load monitor in the system to obtain the latest energy consumption values from registers in the end node. We were able to run this application on the Smartenit gateway, a network-attached computer, and a network-attached Android phone. In each case the data file is stored on the device that is running the application. The Smartenit gateway has a secure digital (SD) card slot for up to 8 GB of additional data storage. A local database can also be created on the gateway for data storage and retrieval. In the present version using the Transmission Control Protocol/Internet Protocol (TCP/IP) application programming interface (API), the application queries each plug load monitor sequentially, moving on to the next plug load monitor as soon as it receives a response. A timeout routine terminates an individual end node query if the gateway doesn't receive a response within an established interval; we initially set this threshold to 3 s. A separate software application was then developed to send the data file to a central data server for live visualization, system diagnostics, data storage redundancy, and off-site analysis.

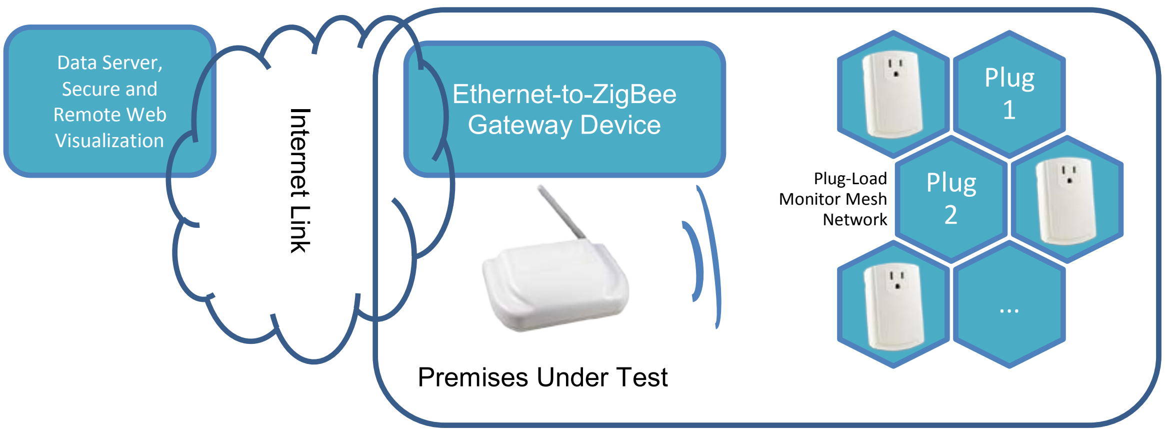


Figure 1. Smartenit proof-of-concept plug load monitoring system

2.3 Initial Testing of Smartenit System

In the fall of 2012 we operated the Smartenit system over a period of several weeks in a small office environment during development, monitoring typical household-variety plug loads including a computer desk power strip, refrigerator, coffee maker, and printer. After working through some system bugs with the product's software developers, a system including 15 end nodes operated without interruption for about 5 weeks before crashing. We discovered that the TCP/IP API we used can introduce problematic behavior, and once the software application was migrated to an operating system (OS) Debian bus (D-Bus) API the system operation improved dramatically. We then deployed the Smartenit system with 15 end nodes in a single-family residence. With end nodes located on three separate floors, we found that one or two end nodes would provide intermittent connectivity, so proper on-site diagnostics would need to be performed for each field deployment to ensure reliable connectivity of all devices in each new environment.

2.4 Smartenit System Updates

Following our initial testing, Smartenit released, at our request, new firmware for the plug load monitors that provide, in addition to electrical energy and instantaneous power values, line voltage, current, and power factor values for each plug.

Subsequently, Smartenit released a version of its gateway software that can be run on an open hardware, open source software single board computer (SBC) called the Raspberry Pi and reduced the cost of a gateway to approximately $80. The Raspberry Pi, rev. B, has more computing power in its ARM-based processor and more memory than the Harmony Platinum Gateway that sells for approximately $350. This software release further reduces the likelihood that the Smartenit product line will become vendor-locked.

The latest system configuration used in our testing includes the Raspberry Pi SBC with a universal serial bus (USB) dongle that comprises a ZigBee coordinator and radio transceiver. We revised the software for querying plug loads, which now offers several configurable parameters including:

- The dwell time between plug load monitor queries
- The wait time before a timeout or retry occurs on a plug load monitor
- The number of retries per plug load monitoring device
- The scan rate in which the entire group of plugs should be queried (5 seconds, 1 minute, etc.)
- The maximum file size of the resultant data file before a new comma-separated value (CSV) based time series file is created (to break time-series data into user-specifiable quantities for post analysis)

Repeater nodes can be added to the network to boost signal strength in weak areas of the mesh network. As firmware and software continued to be developed for this platform, it became important to capture the specific hardware, software, and user configuration for each test run. A separate file was created at the beginning of each run to capture the firmware version and unique address of each plug load monitor, along with the firmware of the USB coordinator/transceiver, the distribution version of the ZBPServer[3] application running on the SBC, and user-specified settings for that specific test run. Each full query of a plug load's six data fields (voltage, current, power, frequency, power factor, and accumulated energy) is accomplished in two sequential subqueries, or groups, of three data fields. We also included more detailed time stamping in later versions of our operating software. The start of each query cycle is recorded as a real-time value, and the response of each node is recorded as an elapsed time from the start of the query cycle. This allows evaluation of response times as one indicator of network quality.

When plugged into a home/office uninterruptible power supply (UPS), this system can be run either as a standalone logging platform or as a network connected system that delivers data back

[3] http://simplehomenet.com/solutions.asp?page_id=HomAidPi

to a secure server. With the UPS in place, the system can recover from power outages during field tests. If properly configured, a field-deployed system can be remotely managed.

Since the switch to the D-Bus architecture, most development and testing have focused on the Raspberry Pi-based gateway. This is the current system and it has exhibited better stability, modularity, and configurability compared to the OEM-offered Harmony Platinum gateway.

A diagram of components for a full Smartenit field test system utilizing the new Raspberry Pi-based gateway is shown in Figure 2. An ecosystem overview is shown in Figure 3. Figure 4 shows a block diagram of the internal elements of the ZigBee firmware. A list of hardware components for this system is given in Table 1.

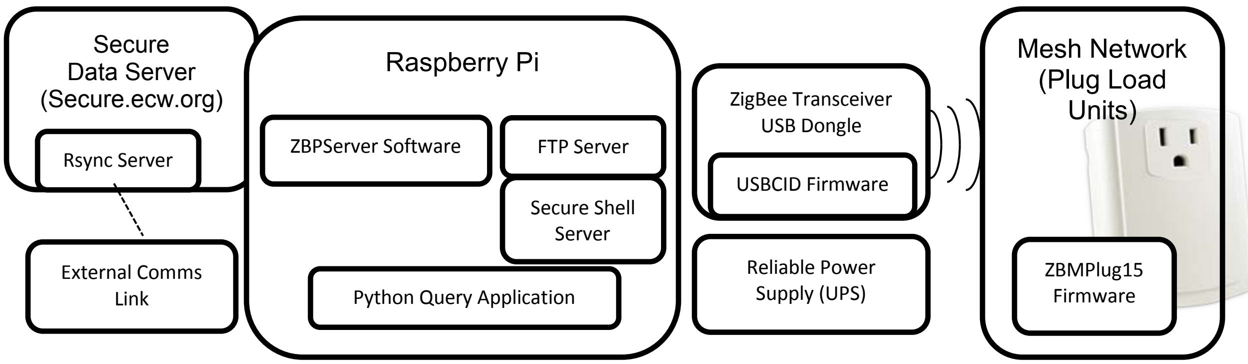


Figure 2. ZigBee system block diagram

Figure 3. SmartenIt ecosystem diagram (with Raspberry Pi)

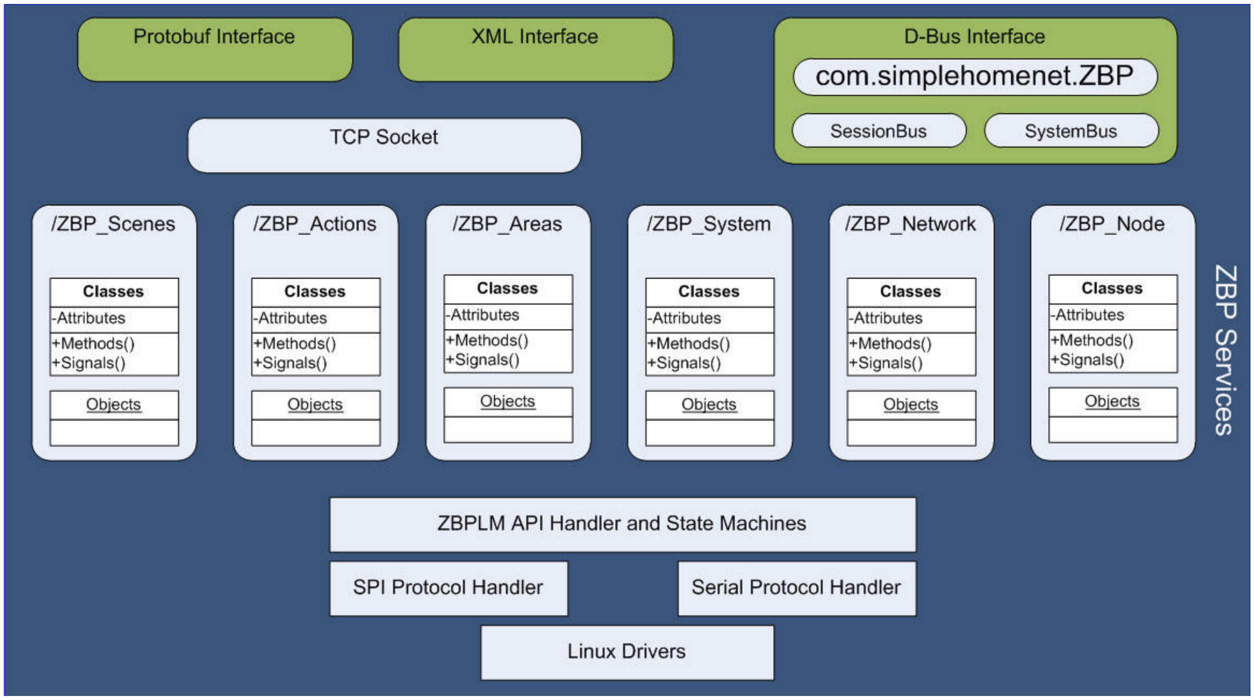


Figure 4. SmartenIt ZBPServer software functional block diagram (Installed on Raspberry Pi)

Table 1. Hardware Components for Smartenit Field Test System

SBC (Raspberry Pi) with a USB dongle ZigBee transceiver (including ~16 GB of local storage)	required
A UPS and AC-to-DC supply for reliable DC power to the SBC (optional: communications equipment)	required
Plug load monitoring wireless ZigBee mesh network routers (ZBMPlug15s)	required
Temporary TCP/IP connection (Ethernet wiring to a router) for setup/takedown	required
External network TCP/IP communication for the duration of testing (cellular modem, wired/wireless Ethernet connectivity, etc.)	optional
ZigBee repeaters (ZBREs) to strengthen radio signal links	optional

Smartenit product features and offerings continue to change, and cross-platform partnerships are emerging (e.g., Blue Line Innovations partnership press release).[4]

2.5 Laboratory-Based Performance of Smartenit System

2.5.1 Accuracy Testing

We performed accuracy testing of 30 Smartenit end node devices with data transmission via a network-attached computer. Load measurement accuracy is not affected by gateway or computer use. Seven devices were used as sample loads; these are listed in Table 2.

[4] www.prweb.com/releases/Smartenit/PCMinterface/prweb9050958.htm

Table 2. Sample Loads for Smartenit Testing

1	15 W Incandescent Light Bulb
2	25 W Incandescent Light Bulb
3	40 W Incandescent Light Bulb
4	100 W Incandescent Light Bulb
5	~450 W Halogen Lamp
6	Box Fan With Low, Medium, and High Settings
7	~870 W Shop Vacuum

Each load was connected through each end node in turn, and operated for about 1 min, resulting in 5–10 measured data points for each load on each end node (i.e., over the course of about 1 minute each load was measured 5–10 times). The reference measurement system was a Continental Control Systems WattNode[5] energy meter using AccuCT model current transformers. The WattNode was polled by a Campbell Scientific[6] CR1000 data logger using Modbus communication. The reference measurements are expected to be within ± 2% of real values across the range of power and power factor observed.

The Smartenit end nodes show a ± 6% range of error compared to the WattNode readings. Larger errors were seen in less than 1% of readings (see Figure 5). The mean error across all end nodes tested is close to zero. Figure 6 shows errors in Watts, including testing across all loads. Additional analysis of the data could establish whether individual end nodes display systematically different errors, and whether greater error values can be correlated to particular loads. Influences may include low power factor of load, localized wireless and wireline electromagnetic interference from the loads, nonlinear measurement attributes of internal components at specific current draw levels, localized line voltage dips due to loading, temperature effects, and time delay of measurement between reference system and plug load measurement on dynamic loads.

[5] http://www.ccontrolsys.com/w/WattNode_Pulse
[6] http://www.campbellsci.com/cr1000-datalogger

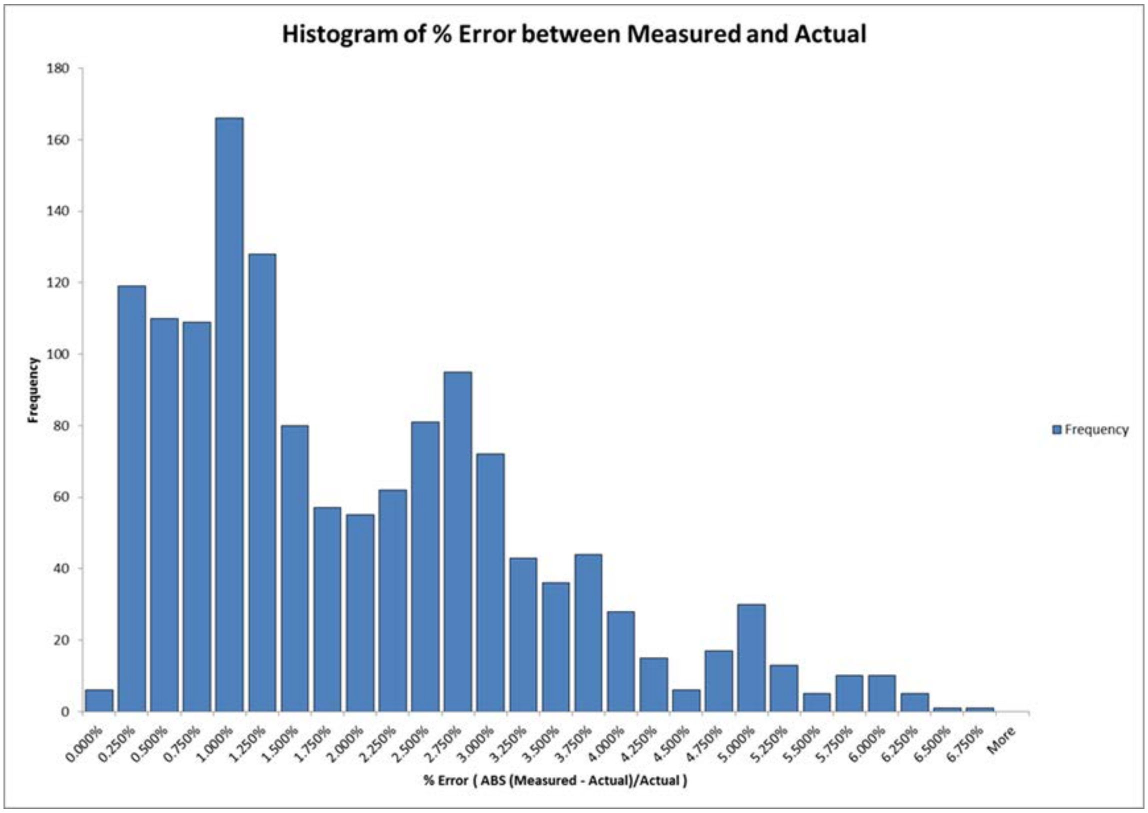


Figure 5. Histogram of energy measurement error

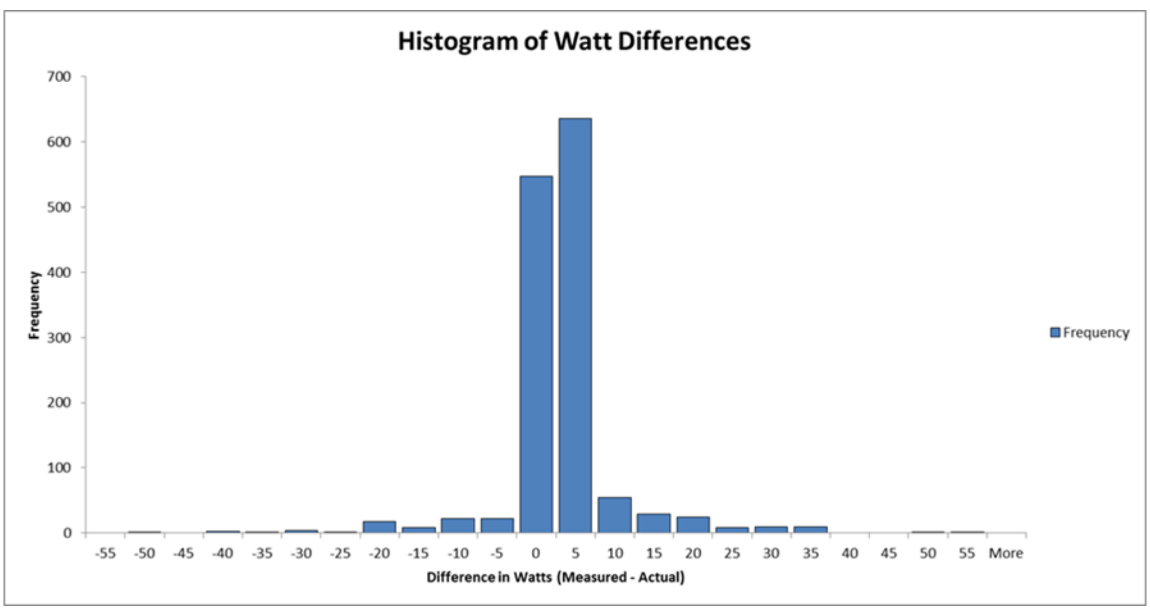


Figure 6. Difference in Watts, measured versus reference test system

2.5.2 Time Delay Testing

The purpose of time delay testing was to identify the time delay between when an electric plug load physically transitions and when this transition is time-stamped by the proof-of-concept system. Variance in this time constant can be caused by a number of factors, including:

- Size of wireless mesh network (e.g., number of end nodes communicating over network and relaying signals).
- Layout/physical distribution of end nodes (e.g., nodes in one straight line, nodes in a cluster, hourglass shaped distribution, can introduce network delay effects particular to the network shape).
- Topology of network (e.g., mesh, tree, star).
- Method/type of query from gateway (e.g., step through each individual end node, timeout or query retries before moving to the next end node, disbanding garbled or misaddressed responses, number of data registers queried per end node).
- Rate of query (e.g., as fast as possible, once every 15 minutes).
- Radio frequency (RF) interference/increased noise floor in the radio spectrum of the mesh network (microwave oven operating, cell phone nearby, WiFi streaming video, etc.)
- Localized RF attenuation/interference/reflectance for each end node (electromagnetic reflectance behind metal refrigerator requires more transmission retries, behind potted plant with leaves/soil containing high moisture attenuates signal, in concrete parking garage, etc.)
- Gateway resource consumption (e.g., API being used by two processes, CPU usage, memory read/write delays).
- Method of switching power at the outlet (using the end node's internally switched control relay to switch loads can also increase the network traffic through the system under test).

