Revealing Protocol Information and Activity from Energy Instrumentation in Wireless Sensor Network

Dong Han[1], Omprakash Gnawali[1], and Abhishek Sharma[2]

[1] University of Houston, USA
{donny,gnawali}@cs.uh.edu
[2] NEC Laboratories America, Inc.
absharma@nec-labs.com

Abstract. In this paper, we present a novel approach to study and reveal network and protocol information from energy instrumentation in wireless sensor network. Unlike prior approaches which focused on analyzing the aggregate statistics of energy efficiency of a network or a protocol, our approach aims at revealing network protocols, application workloads, and topology information by fine-grained energy instrumentation on the nodes. We design a set of features based on various aspects of energy data and use those features to classify and reveal network activity. Results from experiments on three testbeds indicate that our approach can achieve up to 97% accuracy to identify the routing protocols, and infer the network topology with 98% accuracy.

Keywords: Power Measurements; Wireless Sensor Networks; Protocols;

1 Introduction

Energy instrumentation has a long history of research in wireless sensor network. Energy efficient protocols and applications are one of the objectives of Wireless Sensor Network (WSN) research. Energy instrumentation and analysis allows us to determine if the proposed protocol is better than the state-of-the-art. Various hardware-based energy instrumentation, simulation based study of energy footprint, and using radio activity as a proxy for energy has found widespread adoption in the community.

In this paper, we argue that despite the long history of energy instrumentation in WSN, we have not fully understood the implications of energy instrumentation in WSNs. Other communities have found that instrumentation of any type must be performed with care. Otherwise, there can be privacy and security implications. Existing work has indicated that power measurements can also act as side channel with the potential to compromise private information about the users [9]. We study these issues and implications in the context of sensor networks: could the energy instrumentation we collect in almost every sensor network deployment serve as side a channel to reveal unintended information?

We motivate the possibilities with one example from real-world devices. Monster powercontrol is a commercially available smart plug. We measure the power used by the plugs to reveal four properties of the system without any source code. First, we can tell the power state (On/Off) of plug outlet from the energy draw. Second, the current draw gives hints about the periodic communication between the plug and base station. Third, we can verify that the devices query the base station for new commands rather than the base station pushing messages to the devices: the periodic current crests continue even when we turn off the base station. Fourth, the current draw can also give hints about the wireless connectivity between the user devices and the base station.

We evaluate the design of our energy instrumentation and classification accuracy of the features based on energy data by doing extensive experiments on three WSN testbeds. Our results from analyzing four-million energy data and radio activity points, indicate that energy instrumentation and carefully designed features can not only reveal information about the network protocol but also some information about the application and the workload.

In this paper, we make three contributions:

- Design of classification features based on energy data with the goal of revealing protocol, network, and application information.
- Experimental evaluation of those features on three testbeds across multiple protocols, network topologies, and application workloads. We find that classification with those features can identify the routing protocol with more than 97% accuracy and application workloads with 85% accuracy.
- Demonstrate how we revealed the routing topology in the network, including next hop for each node, with just energy instrumentation, with 98% accuracy.

2 Related Work

In this section, we will give an overview of research related to energy measurement, profiling, and their applications in sensor networks and beyond.

Energy Instrumentation: Energy consumption is a significant concern in the design and development of WSN, hence, much progress and various measurement methods have been designed to measure the energy used by the nodes. FlockLab [8] has power meters attached to motes so the researchers could understand energy footprint of their protocols and applications. LEAP2 [11] provides unprecedented capabilities for directly observing energy usage for wireless sensor nodes in real-time, with microsecond-scale time resolution enables power profiling for each hardware subsystem. Researchers proposed a software based on-line code-level energy estimation model, the mechanism uses the current draw of each component during different period and aggregate them together to produce the total energy consumption [4].