We performed time delay testing of a 15 end-node Smartenit system with data transmission via the Harmony Platinum Gateway (this is not the current Raspberry Pi system being used). The reference system was a Campbell Scientific CR1000 data logger with current switches connected to analog inputs. A total of 10 plug load monitors in the system had 40-Watt incandescent work lights as the switched load. The other five plug load monitors remained in the communication network, but they did not have switched loads attached to them. A diagram of the test setup is shown in Figure 7, where Plug1–Plug10 signify the plug load monitors with loads, the lamps signify the individually switched nodes, and the doughnut shapes signify the reference system's current switches.

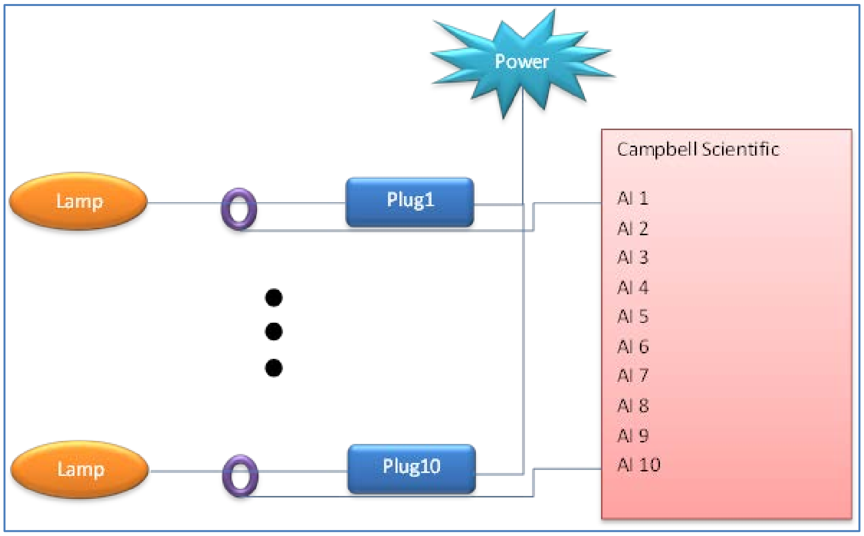


Figure 7. Test setup for time delay testing

The CR1000 recorded the current switch values (on or off) at a rate of approximately 2 ms between readings. The CR1000 clock and the Harmony Platinum Gateway clock were both synchronized to a local National Institute of Standards and Technology network time protocol time server to ensure accurate timestamp values that can be correlated during analysis. Loads were switched at a rate of about once every 15 seconds. The delay between state change measured at the current switch and when the state was measured by the proof-of-concept system was then analyzed for this test system. The typical scan cycle duration for this experiment was 9 seconds.

The time delay results were then analyzed using two comparison points:

1. The time difference between the reference system's 2-ms scan timestamp that contained the load's state change and the timestamp at the beginning of the Harmony Platinum's system scan that contained the state change (Figure 8). Note that, when a node fails to respond to a query, the observed time of the state change via the Smartenit system may be delayed by the time of one or more full scan cycles.
2. The time difference between the reference system's 2-ms scan timestamp that contained the load's state change and the time stamp of the individual plug load's state change response was recorded by the gateway (Figure 9).

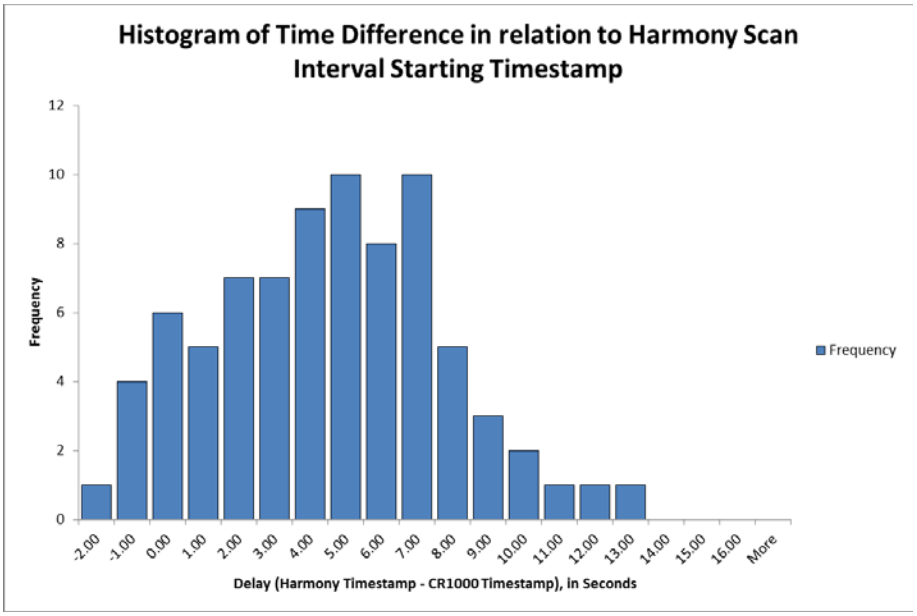


Figure 8. Time delay related to system scan cycle timestamp

The system scan cycle time difference settles at approximately 6 seconds for this 15-end node system.

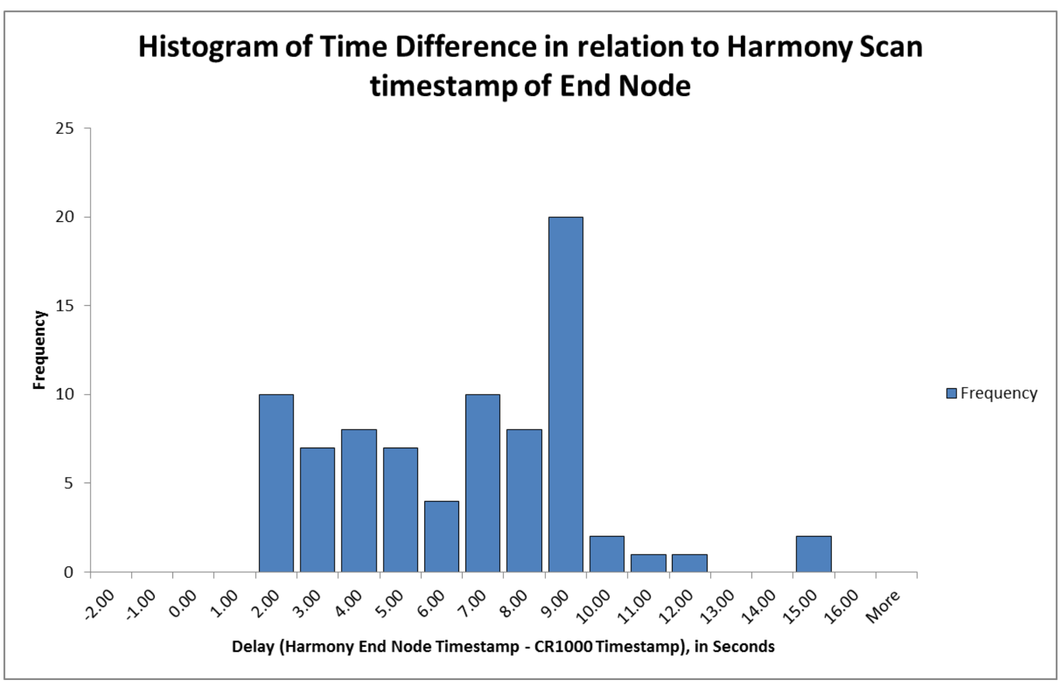


Figure 9. Time difference related to end node timestamp within the system scan cycle

The end node scan timestamp difference appears to settle at around 9 seconds for this particular 15-end node system. A theoretical time difference extends to infinity if a particular end node drops off of the network or is temporarily out of contact with the rest of the system. If a single end node drops out, this can affect system scan cycle time proportional to the timeout value set (currently set to a value of 3 seconds per end node). This timeout value and/or the scan methodology can be changed for each installation. Proper on-site setup diagnostics and system characterization should be performed for every field deployment to ensure reliable connectivity and to minimize timestamp delays for all devices in the specific field deployed environment.

Overall, the Smartenit end nodes show a range of time delay of 2–15 seconds for a test system of 15 plug load monitors. Additional testing should be done to quantify the timing delay for field deployed systems to compare with timing delays seen in these laboratory systems.

Based on our laboratory tests we concluded that the Smartenit system was a promising candidate for a low-cost field-based plug load monitoring system. We verified that the out-of-the-box accuracy of Smartenit end nodes is adequate for general research on plug load energy consumption, and, if used in conjunction with high-accuracy whole-house monitoring, more than adequate for load disaggregation. Studies of the energy use of specific appliances may call for calibration testing of individual end nodes. In addition, we should characterize system accuracy for new electrical attributes, which were not available prior to the company's recent firmware upgrade.

3 Field Testing of Smartenit System

Using the Smartenit field testing system outlined in Section 2.4, several field tests were performed to evaluate the system across a variety of use cases. The performances of these deployments were then validated against both the original success criteria of the system outlined in Section 2.1 and any projected requirements associated with the specific use case.

The setup procedure involved the following general steps. We developed an informal setup guide for use within the team working on the project, and development of a more complete user guide is a candidate for future work on the system.

1. **Determine a location for the Raspberry Pi SBC and ZigBee radio transceiver (gateway).** Setup also requires at least a temporary TCP/IP connection to the Raspberry Pi. The gateway should be centrally located such that good RF link quality can be maintained with several nodes. Therefore, it's worth taking the time to map node locations ahead of setting up the Raspberry Pi and transceiver. We used a USB extension from the Raspberry Pi to place (tape) the transceiver dongle approximately 6 feet off the ground. The Raspberry Pi SBC was then powered on, which started the ZBPserver software.
2. **Assess the site's 2.4 GHz spectrum use and noise.** We used a low-cost 2.4-GHz frequency spectrum analyzer to determine the best channel to run the system on. This is useful because other home devices may be using the 2.4-GHz spectrum such as wireless networks and devices (including microwave ovens that create noise across this band), which may interfere with the Smartenit system. The ZigBee channel with the lowest apparent noise was chosen for the system.
3. **Set up the Smartenit gateway.** We set up each test by logging on to the harmonygateway.com Web interface, locating the Raspberry Pi device, creating a new test area, adding the gateway coordinator, and setting the ZigBee channel. If older tests were using the same gateway, the previous node network was reset and cleared.
4. **Deploy plug monitoring units.** We established plug nodes by sequentially adding them to the newly created network or area. This was accomplished by setting the Smartenit gateway in discover mode using the Web interface, resetting a plug monitor and plugging it into the test plug, and then adding it to the area once the coordinator discovered it. Our general approach to setting up nodes was to sequentially add nodes radially from the transceiver. Once a few nodes were added to the network, we used Smartenit's Link Quality Indicator (LQI) (a graphical display of the active network, displaying RF signal strength to and from each node) to check for weak links in the network.. If the link quality was poor, we added repeaters to boost signal strength.
5. **Launch the query software.** Once all plug monitors were deployed and network link quality was deemed adequate, the query software on the Raspberry Pi was started. The latest version of the software allowed a number of key parameters to be changed. Once the software was started, a data file was immediately created containing network and node information. We monitored the first few minutes of the data file to ensure data were being collected and the system was running properly.

6. **Monitor the data and the network.** Data were stored locally on the Raspberry Pi as the system was running. If a TCP/IP connection were available, the data could be transmitted to a separate server. We monitored system performance continually using the harmony gateway Web interface and by manually checking data quality.

3.1 Description of Field Test Deployment

The system was tested in four building environments, as outlined in Table 3.

Table 3. Summary of Field Deployed Test Environments[7]

Site	Location	Minimum Number of Nodes Tested	Maximum Number of Nodes Tested	Description
MPLS01	Minneapolis, Minnesota	2	8	Lab bench top testing station in a commercial office space
MPLS02	Minneapolis, Minnesota	2	12	Single-family detached wood-frame home, 1.5-story bungalow with basement
MAD01	Madison, Wisconsin	9	10	Single-family detached wood-frame home, 2-story with basement
MAD02	Madison, Wisconsin	15	21	Commercial office space, steel-stud walls and steel-framed cubical partitions

The primary purpose of the field deployments was to examine the reliability of the Smartenit system using the Raspberry Pi gateway to successfully query and receive data from each node in various test conditions. The primary metric of interest was the failure rate defined by:

$$Failure\ rate = \frac{number\ of\ instances\ of\ failed\ queries}{total\ number\ of\ queries}$$

A failed query occurred when the Raspberry Pi software, upon querying a node, was unable to receive a response from the node. This resulted in a "not a number" (NaN) being recorded in the data file.[8] Failure rate was aggregated several ways, including by node, by hour, by site, and by test case. A secondary objective was to observe the stability of the system in each test environment over time.

All the tests for which quantitative results are reported here were done using one of two recent Smartenit ZigBee server firmware versions, the latest of which became available in October 2013.

[7] The designations MPLS01, MPLS02, MAD01, and MAD02 refer to test environments (buildings). These designations with an added digit; e.g., MAD01-2, refer to a specific test performed in that environment.
[8] Each query consists of two separate transmissions between the gateway and node (each of which retrieves one group of three measurement parameters), and failures may occur in either or both groups. In our analysis, we counted a failure of one or both groups for a given node as a failed query.

3.1.1 MPLS01—Baseline Stability Testing

The MPLS01 test system was used as a baseline test system with which to gauge stability in installations with short communication paths and, by implication, reliable signal transmission between nodes. It was installed in an office space on a laboratory bench near an area shielded by a metal filing cabinet to help reduce RF interference from other 2.4-GHz office noise sources such as WiFi (though commercial office RF noise was inherently present in this space). Distances between plug load monitors and the Raspberry Pi gateway were kept within a couple of feet to minimize RF signal attenuation. There were no physical obstructions between any system devices other than plug load wiring. In a few cases, the plug load monitors were plugged into one another, allowing for a compact installation that could run off on the side with little interruption. Electric loads included incandescent lights and some light electronics (cellular modem, occasional laptop charger). The majority of the time the system sat isolated in the space, performing data acquisition of about eight plug load monitors at a scan interval of 15 seconds. Hourly field data were delivered to a remote secure File Transfer Protocol (FTP) site via secure shell over wireline Ethernet.

A number of tests were run with this system at several stages of software development. Results of a test performed with the software version current as of this report are included below as Test MPLS01-1.

Little time was spent on deploying this system or troubleshooting connectivity. Once set up, this system ran without interruption, except occasionally when it was briefly taken down for firmware or software upgrades. The system survived at least two building-wide power outages without a need to restart testing.

The MPLS01 system was also reconfigured late in our testing process to validate the use of a small number of plug load monitors (we used two in our test) at a fast sampling interval, on the order of 1 second. The location and environment for this system were as described above, except that the two plug load monitors were spaced further apart within the 4 ft × 4 ft × 8 ft laboratory space. Electric loads included two incandescent lamps.

LQI data collected during setup show excellent signal strength, with values that exceeded 170 (Figure 10).

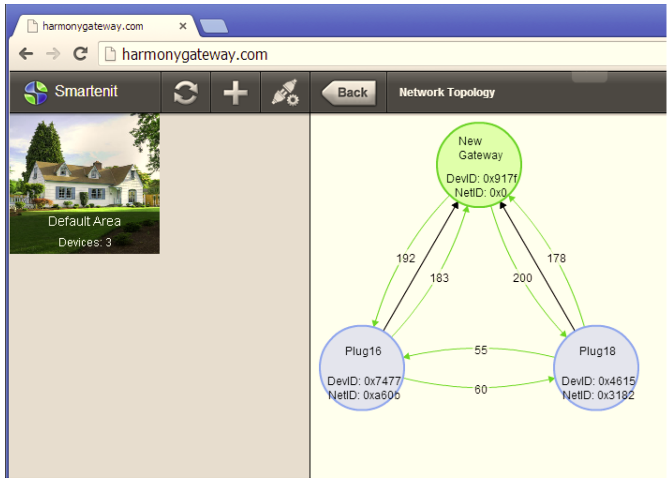


Figure 10. ZigBee network LQI graphic for test MPLS01

3.1.2 MPLS02—Minneapolis Residential Site

One of the more likely use cases for this data acquisition system is field deployment into residential spaces. MPLS02 was one of two single-family detached home sites used as test environments. The site was a 1946 wood-frame bungalow with a basement, located in Minneapolis. The system included a cellular modem data link for remote management and data delivery. The Raspberry Pi gateway, UPS, and cellular link were installed in a corner office on the first floor, and plug load monitors were distributed in electrical outlets in the basement, first floor and knee wall outlets on the second floor. During the initial deployment of the system, no real regard was taken for network link quality between mesh network nodes (the LQI metric), so long as the network seemed to have formed. Later into the field deployment period, a ZBRE was placed mid-height on a basement wall to evaluate whether it could provide a change in system performance. LQI data captured after a ZBRE was added are shown in Figure 11.

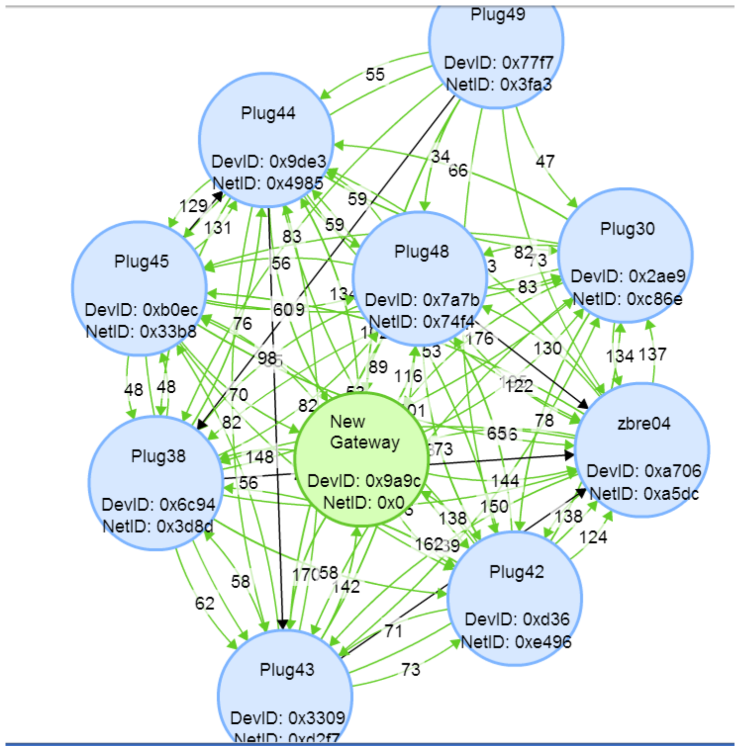


Figure 11. ZigBee network LQI graphic for test MPLS02-2

The LQI numbers from this network topology scan show several weak links between nodes (values < 80), and this can make the low LQI plug load monitors less responsive as the surrounding RF conditions change over the course of a test.

We used a low-cost 2.4-GHz frequency spectrum analyzer to capture the home's ambient RF noise level near the ZigBee coordinator at one point during setup (Figure 12).

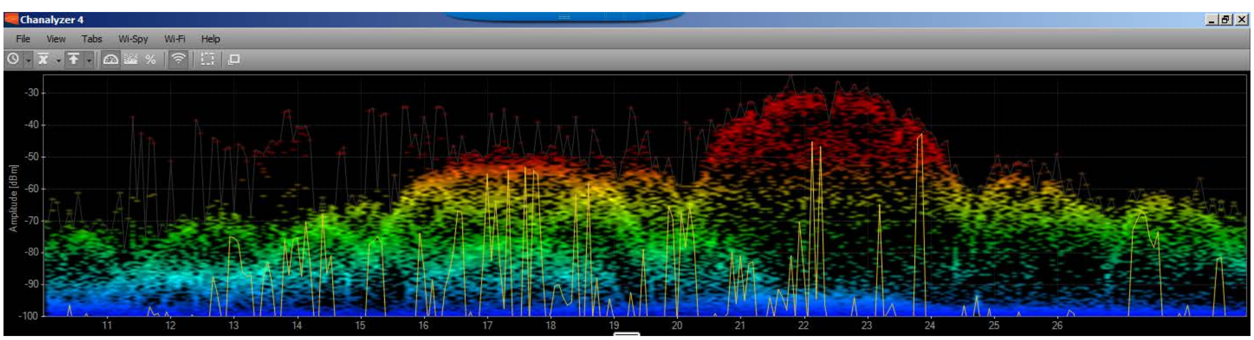


Figure 12. 2.4-GHz spectrum scan graphic for site MPLS02

ZigBee channel 25 was chosen as the operating frequency band for this installation—though channel 11 may have been a better choice based on the above frequency sweep.

This site was used for testing at several points during software development. The results of three tests performed with the software version current as of this report (MPLS02-1, MPLS02-2, and

MPLS02-3) are included below in Sections 3.2.10 through 3.2.12. Plug load monitor locations for one typical test are listed in Table 4.

Table 4. Plug load Monitor Locations for Test MPLS02-2

Plug #	Location
42	Office Lamp
38	Bathroom
43	Basement Fluorescent
zbre04	Basement
30	Living Room – LED Lamp
44	Basement – Halogen
45	Basement – Dehumidifier
48	Basement – Dryer
49	Bedroom – Charger

The system recovered collection after at least two power outages; one lasted more than 2 hours. The cellular link proved effective in performing remote upgrades and system diagnostics and in delivering hourly field data back to a secure FTP server.

3.1.3 MAD01—Madison Residential Site

MAD01 was the second single-family detached home site used for testing. The site was a 1938 wood-frame two-story home with a basement in Madison. The system was linked to the home Internet connection via wireline Ethernet for remote data delivery and local system monitoring and diagnostics. The Raspberry Pi gateway and UPS were installed in a second-floor room, and plug load monitors were distributed in electrical outlets in the basement, first, and second floors. Figure 13 shows link quality data captured during system setup.

The results of four tests run at this site (MAD01-1, MAD01-2, MAD01-3, and MAD01-4) are included in Sections 3.2.2 through 3.2.5.

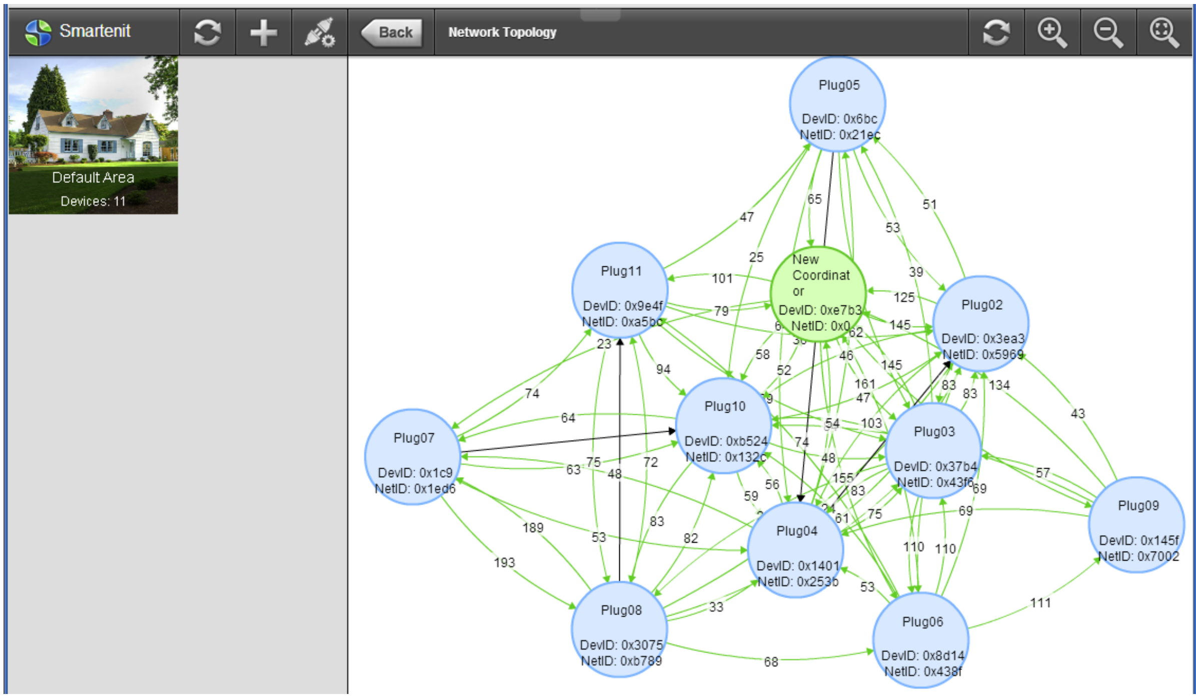


Figure 13. ZigBee network LQI graphic for test MAD01-2

The scan interval at this site was set at 15 seconds, with up to 10 outlets being monitored at one point. Because one plug load monitor became completely unresponsive, the quantity for this site was dropped to 9. Monitoring locations typical of this system are shown in Table 5.

Table 5. Plug Load Monitor Locations for Test MAD01-3

Node	Location
03	Kitchen – Coffee Makers
04	2nd floor MBR – TV
05	2nd floor Office – LED Lamp
06	Kitchen – Microwave
07	Sunroom – TV
08	Sunroom – Stereo
09	Sunroom – Halogen Floor Lamp
10	Basement – Dehumidifier
11	Basement – Washing Machine

A power outage caused the system to fail at one point, and the UPS failed to keep the gateway powered. After this power outage, one plug load monitor became permanently unresponsive.

3.1.4 MAD02—Commercial Office Space, High Quantity Node Testing

The MAD02 setup tested a use case of commercial office space and a high count of plug load monitors. The deployment was in a second-story office suite in a multi-tenant office complex in Madison. The UPS and ZigBee coordinator were located in an office on the outer perimeter, and plug load monitors were distributed throughout the suite, in cubicles and offices. The setup environment had several RF obstructions including metal-studded walls, steel and concrete load-bearing elements, and steel cubicle framing. The environment likely also had a higher RF noise floor compared to any of the other test systems, because numerous electronic communications and computers used the 2.4-GHz unlicensed spectrum. The system had the highest number of plug load monitors of any of the test systems. The network spanned approximately 50 feet, and had the highest density of human occupancy of the spaces used for testing.

The initial setup process resulted in a limited number of joined devices and a network topology shown in Figure 14.

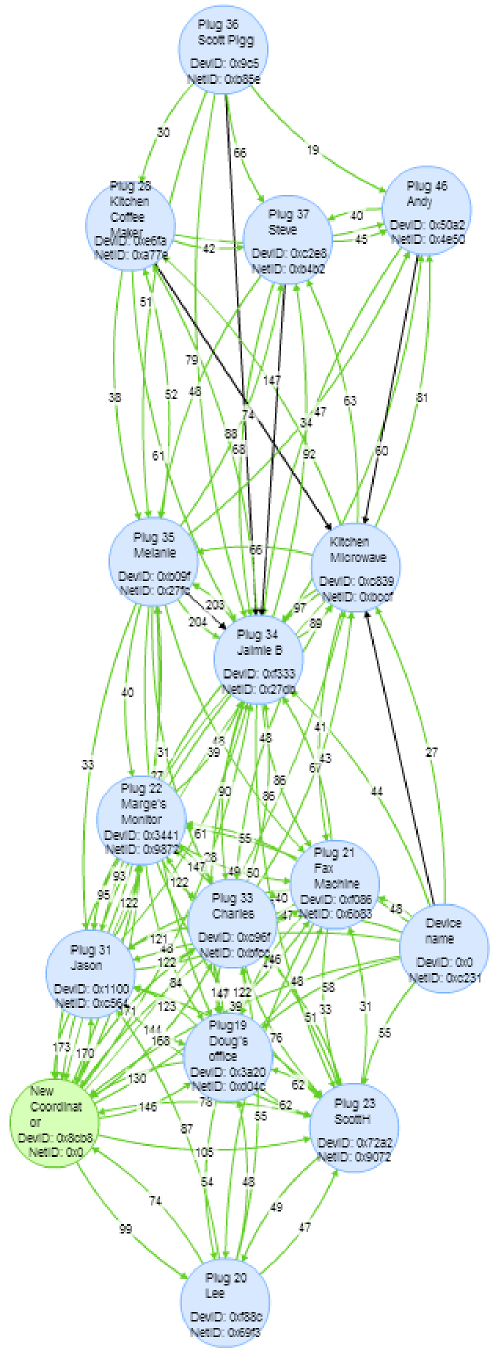


Figure 14. ZigBee network LQI graphic for initial setup at MAD02

A firmware upgrade was required to increase the upper limit in number of devices, and the system deployment was attempted again. The LQI numbers were still too weak to establish a wide mesh network around the entire office. Eventually, the system was deployed in a smaller area (one wing or roughly half the original area) in the office suite. LQI data captured during system setup in the reduced area are shown in Figure 15.

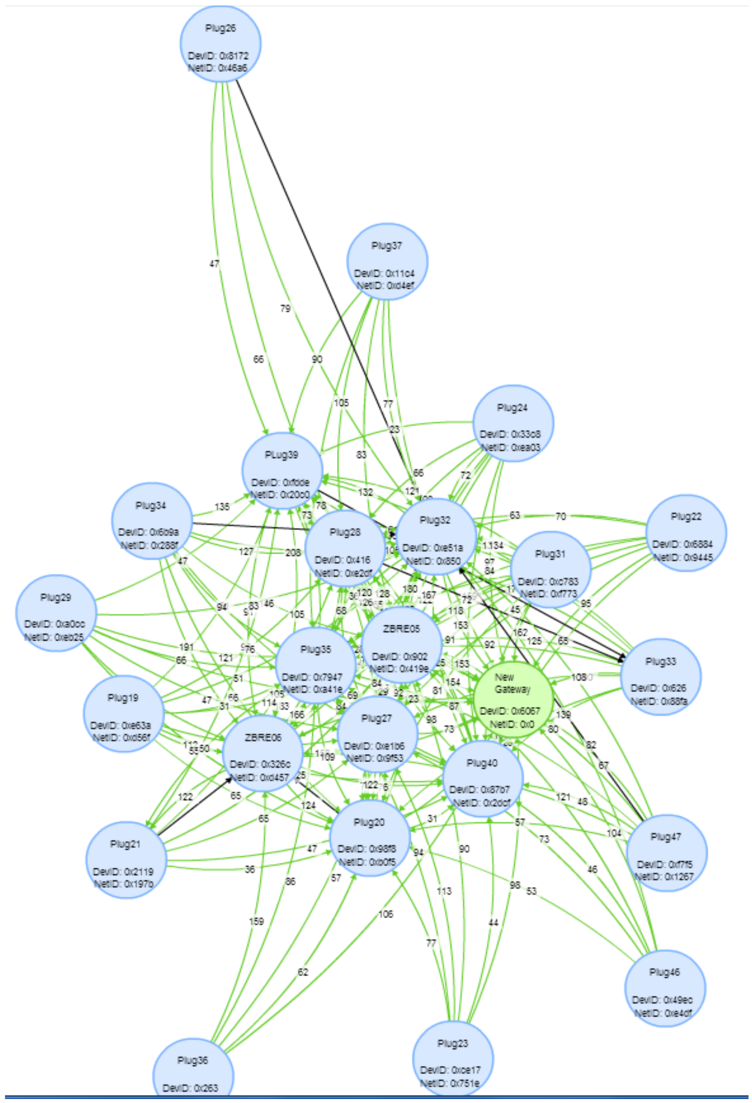


Figure 15. ZigBee network LQI graphic for test MAD02-1, reduced area

Two ZBREs were added to the system, with a total of 21 plug load monitors. Results of two tests (MAD02-1 and MAD02-2) are reported in Section 3.2. The scan interval was varied between 30 seconds and 180 seconds during the field tests.

3.2 Field Test Results: System Operation

3.2.1 Overall Results

The failure rates from 10 test cases are shown in Figure 16, where box ends represent 25th and 75th percentiles and whiskers represent 5th and 95th percentiles of failure rates by hour of test. The overall mean failure rate was approximately 10%. All residential test cases combined had an average fail rate of 3%; the commercial office test site had a combined average fail rate of 32%; these numbers are summarized in Table 6. Four test cases had an average failure rate of less than 1%. The worst test cases occurred in the commercial office space with 21 nodes. Note that the test of a system with just two nodes operating at a higher frequency, which has performed at a very low failure rate, is not included in these results.

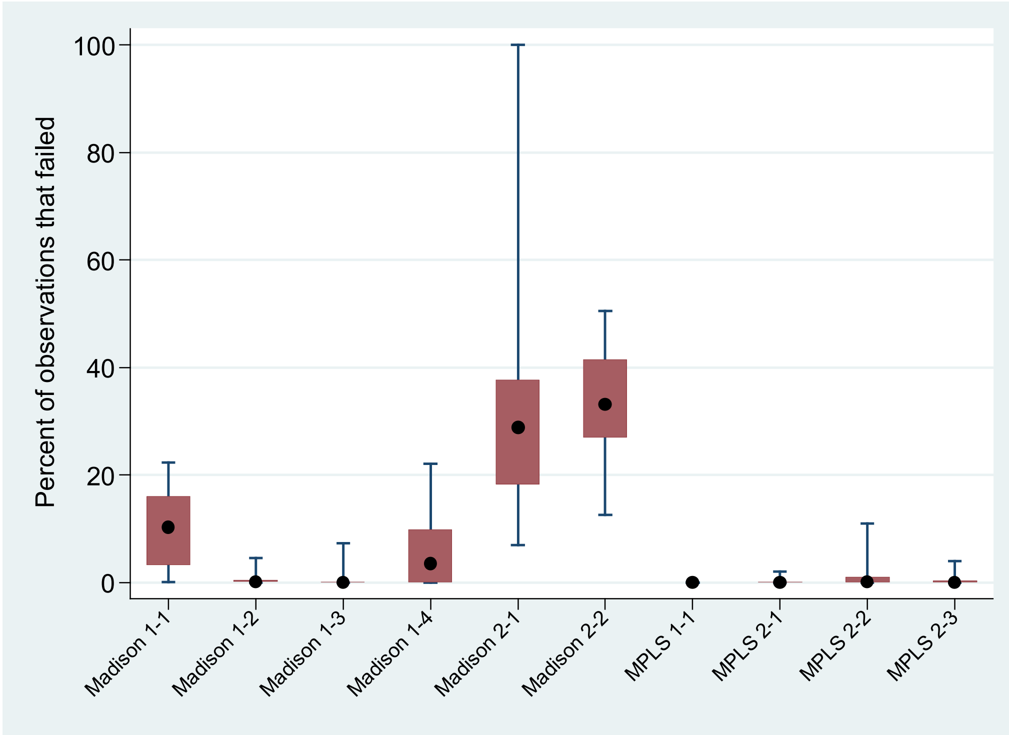


Figure 16. Percent failure characteristics for each test

Table 6. Performance Characteristics of Field Deployment Tests

Test	Number of Nodes	Number of Range Extenders	Test Duration (hours)	Failure Rate (%)
MAD01-1	10	0	88	10.54
MAD01-2	10	0	36	0.72
MAD01-3	9	0	329	0.88
MAD01-4	9	0	53	6.19
MAD02-1[9]	21	2	74	31.32
MAD02-2[10]	21	2	43	33.31
MPLS01-1	8	0	985	0.00
MPLS01-2	2	0	24	< 0.10
MPLS02-1	5	1	68	0.24
MPLS02-2	8	1	146	1.75
MPLS02-3	8	1	162	0.71

[9] Commercial office site, values exclude period of total system failure
[10] Commercial office site, values exclude period of total system failure

3.2.2 MAD01-1 Results

This first test was conducted in the Madison single-family detached home for a total of 88 hours with 10 nodes and no range extenders (repeaters). The overall fail rate was 10.5% with each data group failing nearly simultaneously (i.e., both groups of queried parameters—not one or the other—usually failed). The results are summarized in Figure 17.[11] Two nodes had consistent fail rates greater than 20%; four nodes had a near 0.0% fail rate.

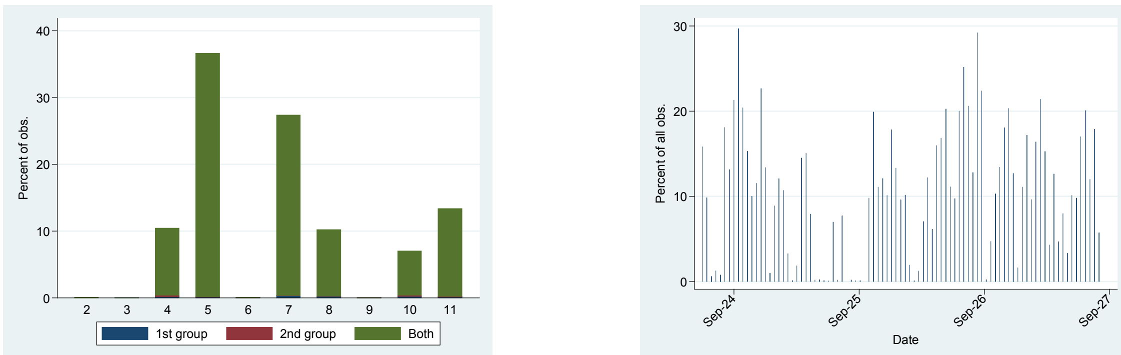


Figure 17. MAD01-1 percent failure characteristics for (L) each node, and (R) all nodes by hour

First group and second group refer to the structure of each query into two subqueries, each of which retrieves three of the six energy measurement parameters available at an end node. Either or both subqueries may fail, and these failure types are counted separately. The x-axis numbers are the end node ID numbers ("Plug2", "Plug3", … , "Plug11").[12]

3.2.3 MAD01-2 Results

The second test in the MAD01 home showed an overall failure rate of 0.72% after running for 36 hours. Most nodes failed simultaneously. Similar to the first test, node 5 had the highest error rate, although nodes showed significant improvement from the first test. The improved performance compared to Test MAD01-1 may be due to the selection of a different ZigBee channel (based on spectrum scan results), and to the reduced effects of human traffic on relocation of the Raspberry Pi gateway from the dining room to a less frequently used upstairs bedroom. This setup did not include a UPS, and the test was terminated unexpectedly by a brief power failure.

[11] Unique node ID numbers from 1 through 49 were assigned to the end nodes for use throughout the project. The nodes used in each specific test were not selected to have sequential ID numbers.
[12] The explanation in the caption for this figure applies to subsequent figures that illustrate similar results (Figures 18–23 and 25–27.)