Applications of Energy Measurement: While the primary reason for energy measurement is to understand the energy used by a sensor network system, researchers have found other use for energy data. Power Trace Testing is presented in [14], which designed a methodology to automatically investigate the

correctness of a WSN system by utilizing non-intrusive power measurement. In the testcase the system was able to detect an unexpected use of hardware component, which is not as scheduled. Dunkels et al. [3] use power state tracking to estimate the wireless network power consumption on network-level. Their approach even can break down the power consumption into individual activities on each node, enable the power profiling the pre-activity energy cost.
Revealing Privacy and Security Information: Researchers proposed a technique that use link-layer header data to infer network topology, de-anonymize servers present in anonymized network, to break their anonymization[10]. Researchers demonstrated even without priori-knowledge of household activities, it is still possible to extract complex usage patterns and privacy information from the household smart meter [9].

3 Features Design

In this section, we describe two novel features that we designed to reveal information about the network.
Radio Awake Length Counter (RALC): We define Radio Awake Length (RAL) as the total time that a node stayed in awake mode during each awake-sleep cycles. The RAL is not a fixed value, it depends on the packet size, the time before a node receives acknowledgment, etc. We used the threshold values 25ms, and half of the LPL settings 100ms to divide the RAL into three categories corresponding to a node only performing CCA check, receiving packets and transmitting packets, respectively. We name these three ranges as T_1, T_2 and T_3 as defined below, where T presents the RAL of each time:

$$T_1 : T \leq 25ms \tag{1}$$
$$T_2 : 25ms < T \leq 100ms \tag{2}$$
$$T_3 : 100ms < T \tag{3}$$

Within 10s disjoint window size, we count the amount of RAL in each of these ranges, and use these three counters are the feature, named Radio Awake Length Counter, i.e., *RALC = [m1, m2, m3]* , where m1 is the number of RALs that satisfy the predicate T_1. m2 and m3 are defined analogously. On Indriya and Twonet without energy meters, we measure RAL using software instrumentation.
Radio Awake Overlap Counter (RAOC): When a node successfully transmits a packet, the intersection of their radio awake time must not be empty. We count the times of two nodes have their radio awake time overlapped during a given period of time, and call it Radio Awake Overlap Counter. We use this feature to help us to infer the network topology in section 5.2.

4 Instrumentation Design

Protocols: A Collection Protocol is designed to reliably collect the data packets generated from every node in the network. In our experiments, we use Collection

Tree Protocol (CTP) [5] and MultiHopLQI (LQI) [13]. A Dissemination Protocol is designed to reliably deliver data packet from the base station to every node in the network. In our experiments, we use Drip [12] and DHV [1].

Testbed and Motes: We instrument the power uses and radio chips activities on three testbeds. FlockLab provides high-resolution power measurement profiling and precise time synchronization on 30 nodes. Indriya [2] has over 100 wireless sensor nodes. Twonet [7] is a testbed with 100 dual-radio nodes, which can operate in 2.4 GHz and 900 MHz. We set the Twonet nodes to run on 900 MHz to verify that our proposed approach works with 900 MHz as well. We use TinyOS for our experiments.

Low Power Listening (LPL): When using LPL, the node wakes up periodically to perform Clear Channel Assessment (CCA) to save energy. The node stays awake until the packet is received if it detects any preamble on the wireless medium. Otherwise it turns off its radio and switches back to sleep mode to save energy. In this study, we set LPL sleep interval to 200ms.

Experiment Configurations: Each experiment runs for an hour. Though it is impossible to ensure exactly the same workload across collection and dissemination protocols, we tried out best to make the workload similar across protocols by matching the packet sending interval, using the similar payload size with same sink node for all of the four experiments in each set.

Classifier Training: We use a 10s disjoint window to extract the RALC. Hence, for an one hour experiment, we have 360 feature vectors. In each set, four experiments generate 1440 feature vectors. We test four classifiers, J48, Logistic, Bagging and NaiveBayes. These are implemented in Weka [6], which has a collection of machine learning algorithms for data mining tasks. We also perform 10-fold cross-validation.