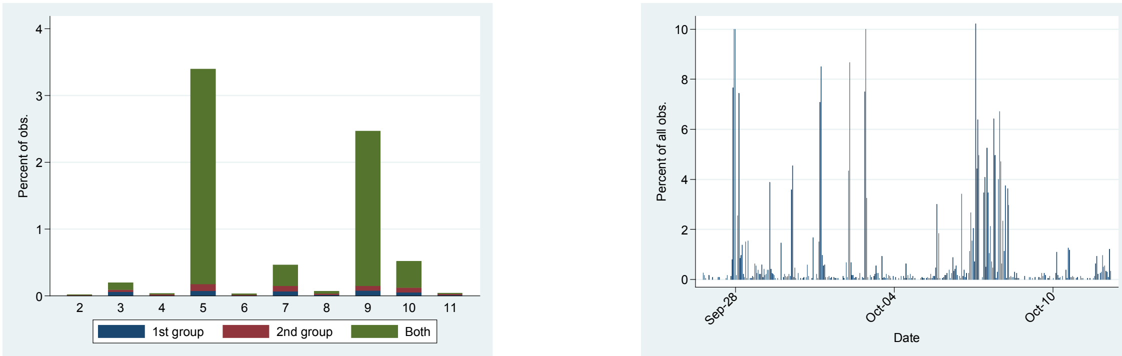


Figure 18. MAD01-2 percent failure characteristics for (L) each node, and (R) all nodes by hour

3.2.4 MAD01-3 Results

For the third test in the MAD01 home, the timeout value used in the operating code was reduced from 1.75 to 0.5 seconds. The test resulted in a mean fail rate of 0.88% and ran the longest at 329 hours with 9 nodes. One node had an error rate of 3.6 %; all others were less than 1.6%.

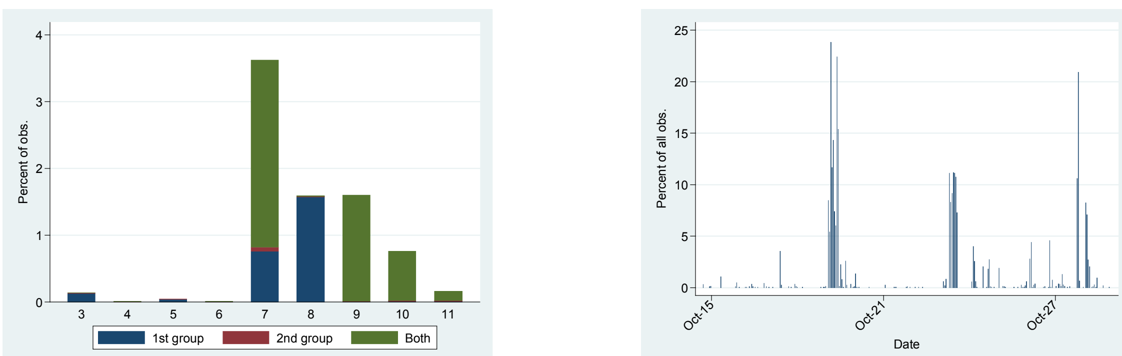


Figure 19. MAD01-3 percent failure characteristics for (L) each node, and (R) all nodes by hour

3.2.5 MAD01-4 Results

The fourth test in the MAD01 home was the first to use the October 2013 Smartenit firmware upgrade. The test ran with nine nodes for 53 hours, and resulted in a mean failure rate of 6.2%. About half the failed queries were for the first query data group only. Most of the error was attributable to two nodes that had mean failure rates of 26% and 28%, respectively. Without more comparative testing, it is impossible to say whether the firmware upgrade is the cause of increased failures compared to the previous test.

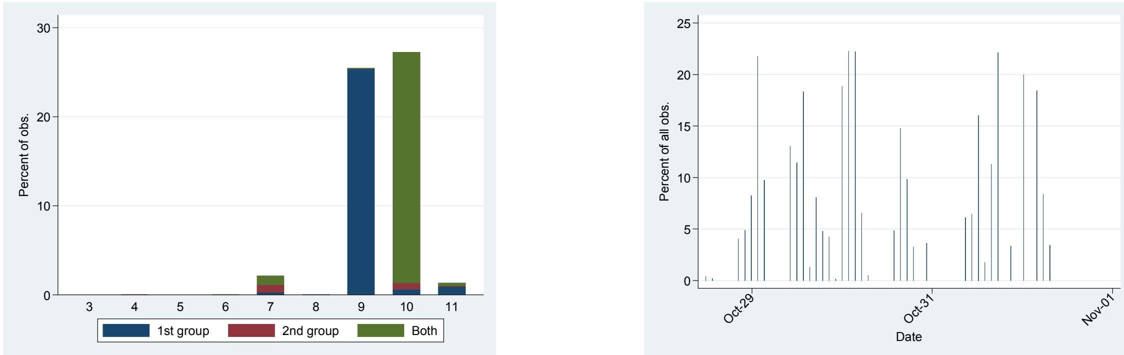


Figure 20. MAD01-4 percent failure characteristics for (L) each node, and (R) all nodes by hour

3.2.6 MAD02-1 Results

The first test in the Madison commercial office space (MAD02) resulted in an overall failure rate of 31% with 21 nodes and two range extenders. This test ran for 74 hours before the entire system failed; more than 95% of queries returned NaN. No clear pattern emerged of certain nodes performing better than others, although there seemed to be a diurnal pattern corresponding to better results obtained during evening hours. Once the system failed it could not recover without reinitializing the entire system. Based on analysis of this particular failure, the root cause appeared to be an issue with the Zigbee transceiver—where the current transceiver will crash if it is overwhelmed with traffic (and its crash log files confirmed this type of failure). The manufacturer acknowledged this issue with large node networks, and further integration testing will be necessary to confirm the bug is fixed in future transceiver updates.

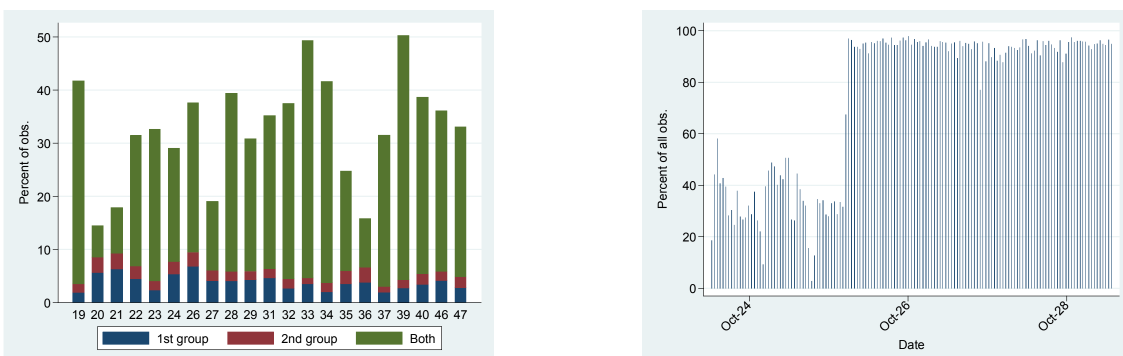
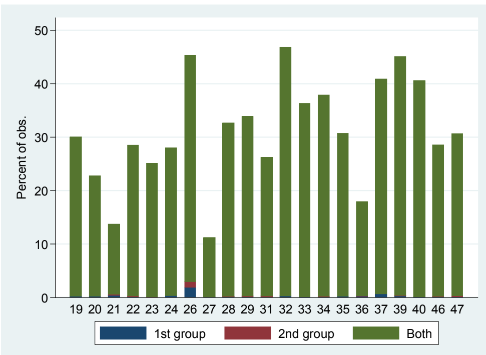


Figure 21. MAD02-1 percent failure characteristics for (L) each node but excluding period of system collapse, and (R) all nodes by hour of test

3.2.7 MAD02-2 Results

ZigBee networking protocols allow for repeated attempts to complete a communication transaction if no acknowledgment is received. These repeated transmissions are controlled by the ZigBee server firmware. We suspected that the collapse of our test system into a mode of almost complete failure may have been due to the overloading of the network with these repeated attempts to communicate. As one test of this hypothesis, for Test MAD02-2 we increased the system scan interval (i.e., reduced the scan rate) from 30 to 180 seconds and increased the software timeout from 1.5 to 3 seconds, both of which should have allowed more time to clear ZigBee transactions. This test was run with 21 nodes and two range extenders, and with nodes somewhat closer on average to the Raspberry Pi gateway. The test resulted in an overall failure rate of 33%. This test ran for 43 hours before experiencing a complete system failure with 100% failed queries. Similar to the first test, there was apparently a diurnal pattern of failure rate with better results occurring during evening hours. As noted in MAD02-1, an anticipated firmware update may resolve this particular failure mode.



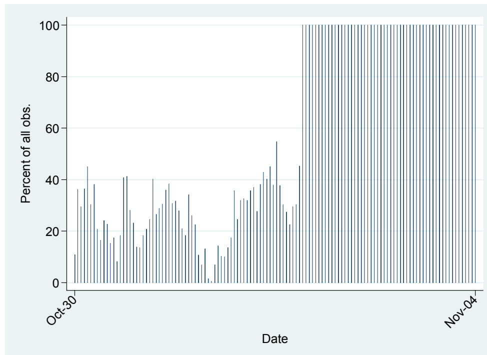
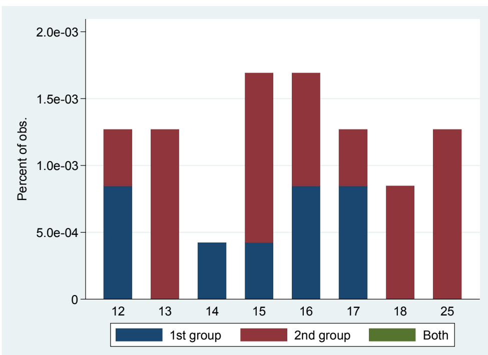


Figure 22. MAD02-2 percent failure characteristics for (L) each node but excluding period of system collapse, and (R) all nodes by hour of test

3.2.8 MPLS01-1 Results

This test was performed on a desktop in a commercial office space. This test resulted in a mean failure rate of 0.001% and ran for 985 hours with eight nodes. The eight nodes were all within 5 feet of one another, including the gateway, and all with a clear line of sight. These test results show that the system can operate reliably when communication paths are short.



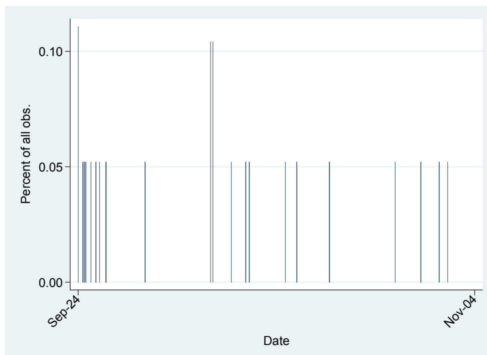


Figure 23. MPLS01-1 percent failure characteristics for (L) each node, and (R) all nodes by hour

3.2.9 MPLS01-2 Results

We conducted a test with two nodes on a bench to observe the maximum sustained speed of successfully querying a node. For both nodes tested, almost 99% of the successful query responses were received within 0.5 seconds, indicating a minimum delay caused by mesh networking or repeated transmission attempts (Figure 24). We ran this test for approximately 24 hours with no range extenders.

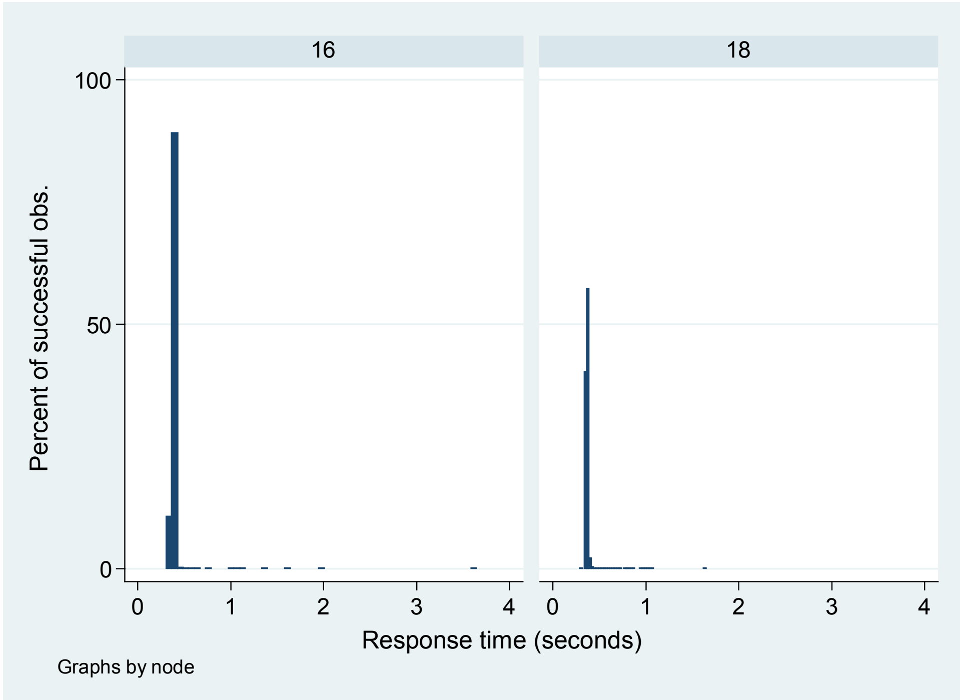


Figure 24. Query response time distribution for MPLS01-2

3.2.10 MPLS02-1 Results

Three separate tests were conducted in a Minneapolis single-family detached home (MPLS02-1, -2, and -3) that showed a combined average failure rate of 0.9% with up to eight nodes and one range extender. The first test, MPLS02-1, showed an average failure rate of 0.24% with five nodes and ran for 68 hours.

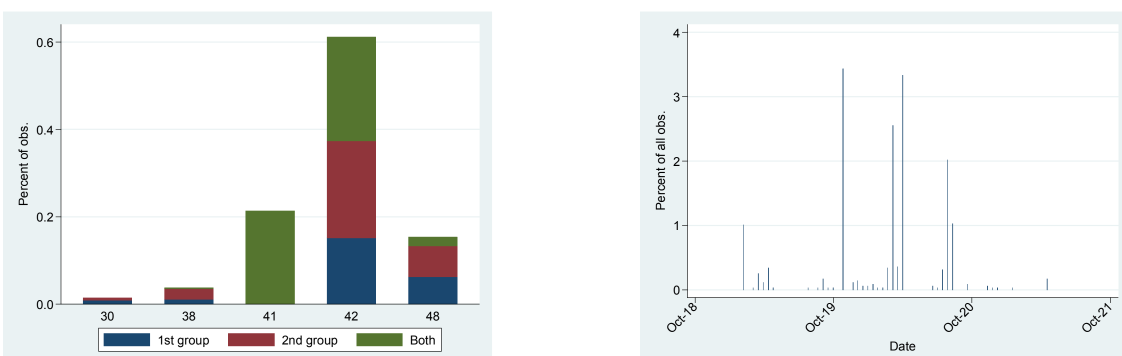


Figure 25. MPLS02-1 percent failure characteristics for (L) each node, and (R) all nodes by hour

3.2.11 MPLS02-2 Results

The second test in the Minneapolis home (MPLS02) resulted in a mean overall failure rate of 1.8% and ran for 146 hours with eight nodes and one range extender. The change in the number of nodes along with the addition of the range extender compared to the previous test makes it difficult to draw specific conclusions from the results.

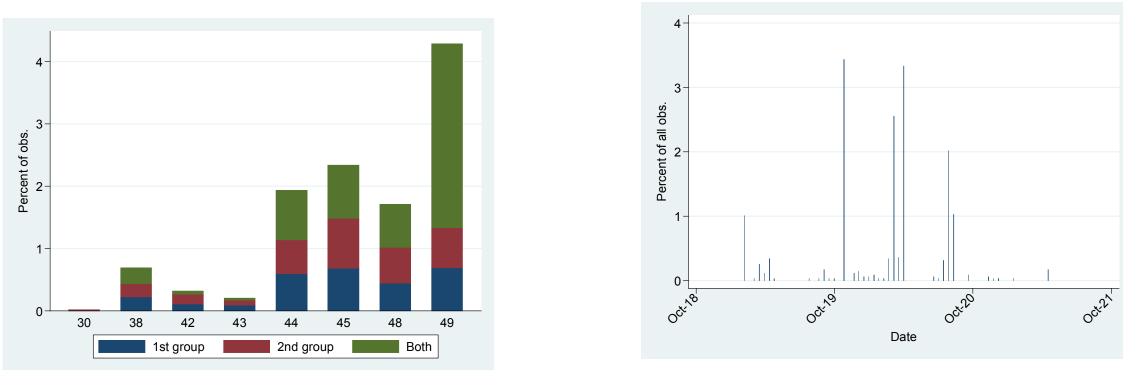


Figure 26. MPLS02-2 percent failure characteristics for (L) each node, and (R) all nodes by hour

3.2.12 MPLS02-3 Results

The third test in the MPLS02 home resulted in an average overall failure rate of 0.7% and ran for 162 hours with eight nodes and one range extender.

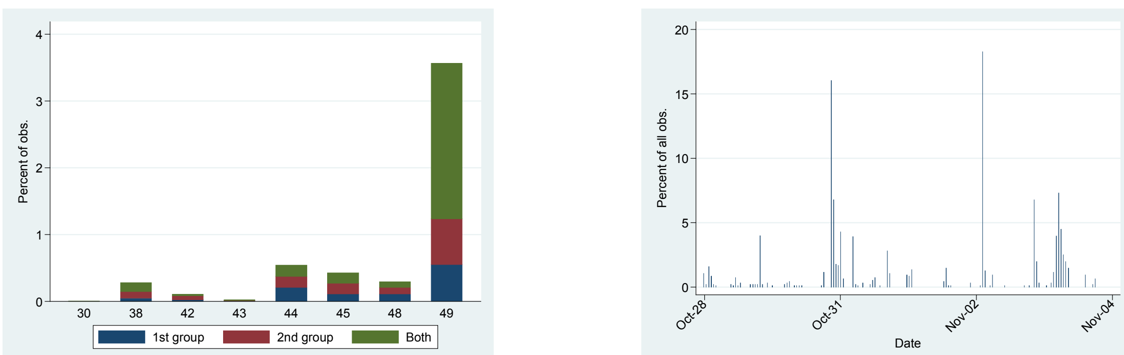


Figure 27. MPLS02-3 percent failure characteristics for (L) each node, and (R) all nodes by hour

3.3 Field Test Results: Microwave Interference

The residence used as the MAD01 (residential) test site is equipped with a whole-house electric monitoring system using WattNode electric power metering devices installed at the main distribution panel to meter each circuit independently. We used data from this system to identify microwave oven operation, and evaluated the relationship between microwave operation and plug load monitoring query failures during each 1-minute period. An average power of 20 Watts or more during any 5-second measurement interval as observed on the WattNode system was used as an indication of microwave operation during that minute. This analysis includes combined data from tests MAD01-2, and MAD01-3 and MAD01-4.

The results in Table 7 show solid evidence that microwave oven operation in this home is associated with an increase in plug load query failures. With only one exception, positive differences of greater than 3% in query failure rate are significant at a probability level of about 0.01 or better, while very small and negative differences generally show poor statistical significance. The overall mean query failure rate across the three tests considered here was 4.4%.

Table 7. Impact of Microwave Oven Operation

Query Failure Rate			Statistical Significance of Difference (p-value)	Test Number	Node	n
Microwave OFF	Microwave ON	Difference (% pts)				
28%	53%	24.90	0.003	4	10	2914
26%	47%	20.59	0.011	4	9	2914
4%	8%	3.57	0.002	3	7	19778
4%	7%	3.36	0.006	2	5	14920
0%	3%	3.33	0.000	4	8	2914
0%	3%	3.33	0.000	4	6	2914
0%	2%	1.43	0.002	2	3	14920
0%	0%	0.36	0.002	2	6	14920
3%	7%	3.30	0.320	4	11	2914
3%	5%	1.51	0.189	2	9	14920
1%	1%	0.63	0.213	3	10	19778
1%	1%	0.09	0.855	2	10	14920
0%	0%	0.07	0.790	3	3	19778
1%	1%	0.06	0.924	2	7	14920
0%	0%	0.01	0.965	3	11	19778
0%	0%	0.00	.	4	3	2914
0%	0%	0.00	.	4	5	2914
0%	0%	–0.02	0.792	3	4	19778
0%	0%	–0.03	0.768	3	6	19778
0%	0%	–0.03	0.919	4	4	2914
0%	0%	–0.05	0.706	2	4	14920
0%	0%	–0.06	0.689	2	2	14920
0%	0%	–0.07	0.674	2	11	14920
0%	0%	–0.10	0.565	3	5	19778
0%	0%	–0.17	0.505	2	8	14920
2%	1%	–0.85	0.233	3	8	19778
2%	1%	–0.87	0.227	3	9	19778
6%	0%	–5.62	0.182	4	7	2914
1.5%	2.0%	0.50	0.002	ALL	ALL	353,428

The microwave oven was connected to node 6, which experienced fewer failures and a slighter increase in failure than some other nodes. This indicates that proximity to the microwave oven was not the dominant factor in interference; rather, directional effects and interruption of signals using a transmission path through the microwave node may have been factors. Although multiple other factors must clearly be at play, microwave oven operation stands out as one significant factor in data loss.

3.4 Field Test Results: Accuracy

As mentioned in the preceding section, the residence used as the MAD01 test site is also equipped with a circuit-level electric monitoring system. We used data from this system to check the accuracy of the plug load monitoring system. In general, the power measured at an individual plug load can be assumed equal to the circuit-level power only when the plug device is the sole load on the circuit. In this case, there were four such plug loads: coffee maker, microwave oven, washing machine, and dehumidifier.

We used energy consumption data, rather than instantaneous power, for this comparison, as energy consumption values reduce the effect of any short-term power fluctuations, and are expected to provide a more robust comparison. Although the circuit-level WattNode system collects data at 5-second intervals, and the plug load system every 15 seconds, we recognized that small differences in the exact timing of data collection would mean comparisons of measured values within any specific time span would be subject to errors. (Timing of data collection for the plug load system in particular is inexact, because the time for completing ZigBee communications with each node varies.) To minimize this time offset problem, we summarized data over 1-hour periods.

We also found the two systems behave somewhat differently when loads less than a few Watts were measured. The WattNode system records data at a higher numeric resolution, and often shows a positive power consumption value over a period for which the Smartenit system records no usage. However, the WattNode system also sometimes displays erroneous small power consumption values when actual consumption is very low or zero. These errors are likely due to the placement of current transformers closely together in an electrically noisy distribution panel. We don't so far have a systematic method for determining which low-level readings are accurate and which are noise-induced. To eliminate low-level errors from our analysis, we excluded data in which the average power consumption is below 0.2 Watts (as measured by the plug load system).

Figure 28 shows the distribution of the differences between the Smartenit plug load measurement system and the WattNode circuit level monitoring system, presented as fractional differences.[13]

[13] Positive values indicate a higher WattNode reading, negative values indicate a higher plug load reading. The WattNode system values are used as the denominator.

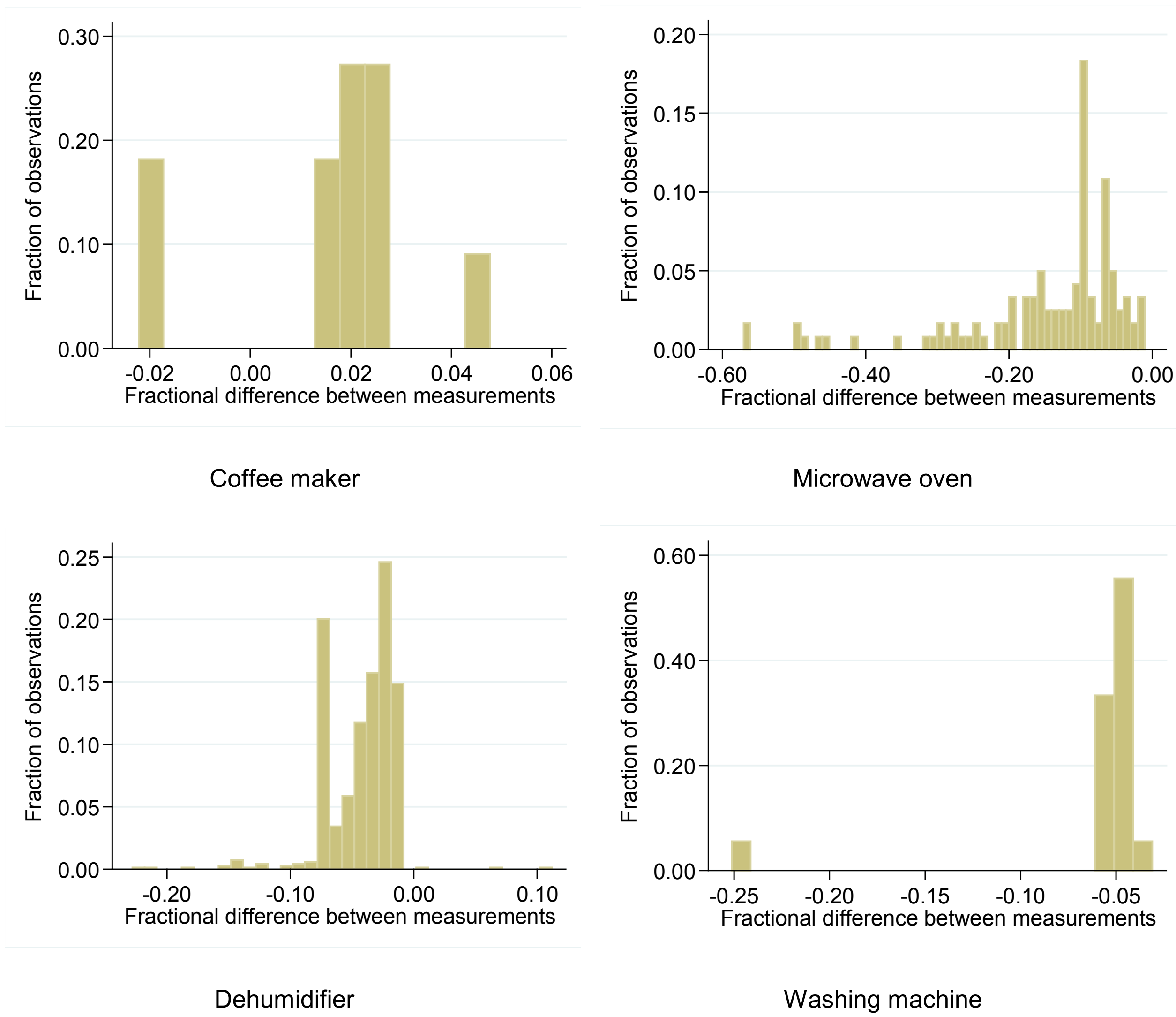


Figure 28. Comparison of energy measurements using two systems

The differences in measured values between the two systems are typically greater for the microwave oven than the other loads, with a mean difference of nearly 15% (Table 8). Typical differences for the other loads are much smaller. We can't explain why the difference is larger for one particular appliance. The appearance of many negative values may indicate a bias toward lower measured energy values with the Smartenit plug load system as compared to the WattNode system, but more comparisons would have to be made across a larger number of devices to reach a firm conclusion.

In general, these comparisons indicate reasonable agreement between two measurement systems. This supports the use of the Smartenit system for identifying operating cycles of specific loads, and for characterizing electrical energy use within an error of about 10%–15%. More evaluation work should be done before using the Smartenit system in applications where errors greater than 10% would be problematic.

Table 8. Differences in Energy Measurement With Two Systems

	Median Difference in Hourly Measured Energy Consumption, as a Fraction of WattNode System Value (%)	Mean Difference in Hourly Measured Energy Consumption, as a Fraction of WattNode System Value (%)
Coffee Maker	+1.9	+1.6
Microwave Oven	–10.0	–14.8
Dehumidifier	– 3.4	–4.2
Washing Machine	–5.0	–6.0

4 System Cost and Performance Discussion

The Smartenit field test system proved to be a fairly low-cost solution for specific data acquisition requirements. The tradeoffs are discussed below.

4.1 System Performance

4.1.1 Setup and Deployment Procedures Affect System Performance

The system performance as evaluated under the primary metric of failed queries/total queries was largely determined by the environmental constraints under which the system was deployed. Having procedures that included feedback about the environment during deployment helped with the field test's overall success. In some cases, a particularly noisy, highly attenuated, or highly dispersed environment proved particularly challenging for deployment.

Evaluating the available 2.4-GHz unlicensed spectrum during field deployment provided insight into the ambient noise floor associated with each ZigBee mesh network channel at that location. Choosing a ZigBee channel with a low noise floor helped in maintaining a network over the duration of the test. This deployment step helped with overall system performance—even with the acknowledgment that the RF interference will change over the course of the test and that the frequency sweep done is a localized point of reference.

After deploying and evaluating several use case tests, it was clear that mesh network node placement directly affected system performance. If a node was located behind or immediately adjacent to RF attenuation or RF reflecting objects (near-field RF effects) such as metal lath/plaster, steel studs, sheet metal, competing 2.4-GHz transmitters, and soil/potted plants, that node's communication with the mesh was affected. The performance of that node was also affected by increased distance from other nodes in the network. A way of evaluating the net effects of attenuation and noise was distilled to a single comparison number called LQI that is available during test setup. Keeping the mesh network nodes above some threshold (> 85 or so) appeared to reduce the number of failed queries. Iteratively adjusting node placement, adding repeaters and re-evaluating LQI did in many cases improve the system performance. Placing the plug load monitor on a short appliance extension cord helped in some cases to minimize near-field RF effects. Note that like RF noise floor evaluation, the LQI evaluation was a very specific reference state of the system during deployment and the state changes during the deployment. Deploying a system with different minimum LQI thresholds may prove an effective way of ensuring better performance, but this was not evaluated directly.

Raspberry Pi placement appeared to be more valuable than other nodes' placements. Placing the device more central and higher in the tested space seemed to increase the reliability of routed communications.

Points of failure for system setup and operations included:

- Failure on startup after a node is discovered but is deemed unknown by the system
- System becoming unresponsive on power failure (requiring UPS)

- Unknown complete system failure after several hours of successful operation, and inability to automatically recover (attributed to be a known transceiver firmware bug—but it remains an issue to be fully confirmed and resolved).

During setup, it was important to ensure the load being observed would not exceed the rating of the plug load monitor. In one case, a plug load monitor became damaged because it drew current over its allowable range.

Finally, the user-specified settings seemed to alter the success rate of queries for each test run. Tests with many nodes, few or no retries, and short timeouts resulted in higher failed queries.

4.1.2 Success Criteria for Obtaining Better System Performance

The deployment methods for issuing the use case test systems were not rigidly defined, and some trial and error was necessary to understand the effects of various settings. Communicating the lessons learned, and training field deployment staff on the environmental effects on the systems helped to improve each system's performance. Having personnel familiar with radio-based networks and effects on radio communications was important during the debugging of particular field deployment issues.

Systems generally appeared to perform well in the localized laboratory bench and the residential wood-frame construction environments, but performance suffered when the system was deployed in the shared commercial office space setting.

Performance seemed to improve with a fewer number of nodes deployed per system, with 8–12 nodes showing the most promise. Unfortunately, the high node count system was the one tested in the commercial office setting, so it is difficult to determine which system attribute contributed more against system performance.

Systems that had the latest known firmware for each device seemed to perform better, and the firmware changed for each device a number of times over the evaluation period. Firmware version updates were not automatic, making it possible to revert back to older firmware versions if desired—or to define an overall known platform based on specific firmware versions. Once a specific firmware set is stable for all deployment use cases, this "known good" firmware combination can be fixed ("version-pinned") in future deployments.

4.1.3 Suitability of Test System as a Tool for Field Research

Overall, the system appears to provide a new set of tools that complement the existing tools for field research. The system, as it stands, appears suitable for both short-term, rapid data gathering (~1 second resolution), and for more extended long-term monitoring. The caveat is that the deployment environment and installation choices greatly affect the success of long-term monitoring.

The Raspberry Pi platform appeared to perform stably when reliable power was maintained. In the cases where a specific plug load monitor lost power, the plug did appear to automatically rejoin when power was restored. In the case where it was able to rejoin, the accumulated energy values appeared to be maintained during the communication outage. If it was unable to, it did not make the Raspberry Pi data acquisition platform unstable.

This system would not be ideally suited for power quality measurements, where rapid sampling is desired over the course of each phase cycle. In these cases, a single-outlet power quality monitoring device is recommended.

This system would not be ideally suited for cases where 100% uninterrupted data sampling and delivery are required. The nature of this low-power wireless network makes it nearly impossible to guarantee 100% sample delivery. Several deployment strategies can be employed to improve system performance, but clearly the system at the time of this writing will at times provide missed data points.

4.2 System Cost

The total estimated cost for the system, including hardware and deployment labor, is $4100. Almost $1600 of this total is labor, assuming a rate of $100/hour. Also, this example assumes 12 monitors and a cellular link, which separately cost almost $1000 (i.e., 24 monitors will cost almost $2000 to purchase). The following sections break out costs by base system, plug load quantities/repeaters, communication options, deployment tools, and labor estimates.

4.2.1 Base System Costs

Table 9 lists basic system components and their estimated cost in U.S. dollars, excluding the cost of plug load monitor or repeater nodes.

Table 9. Base System Cost Breakout

Description	Quantity	Unit Cost	Total
Raspberry Pi SBC	1	$35	$35
SBC Enclosure	1	$10	$10
SmartenIt USB ZigBee Transceiver	1	$50	$50
Power Supply + Cable	1	$8	$8
SD Card (16Gb)	1	$30	$30
UPS Power Strip	1	$50	$50
		Total:	**$343**

4.2.2 Mesh Network Routers and Repeaters

Table 10 lists the incremental costs of each plug load monitor ZigBee router and ZBREs.

Table 10. Mesh Network Router and Repeater Costs, Per Unit

Description	Unit Cost
Plug Load Monitor, ZBMPlug15	$80
ZBRE	$50

4.2.3 External Communication Options

External data communications are an effective way to automate data delivery, verify operation, and remotely administer tests during the research project—though they aren't required for the system's basic operation. Table 11 lists two external communication cost estimates for cellular modem and/or DSL/cable modem.

Table 11. External Communications Options

Description	Quantity	Unit Cost	Total
Cellular Modem Option			
Raven XT	1	$400	$400
Antenna	1	$150	$150
12V Power Supply	1	$30	$30
USB Mini Data Cable	1	$1	$1
Data SIM Card	1	$10	$10
Monthly Data Plan	*var.*	$30/mo	$30/mo
		Total:	**$593 + $30/mo**
Cable/DSL Modem Option			
Cable/DSL Modem	1	$50	$50
Cabling	2	$5	$5
Monthly Data Plan	*var.*	$30/mo	$30/mo
		Total:	**$55 + $30/mo**

4.2.4 Deployment Tools

Deployment tools are costs associated with the number of field installers required for a study. Spare part costs are excluded in the following breakout, and the field installer is assumed to have a laptop already provided.

Table 12. Field Deployment Tools

Description	Quantity	Unit Cost	Total
Ethernet LAN Router + Patch Cables	1	$35	$35
MetaGeek Chanalyzer Software + Wi-Spy 2.4GHz valyzer	1	$200	$200
		Total:	**$235**

4.2.5 Labor Estimates

Several steps are required to orchestrate field deployment, including purchasing and procurement, system staging, field installation/bring-up, field monitoring/remote troubleshooting, and decommissioning. The following is one estimated breakout of costs, assuming $100/hr labor costs. Costs do not include analysis or reporting. Actual costs may vary greatly depending on the site location, hourly rate, duration of the study, etc.

Table 13. Labor Cost Estimate

Description	Quantity (hour)	Unit Cost (per hour)	Total
Purchasing and Procurement	2	$100	$200
System Staging, Shipping	2	$100	$200
Field Installation/Bring-Up	2.5	$100	$250
Field Monitoring/Remote Troubleshooting	8	$100	$800
Decommissioning	1	$100	$100
		Total:	**$1550**

4.2.6 Example Deployment Cost Estimate

An example deployment may be the monitoring of a single-family detached home with 12 MELs of interest. During field installation, it is determined that two ZigBee repeaters are necessary to bring the LQI levels above 85. A cellular modem is chosen to maintain data delivery and connectivity over the course of 12 months. Table 14 shows the total costs.

Table 14. Sample Deployment Cost Estimate

Description	Quantity	Unit Cost	Total
Base System	1	$343	$343
ZigBee Plug Load Monitors	12	$80	$960
ZBREs	2	$50	$100
Cellular Link	1 + 12 mo	$550 + $360	$910
Deployment Tools	1	$235	$235
Labor	1	$1550	$1550
		Total:	**$4098**

5 Summary and Future Work

We established the performance requirements for a wireless electrical plug load monitoring system usable in residential and light commercial settings, and selected a commercially available system that appeared to offer the desired functionality for electric plug load monitoring in field research applications.

Much of the effort on the project was directed toward development of software to manage repeated querying of a number of end nodes, and to make this software helpful to the user with features such as recording the identity of each end node (a name provided by the user) in data files, and information on the delay from the start of each query cycle to the response from each individual end node. The release by the hardware manufacturer of several firmware updates during the course of the project made our software development more difficult. The switch to the Raspberry Pi single board computer and D-Bus internal architecture improved system performance, but also required additional development effort.

The system as currently developed was evaluated based on the rate of successful data queries, reliability over a period of days to weeks, and accuracy. It offers good overall performance when deployed with up to 10 end nodes in a single-family residential environment; deployment with more nodes and in a commercial environment is currently less robust. As such, it offers a tool that can be used in selected field research projects, with the recommendation that system behavior is observed throughout the data collection period to ensure continued reliable operation.

Future work on this technology might usefully include additional testing of the system as currently configured and further development of the system on the current hardware platform. Some specific suggestions include:

- Perform additional testing to more clearly establish the effects of RF noise, building construction type, and number of nodes on overall system performance.
- Gain further understanding of ZigBee firmware protocols, especially as they concern repeated attempts to complete a transaction after an initial failure (a possible cause of catastrophic system failures).
- Experiment with more routine use of range extenders in commercial building environments, to determine their effectiveness in overcoming RF transmission issues.
- Experiment with positioning of end nodes and gateway relative to one another and to nearby objects likely to affect RF communications. Working with the manufacturer to explore alternative antenna configurations that could be used at the gateway to increase signal strength and sensitivity.[14]
- Perform a more quantitative evaluation of the use of network link quality investigation (e.g., using the Smartenit LQI available through its Web interface), and enforce minimum link quality values during system setup.

[14] Antennas are integral to the end nodes and to USB plug-in transceiver used with Raspberry Pi gateway, and we did not attempt any modifications.

- Rewrite gateway operating code to accommodate end nodes that are initially discovered as part of the wireless network, but fail to respond during operating software startup.
- Draft a complete user's guide, including methods for evaluating the suitability of a building for use of wireless monitoring, setup procedures, and diagnostic/troubleshooting methods.

The field of building monitoring via wireless technology is clearly developing quickly. New products offering improved functionality and/or lower cost may enter the market at any time. Before proceeding with significant additional work on the current system, it would be wise to survey current product offerings for alternatives. Although the development of a functional system based on new hardware would likely take some significant effort, experience gained under the current project should allow greater efficiencies in the process.