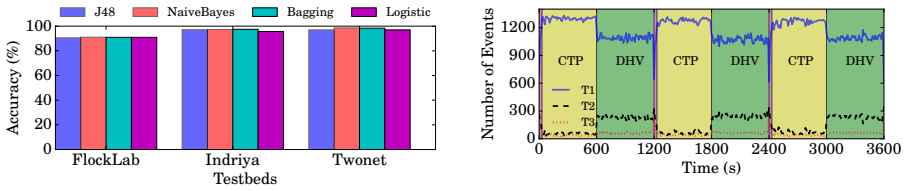
5 Experiment Settings and Evaluations

In this section we describe results on how accurately we are able to infer network and application aspects using features derived from energy instrumentation.

5.1 Identify Routing Protocols

Classify Network Protocols: We plot the classification accuracy results by using RALC across three testbeds in figure 1(a). The first group in figure shows all of four algorithms on FlockLab can achieve similar accuracy, and the average accuracy to classify the network protocol from RALC is more than 90%. The 2nd and 3rd group in figure 1(a) show the classification accuracy of using software measured RALC on Indriya and Twonet, where the average accuracy above 97% and 98%, respectively. This experiment show two highlights of RALC: It gives a robust results over the four classifiers. It generates a stable accuracy results over three testbeds, with different network layout and different radio bands.

To test the performance of RALC with external Wi-Fi interference, we repeat same experiments using two different channels, which one is overlapped with Wi-Fi, the other one is not. The results show the feature RALC can tolerant Wi-Fi



(a) Classification accuracy using RALC across three testbeds.

(b) Unsupervised clustering of two alternative network protocols.

Fig. 1: comparison of accuracy by using different features over 4 algorithms.

interference and achieve similar classifier accuracy. We also evaluate RALC by training it on sample sets from one experiment configuration and then testing it on data from another experiment configuration. We have 7 such cases and RALC's accuracy was between 82-97%.

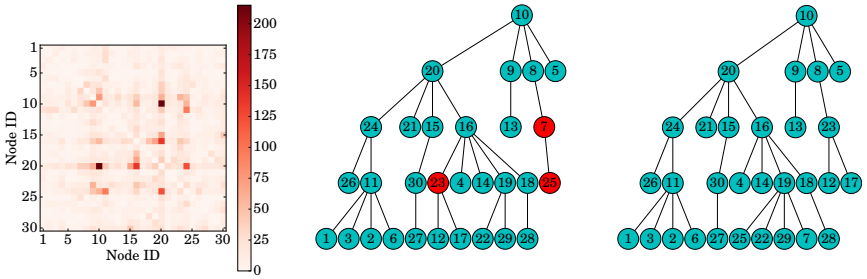
The reason why RALC gives a high classification results is because it can capture the unique patterns between the protocols. The control messages of each protocol is designed uniquely, e.g. the time interval between transmit control packets and number of control packets. While the workloads from the application layer are the same, using RALC makes it feasible to distinguish the protocols by looking at the patterns in radio activities triggered by transmit and receive packets, including data packets and control packets.

Cluster Analysis of Alternating Protocols: Next, we evaluate the effectiveness of RALC for clustering two protocols running during different periods. We switch back and forth between CTP and DHV protocols during a one-hour experiment. We use a general non-parametric cluster algorithm, MeanShift to cluster the RALC from the measurements. In figure 1(b), yellow and green backgrounds show the periods with correct clustering, while red shows the mis-clustered period. Out of 360 snapshots, only 18 of them were mis-clustered; thus, the percentage of correctly clustered snapshots is 95%. All of mis-clustered periods happen right after CTP starts. During the warm up period of CTP, the nodes exchange a lot of control packets to setup routing paths compared to the stable period. This causes the algorithm to mis-cluster CTP as DHV. This experiment shows that RALC can correctly identify the protocols running during different periods, and can also detect the moment when the protocol changes. Because RALC can capture the change in control overhead caused by a protocol switch. Hence, we expect our proposed approach can also cluster three or more protocols.

5.2 Infer Network Topology

Next, we study the effectiveness of RAOC in revealing information about the network topology and routing path for each node.