References

Bensch, Ingo, Scott Pigg, Karen Koski, and Rana Belshe. *Electricty Savings Opportunities for Home Electronics and Other Plug-In Devices in Minnesota Homes.* Madison, W.I.: Energy Center of Wisconsin, 2010.

Hendron, Robert, and Mark Eastment. "Development of an Energy-Savings Calculation Methodology for Residential Miscellaneous Electric Loads." *ACEEE Summer Study on Energy Efficiency in Buildings.* Washington, D.C.: ACEEE, 2006.

Roth, Kurt, Kurtis Mckenney, Chris Paetsch, and Ratcharit Ponoum. "U.S. Residential Miscellaneous Electric Loads Electricity Consumption." *ACEEE Summer Study on Energy Efficiency in Buildings.* Washinton, D.C.: ACEEE, 2007.

U.S. Energy Information Administration. *Annual Energy Outlook 2013.* Annual Report, Washington, D.C.: U.S. Department of Energy, 2013.

Appendix A: Plug Load Monitoring Devices

The following table identifies 27 plug load monitoring or related solutions, most of which claimed some type of wireless interface, as identified in the marketplace as of February 2012. Product offerings may have changed since initial search.

II	Manufacturer	Model	Device Form Type
1	EnergyHub	EnergyHub Socket	Wall Plug (US Plug: NEMA 5-15, 3-pin grounded)
2	PMI	Eagle 120	Wall Plug
3	SafePlug	SafePlug Model 1202	Wall Plug
4	SimpleHomeNet	ZOE-MP1 (#5010P), among others in 5010x product line	Wall Plug
5	Digi International, Inc.	Xbee Smart Plug ZB, XR-Z14-CW2P6	Wall Plug
6	Entek	TC Series	Wall Mount and Wall Plug
7	HAI	89A00-1SHT 15A Plugin Load Control Module	Wall Plug
8	E-Mon	Wireless Monitor	Wall Mount with CTs
9	JetLun	Appliance Gauge RD75613 (homeplug and ZB) / RD75615 (zigbee only)	Wall Plug
10	JetLun	Circuit Meter RD71219(HomePlug)/ RD77719 (ZigBee)	Wall Mount with CTs
11	JetLun	Panel Meter RD77720/ RD71220/ RD77724/ RD71224	Wall or Panel Mount
12	RCS	PMC12M-ZW, PCM12M-ZB	Wall Plug
13	Omnipas	ECOTAB	Korean Wall Plug
14	OmniSystem	Omni Plug OWMH-12-15C	Korean Wall Plug
15	Schneider Electric	Wiser EER40200 Smart Plug	Wall Plug
16	Tendril	Volt Outlet	Wall Plug
17	ThinkEco	Modlet SE	Wall Plug
18	Develco	ZigBee Wallmount Relay	Wall Plug
19	WattsUpMeters	Watts Up? .NET, 72222	Corded Floor Outlet
20	Aeon Labs	Smart Energy Switch	Wall Plug
21	PowerHouse Dynamics	eMonitor	Breaker Panel access w/CT clamps; Sub-ckt only
22	P3 International	P4220 Kill-A-Watt Wireless consumption sensor	Wall Plug
23	Smarthome	2423A1 iMeter Solo	Wall Plug
24	Plogg International	Plogg-Zgb	Wall Plug
25	PowerSave, Inc.	Individual Appliance Monitor (IAM)	Wall Plug
26	DoSafe Corporation	Not Indicated	Wall Plug
27	GE	Nucleus Energy Sensor	Wall Plug

Appendix B: Field Trial Results Graphs

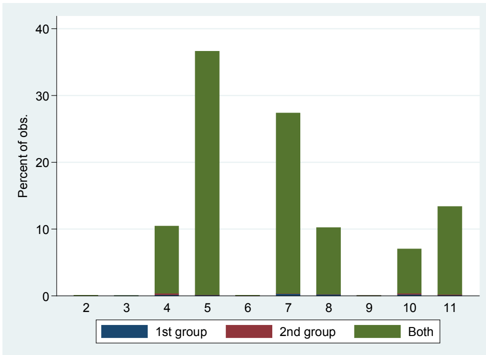
The following pages include graphs characterizing the performance of the Smartenit system under the tests discussed in the body of the report. A guide to the organization and interpretation of the graphs follows.

Percent failed observations, by node (the x axis number is an abbreviation for node number; e.g. “11” corresponds to “Plug11”). *Failed observations are categorized (with color) by whether the first, second, or both groups of values delivered in response to a query were missing. Provides a visual characterization of the quality of communication with each node. X axis values are unique node ID numbers.*	Percent failed observations, by hour of day (by node). *Panel graph by individual node showing percent of failed observations, categorized (with color) by whether the first, second, or both groups of values delivered in response to a query were missing. Diurnal patterns should show up on these graphs.*
Percent failed observations, by hour of test. *Percent of failed observations summarized by each hour of the test period. Provides visualization of evolution of system behavior over time, including catastrophic collapse in some cases.*	Percent failed observations, by hour of test (by node). *Panel graph by individual node showing percent of failed observations summarized by each hour of the test period. Similar visualization of evolution of behavior, by individual node.*
Correlation between failure rate and average response time for all observations. *Graphs show expected increase in response time as a function of failure rate, with maximum time bounded by timeout established in code.*	Correlation between failure rate and response time (99^{th} %) for successful observations. *99^{th} percentile of response times for successful queries, allows exploration of response times without the effects of failed queries.*
Distribution of response time among successful observations (by node). *Distribution of response times, offering another view of network quality*	Response time (99^{th} %) by hour for successful observations (by node). *99^{th} percentile of response times for successful queries over each hour of test, providing another view of evolution of performance over time.*

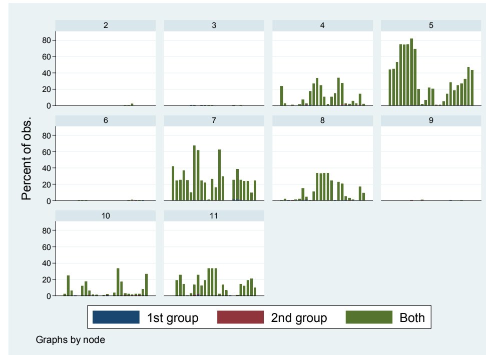
Note that axis scaling is allowed to vary across graphs from different tests. Failure rates in particular differ by several orders of magnitude across the tests.

Test MAD01-1

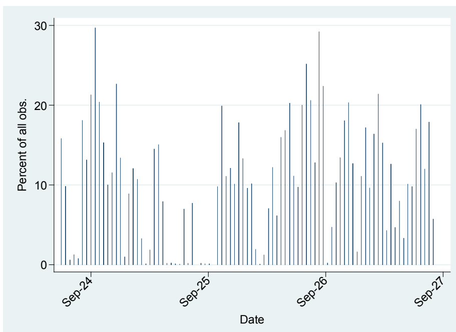
Percent failed obs, by node



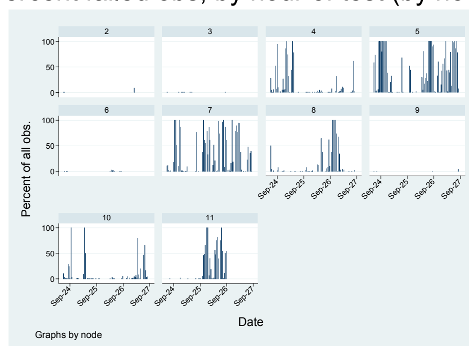
Percent failed obs, by hour of day (by node)



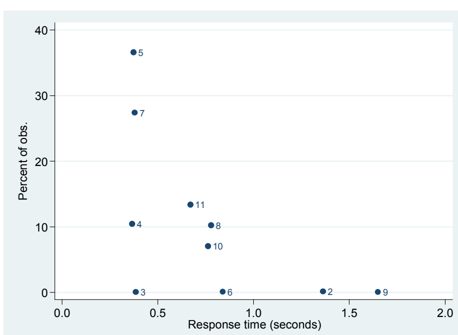
Percent failed obs, by hour of test



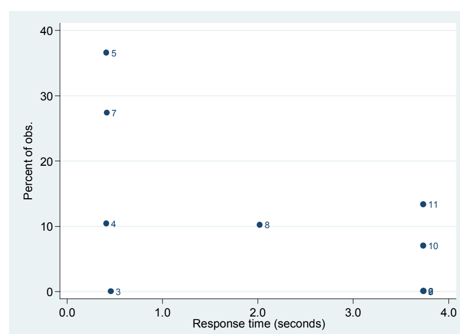
Percent failed obs, by hour of test (by node)



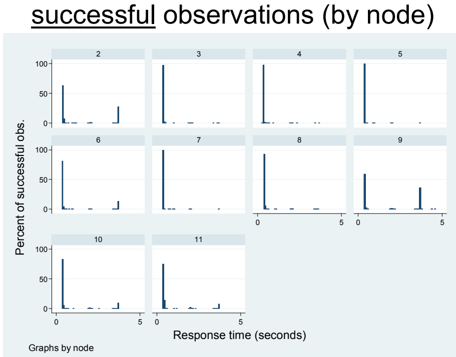
Correlation between failure rate and mean response time for <u>all</u> obs



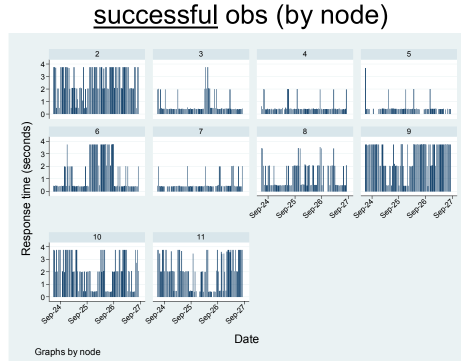
Correlation between failure rate and response time (99^{th} %) for <u>successful</u> obs



Distribution of response time among <u>successful</u> observations (by node)

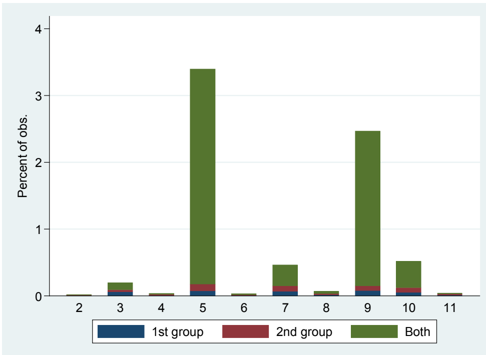


Response time (99th %) by hour for <u>successful</u> obs (by node)

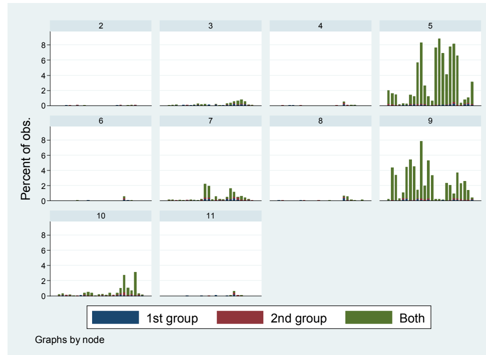


Test MAD01-2

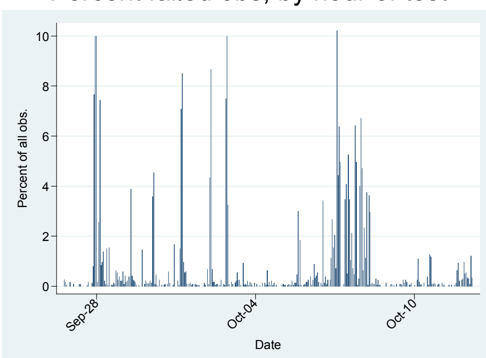
Percent failed obs, by node



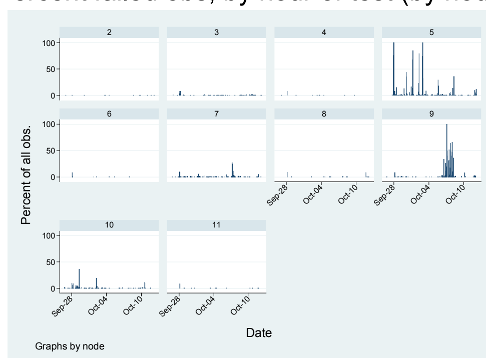
Percent failed obs, by hour of day (by node)



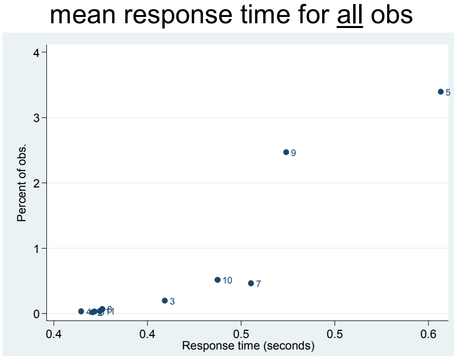
Percent failed obs, by hour of test



Percent failed obs, by hour of test (by node)

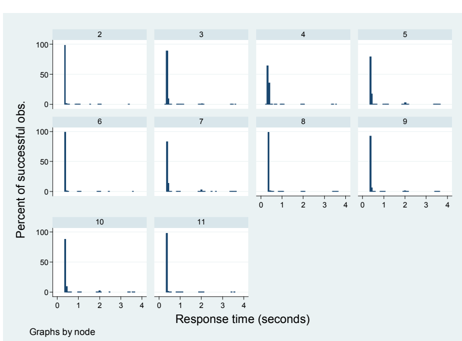


Correlation between failure rate and mean response time for all obs

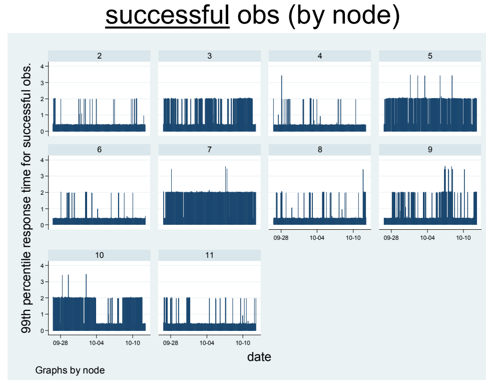


Correlation between failure rate and response time (99th %) for successful obs

Distribution of response time among successful observations (by node)

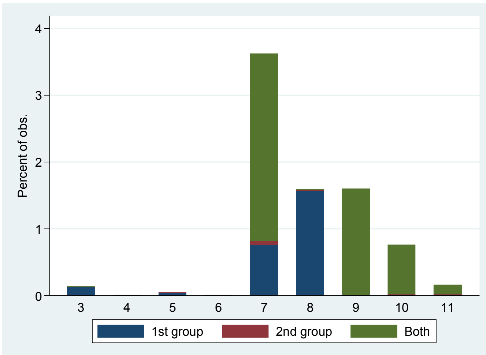


Response time (99th %) by hour for successful obs (by node)

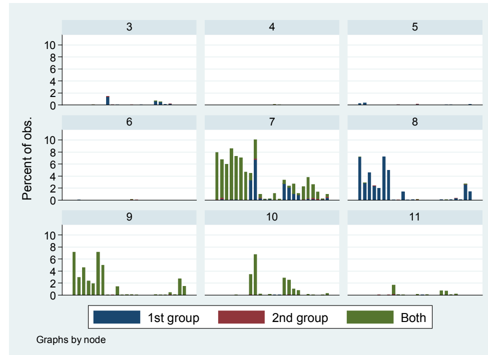


Test MAD01-3

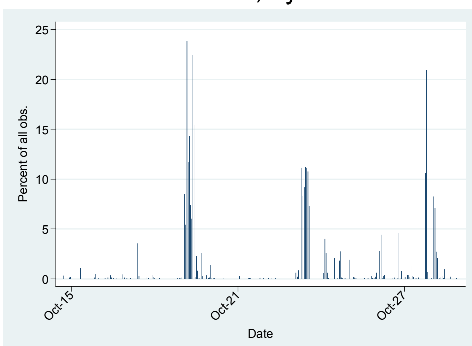
Percent failed obs, by node



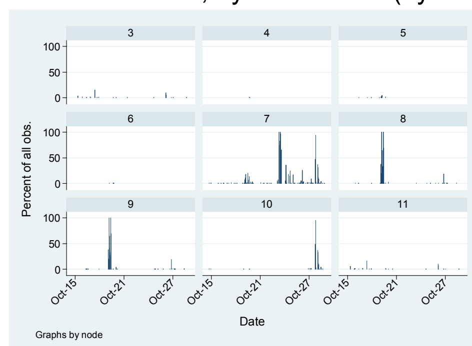
Percent failed obs, by hour of day (by node)



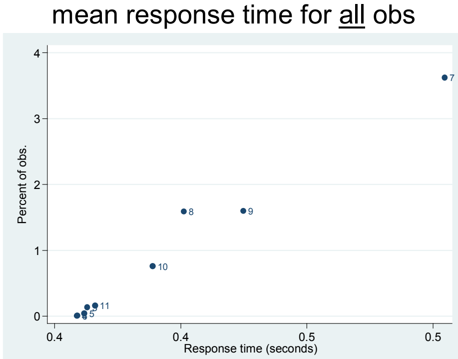
Percent failed obs, by hour of test



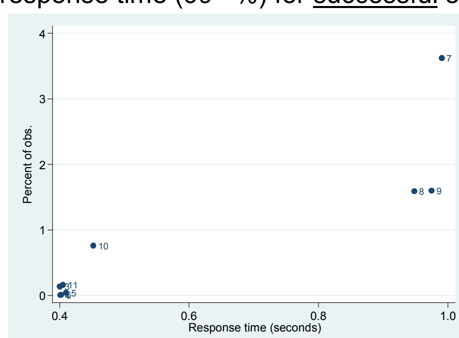
Percent failed obs, by hour of test (by node)



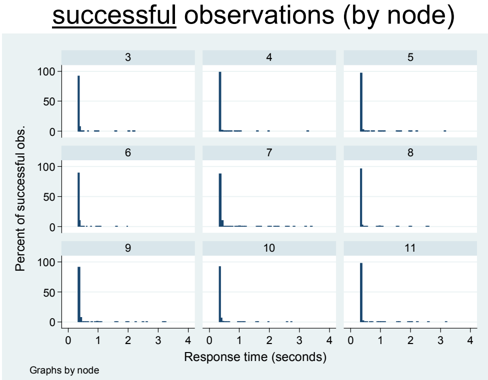
Correlation between failure rate and mean response time for <u>all</u> obs



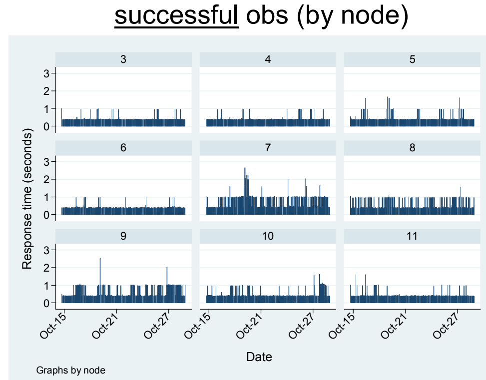
Correlation between failure rate and response time (99th %) for <u>successful</u> obs



Distribution of response time among <u>successful</u> observations (by node)

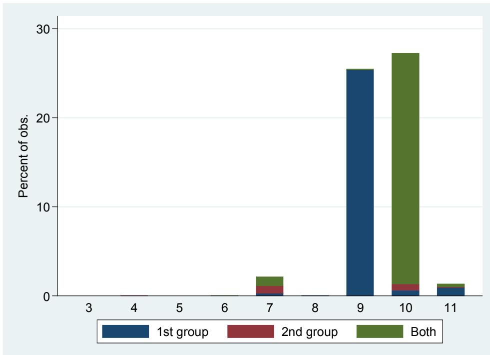


Response time (99th %) by hour for <u>successful</u> obs (by node)

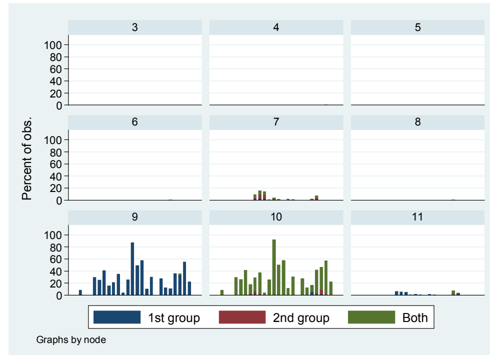


Test MAD01-4

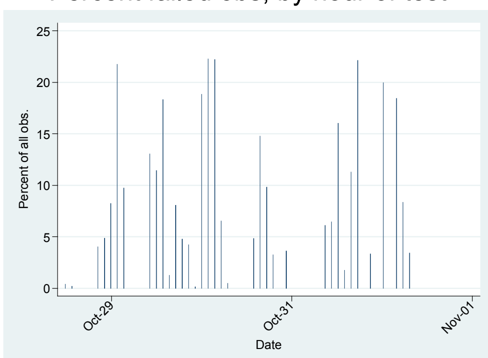
Percent failed obs, by node



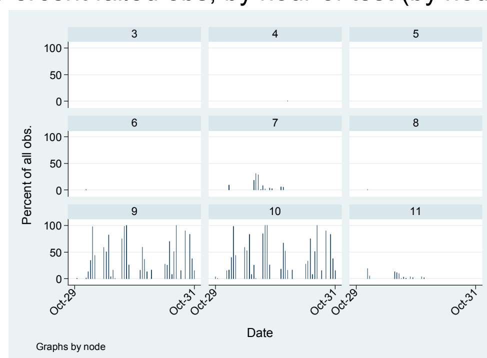
Percent failed obs, by hour of day (by node)



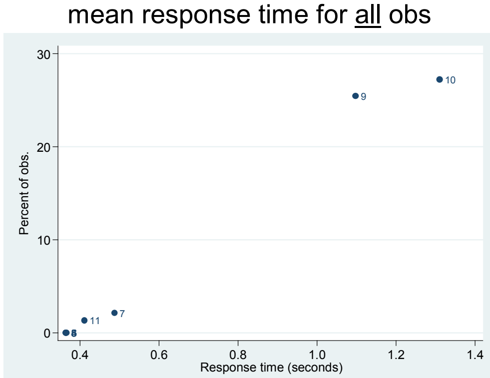
Percent failed obs, by hour of test



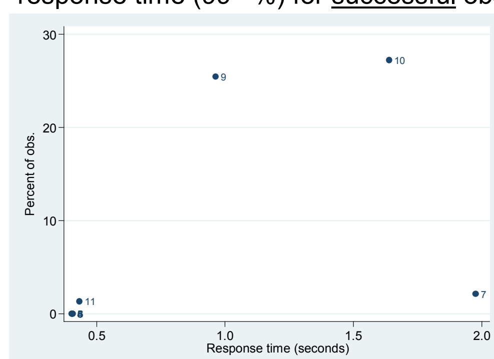
Percent failed obs, by hour of test (by node)



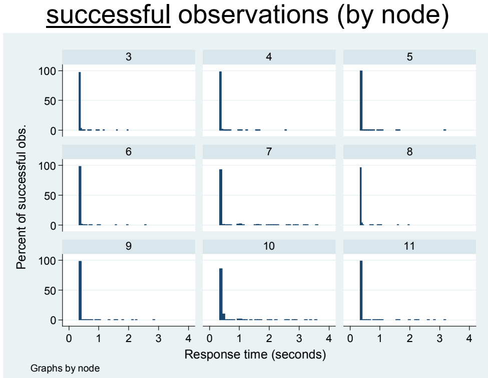
Correlation between failure rate and mean response time for <u>all</u> obs



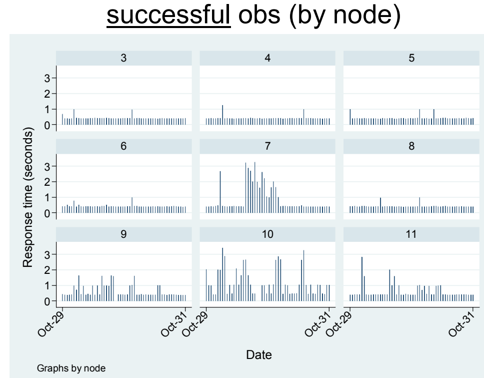
Correlation between failure rate and response time (99th %) for <u>successful</u> obs



Distribution of response time among <u>successful</u> observations (by node)

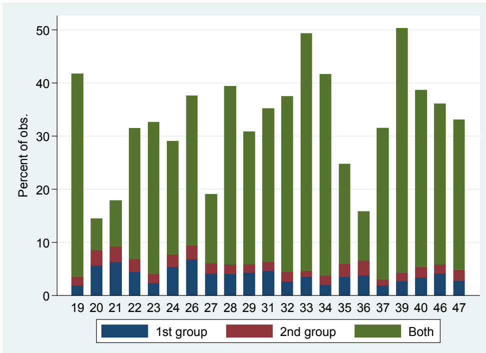


Response time (99th %) by hour for <u>successful</u> obs (by node)



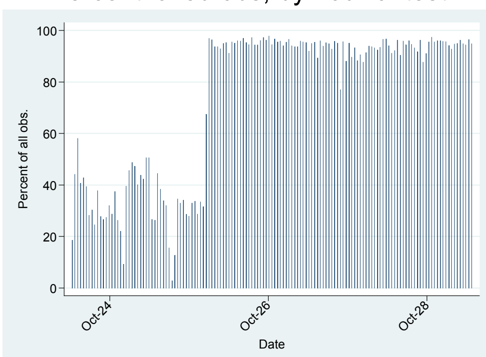
Test MAD02-1

Percent failed obs, by node, *while system was working*

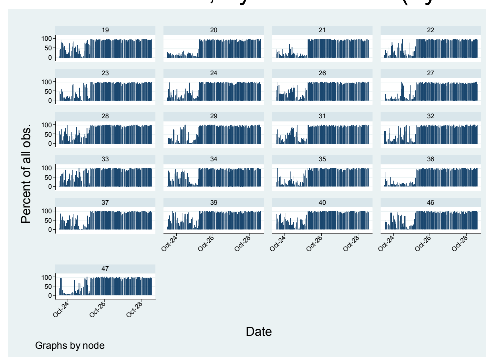


Percent failed obs, by hour of day (by node) , *while system was working*

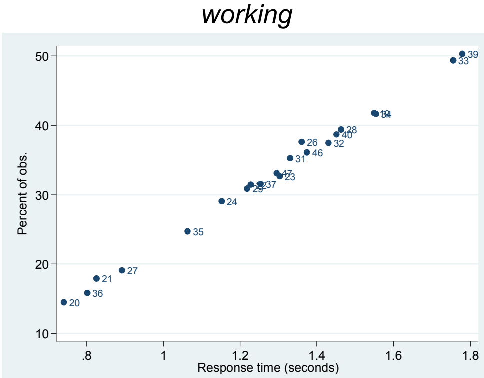
Percent failed obs, by hour of test



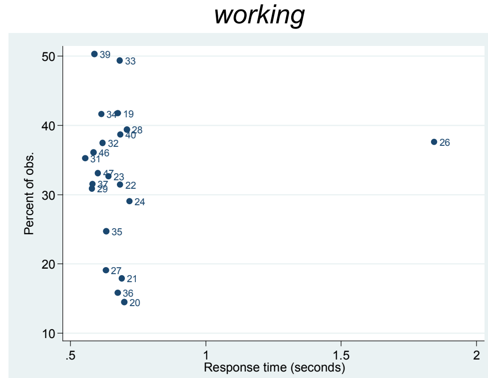
Percent failed obs, by hour of test (by node)



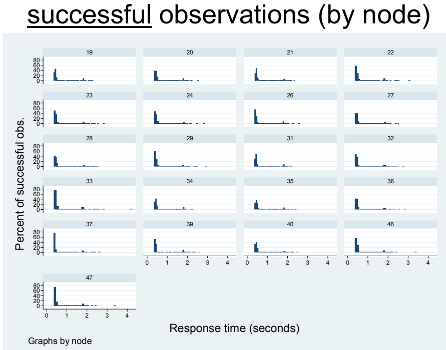
Correlation between failure rate and mean response time for <u>all</u> obs, *while system was working*



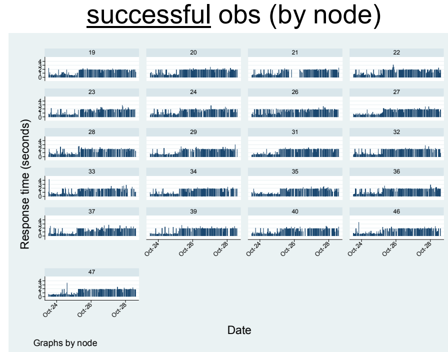
Correlation between failure rate and response time (99^{th} %) for <u>successful</u> obs, *while system was working*



Distribution of response time among <u>successful</u> observations (by node)

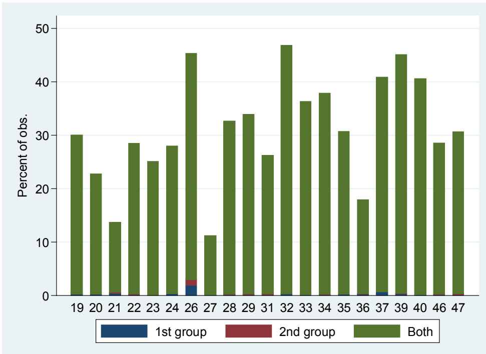


Response time (99th %) by hour for <u>successful</u> obs (by node)



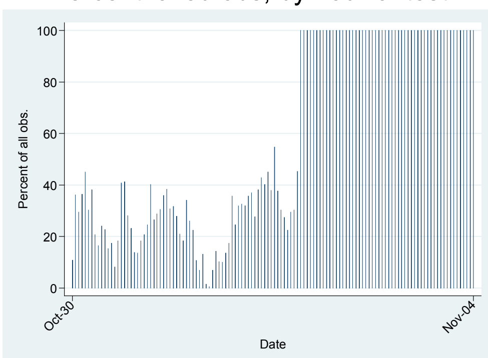
Test MAD02-2

Percent failed obs, by node,
while system was working

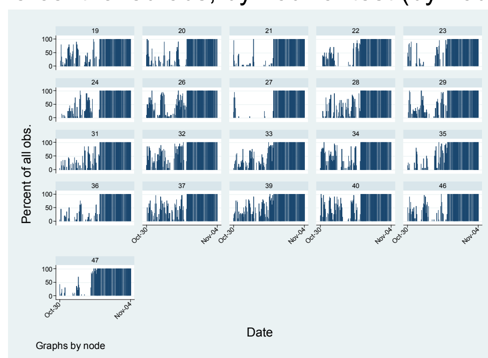


Percent failed obs, by hour of day (by node),
while system was working

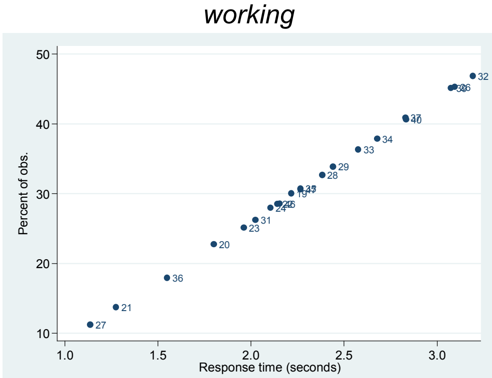
Percent failed obs, by hour of test



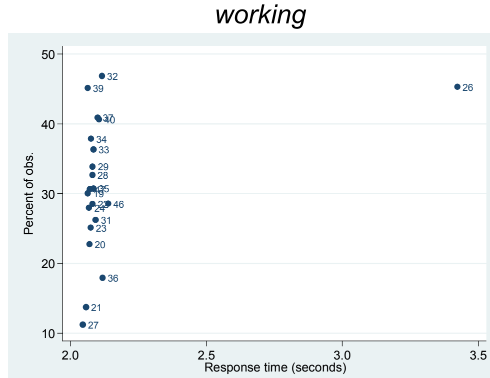
Percent failed obs, by hour of test (by node)



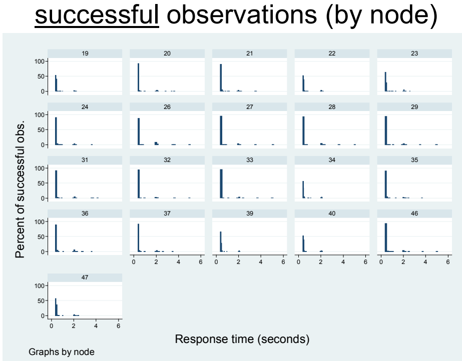
Correlation between failure rate and mean response time for <u>all</u> obs, *while system was working*



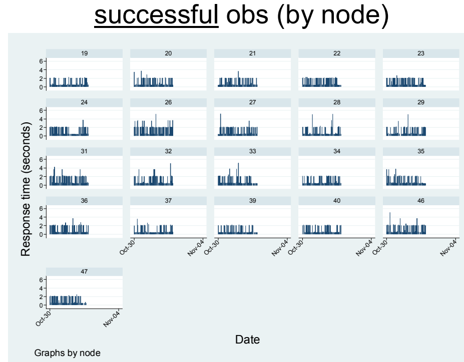
Correlation between failure rate and response time (99^{th} %) for <u>successful</u> obs, *while system was working*



Distribution of response time among <u>successful</u> observations (by node)

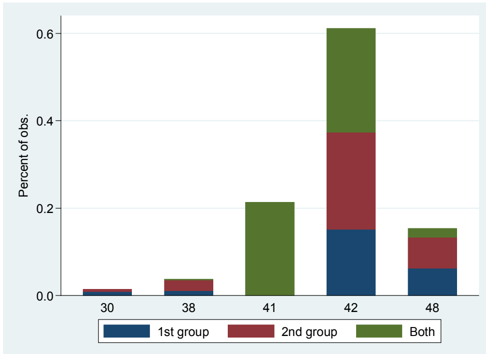


Response time (99th %) by hour for <u>successful</u> obs (by node)

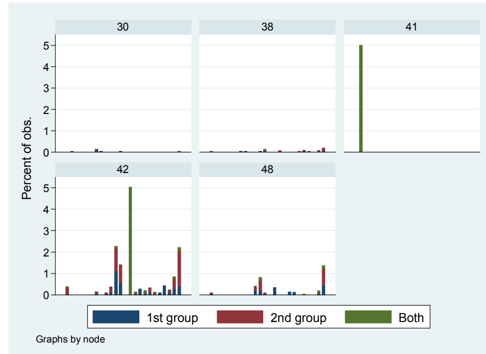


Test MPLS02-1

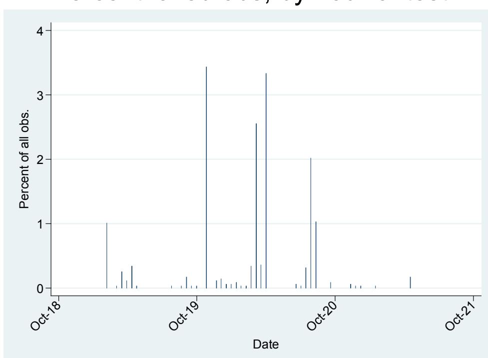
Percent failed obs, by node,
while system working



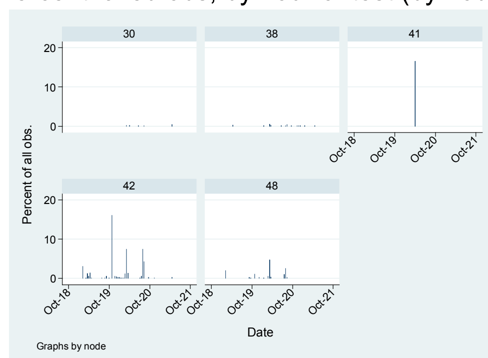
Percent failed obs, by hour of day (by node),
while system working



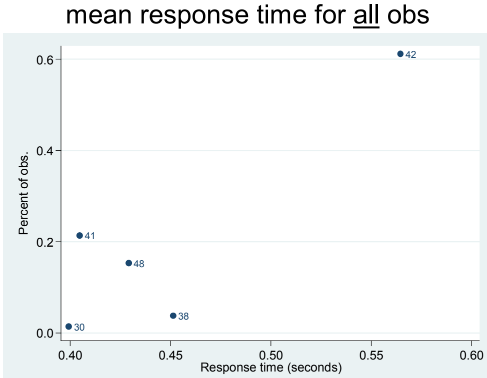
Percent failed obs, by hour of test



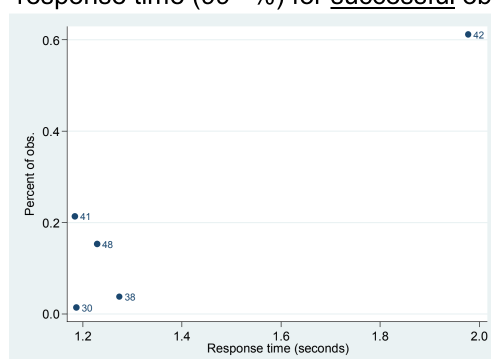
Percent failed obs, by hour of test (by node)



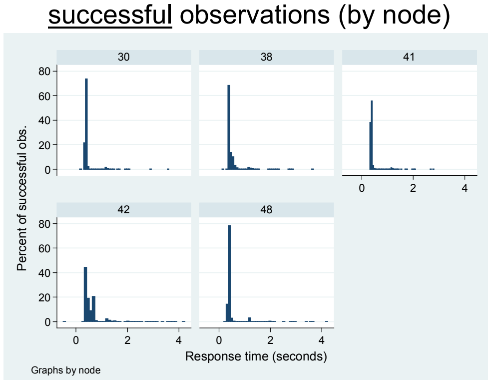
Correlation between failure rate and
mean response time for <u>all</u> obs



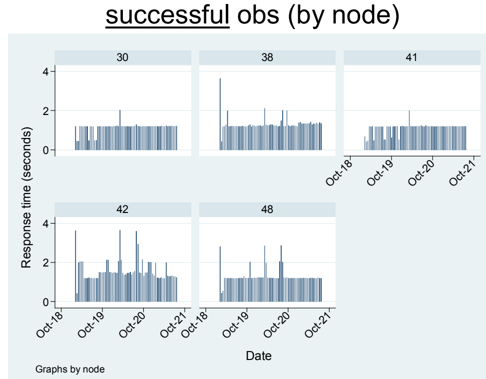
Correlation between failure rate and
response time (99th %) for <u>successful</u> obs



Distribution of response time among
<u>successful</u> observations (by node)

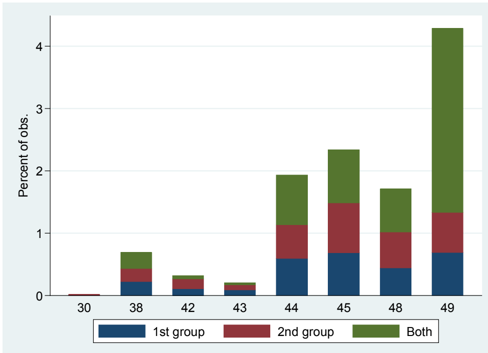


Response time (99th %) by hour for
<u>successful</u> obs (by node)

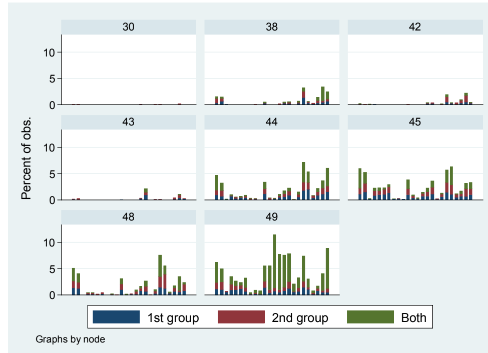


Test MPLS02-2

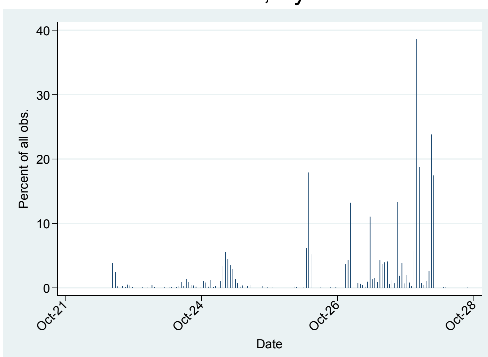
Percent failed obs, by node, while system working



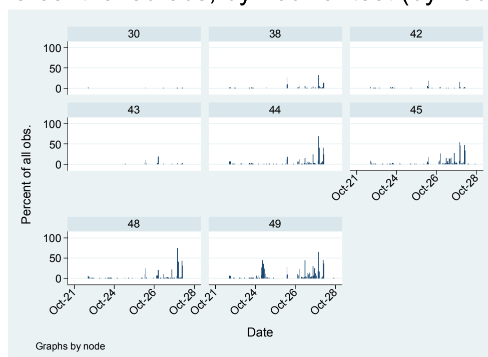
Percent failed obs, by hour of day (by node), while system working



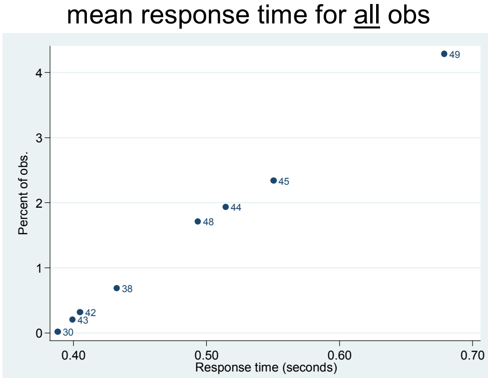
Percent failed obs, by hour of test



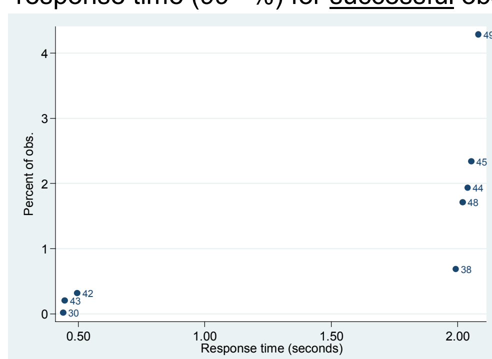
Percent failed obs, by hour of test (by node)



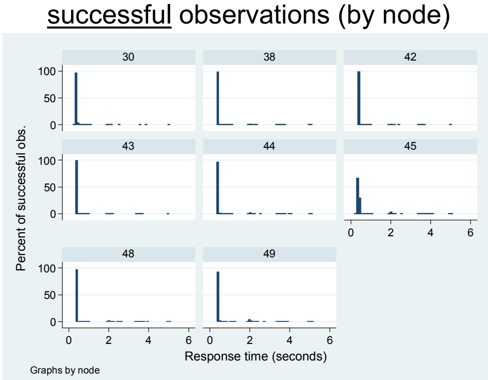
Correlation between failure rate and mean response time for <u>all</u> obs



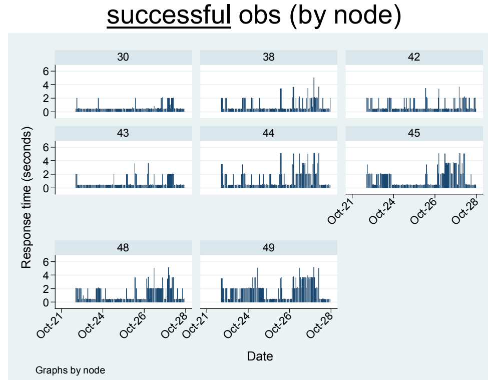
Correlation between failure rate and response time (99th %) for <u>successful</u> obs



Distribution of response time among <u>successful</u> observations (by node)

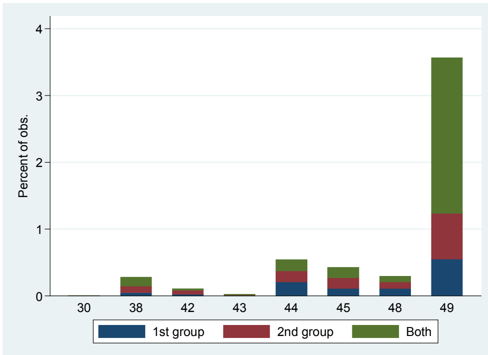


Response time (99th %) by hour for <u>successful</u> obs (by node)

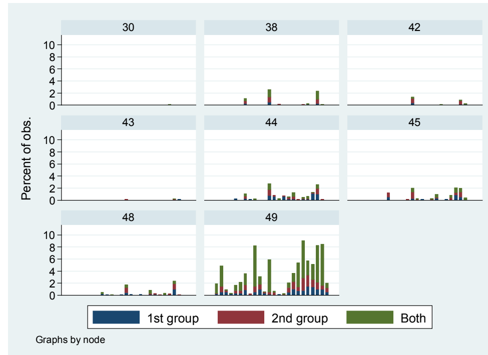


Test MPLS02-3

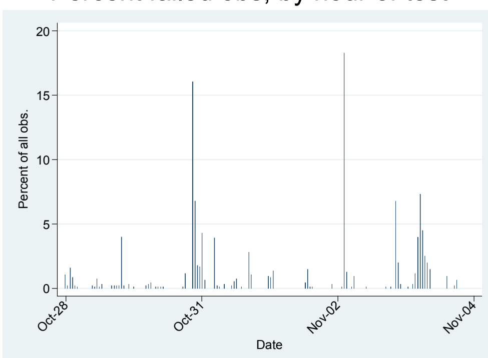
Percent failed obs, by node



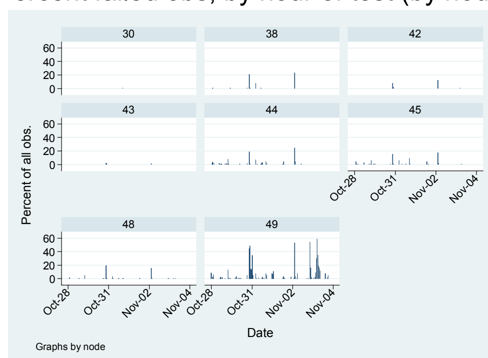
Percent failed obs, by hour of day (by node)



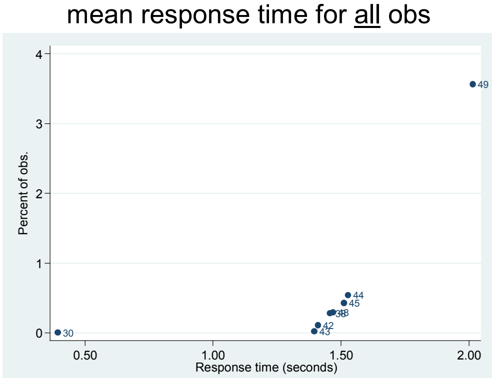
Percent failed obs, by hour of test



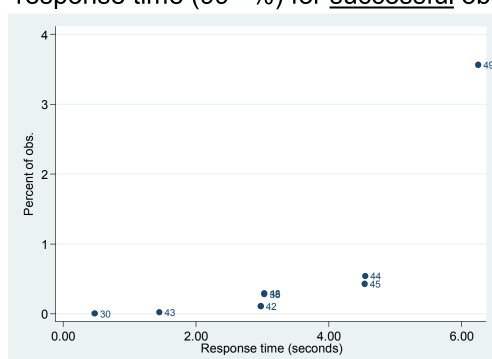
Percent failed obs, by hour of test (by node)



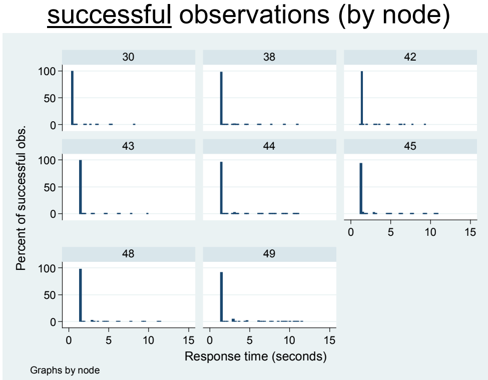
Correlation between failure rate and mean response time for <u>all</u> obs



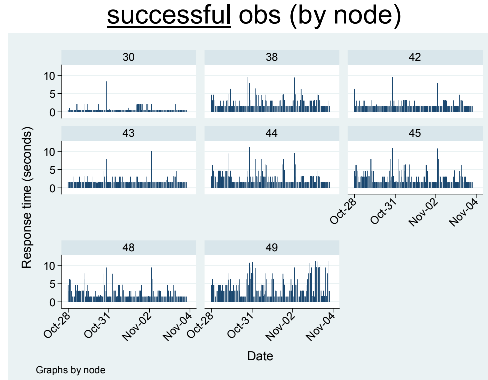
Correlation between failure rate and response time (99th %) for <u>successful</u> obs



Distribution of response time among <u>successful</u> observations (by node)

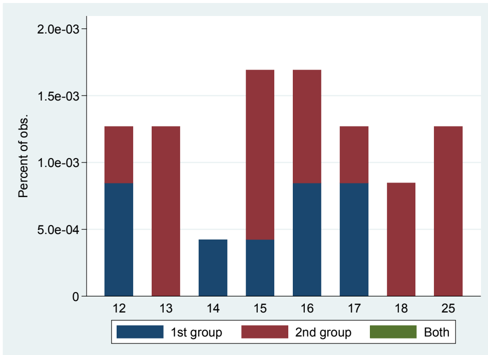


Response time (99th %) by hour for <u>successful</u> obs (by node)

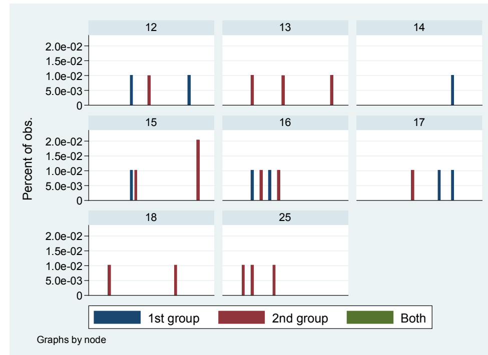


Test MPLS01-1

Percent failed obs, by node

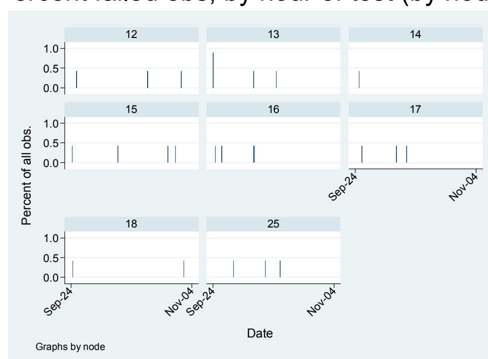


Percent failed obs, by hour of day (by node)

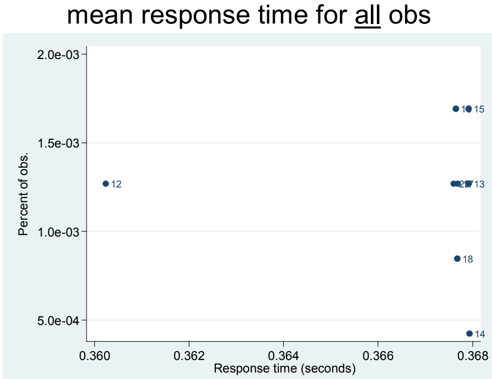


Percent failed obs, by hour of test

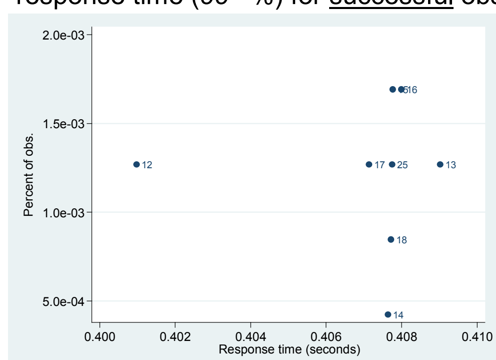
Percent failed obs, by hour of test (by node)



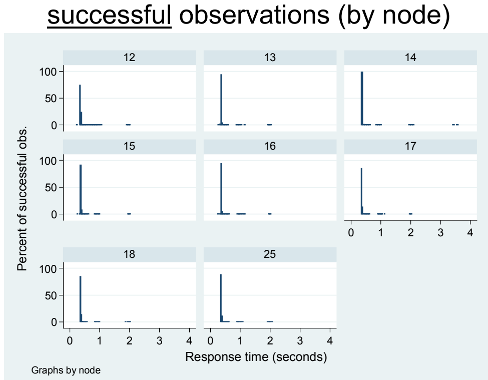
Correlation between failure rate and mean response time for <u>all</u> obs



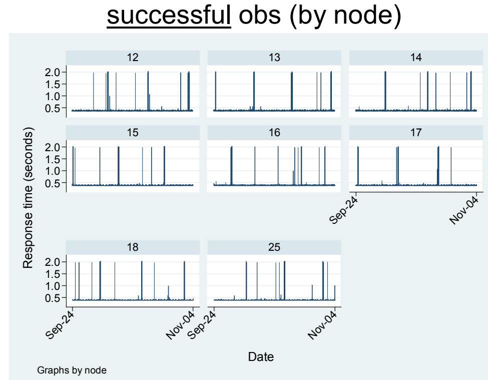
Correlation between failure rate and response time (99th %) for <u>successful</u> obs



Distribution of response time among <u>successful</u> observations (by node)

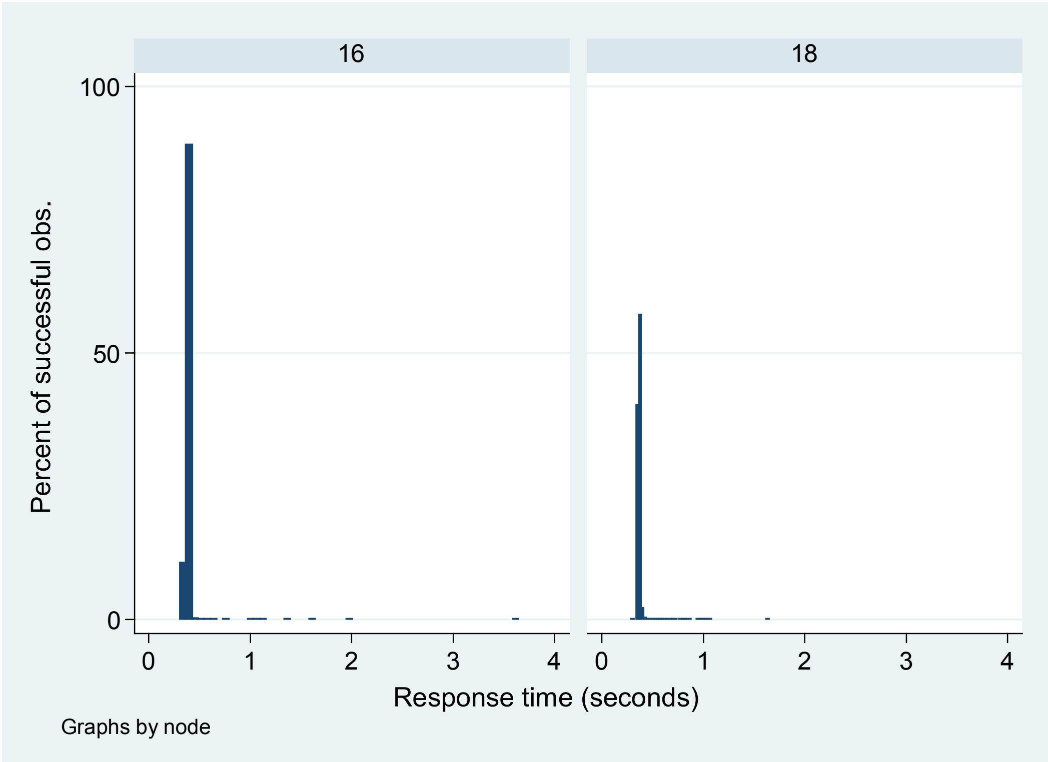


Response time (99th %) by hour for <u>successful</u> obs (by node)



Test MPLS01-2

Response Time Distribution, by Node



Appendix C: Sample Data

The following provides an example excerpt of a data file from a system with two end nodes. Each record includes timestamp and query failure information, followed by data from each end node in order, including the six energy measurement parameters (kWh, power, power factor, voltage, current, and line frequency), and the time measured from the start of the query cycle to the response from that node. It is a plaintext comma-separated value file type. The second header line is wrapped in this screen capture.


```
"TOA5","SimpleHomeNet","HarmonyGateway","12345","6005A","HarmonyGateway_Query.py","12345","1_Second_Results"
"TIMESTAMP","RECORD","FAILED_QUERIES","kWH_Plug18","Watts_Plug18","PF_Plug18","Volts_Plug18","Amps_Plug18","LineFreq_Plug18","TimeDelta_Plug18","kWH_Plu
g16","Watts_Plug16","PF_Plug16","Volts_Plug16","Amps_Plug16","LineFreq_Plug16","TimeDelta_Plug16","ScanDuration"
"2013-11-04 20:17:25",1,0,0.00864,21.3,0.97,113.15,0.19,59.9,0.365430831909,0.12045,122.7,0.97,115.62,1.05,59.9,0.756311893463,0.756443977356
"2013-11-04 20:17:26",3,0,0.00864,21.0,0.98,113.0,0.18,59.9,0.371116876602,0.12062,122.7,0.97,115.58,1.05,59.9,0.745131969452,0.745268821716
"2013-11-04 20:17:27",5,0,0.00866,21.3,0.97,112.95,0.18,59.9,0.374243974686,0.12062,122.4,0.97,115.36,1.05,59.9,0.750108003616,0.750239849091
"2013-11-04 20:17:28",7,0,0.00866,21.0,0.98,112.74,0.18,59.9,0.366488933563,0.12062,122.4,0.96,115.08,1.05,59.9,0.742457866669,0.74259185791
"2013-11-04 20:17:29",9,0,0.00866,21.0,0.98,112.8,0.18,59.9,0.360207080841,0.12062,122.1,0.97,115.05,1.05,59.9,0.734325170517,0.736908197403
"2013-11-04 20:17:30",11,0,0.00866,21.0,0.98,112.71,0.18,59.9,0.36651802063,0.12062,121.8,0.97,114.84,1.05,59.9,0.756945848465,0.757097959518
"2013-11-04 20:17:31",13,0,0.00866,21.0,0.98,112.74,0.18,59.9,0.388242006302,0.12078,121.8,0.97,114.82,1.05,59.9,0.778038024902,0.778167009354
"2013-11-04 20:17:32",15,0,0.00868,21.0,0.98,112.84,0.18,59.9,0.365505933762,0.12078,122.1,0.97,114.82,1.05,59.9,0.733167886734,0.733298778534
```