Parent Node and Routing Path: The RAOC across two nodes can be used to estimate the parent-child relationship across the network running multi-hop collection protocol . We remove the radio overlap length that are too short



(a) Heatmap of RAOC (b) Inferred Topology (c) Actual Topology

Fig. 2: Use RAOC to find the parent node for each node, and reveal the topology of whole network. Red circles indicate the node with wrong parent node.

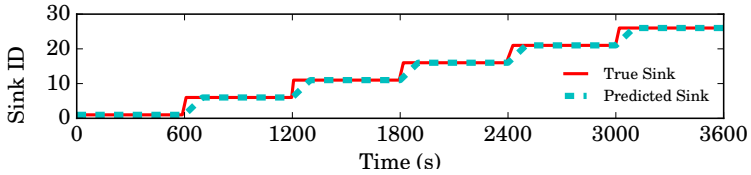
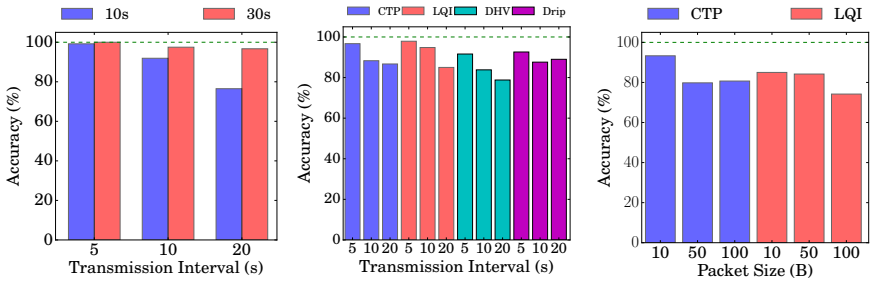


Fig. 3: The detected sink node by only using RALC compare to the ground truth.

(less than 0.025s) to focus on significant overlaps in our analysis. Figure 2(a) shows the heatmap of RAOC across every node-pair during a 200s window size. The darker color represents those two nodes having larger overlapped times. A heuristic to find the parent for a node is to simply designate the node with which a node has the largest overlap as its parent. For example, for y=20, the pixel at x=10 is darkest. Hence, we guess that node 10 is the parent of node 20. If multiple nodes have same overlap length, we use overlap information from adjacent time window. We use this heuristic for each node in the heatmap and construct the routing topology, which is shown in figure 2(b). We found that this inferred topology based on the heatmap is surprisingly close to the actual routing topology shown in figure 2(c). Only the nodes marked red had the wrong estimate of routing parent. We ran CTP and LQI multiple times on testbed and used the heuristics above to estimate the routing topology. The estimation accuracy across the experiments was 97.8% and 90.2% for CTP and LQI protocol, respectively.

Sink Node: Next we study how to identify the sink with RALC. During each 100s window period, the nodes with the maximum number of T_2 had the highest possibility to be the sink node, since the T_2 could reflect the number of receive events. We ran CTP for one hour, where the sink node changed every 600s. The red curve in figure 3 shows the true sink node ID while the blue curve shows the predicted sink node ID. The result shows that identifying the sink using RALC is accurate and feasible. The slight lag between predicted and actual sink is due to the 100s window when we calculate RALC.



(a) Transmission interval 5s, 10s, 20s on CTP.

(b) 12 combinations of intervals and protocols.

(c) 6 combinations of packet size and protocols.

Fig. 4: True Positive Rate when using RALC to detect application workloads.

5.3 Determine Application Workloads

In this section, we evaluate the effectiveness of RALC to distinguish different application workloads, including application layer packet transmission interval and payload size. In this section all of the experiments were run on FlockLab. We used J48 algorithm to perform the classification test.

Packet Transmission Interval on Same Protocol: We first run CTP with data being generated at three different intervals: 5s, 10s, and 20s. We calculate the True Positive Rate (TPR) when classifying each interval from the mixed dataset. We use 10s as the window size to calculate the RALC, and its corresponding TPRs are plotted with blue color in figure 4(a). The TPR is 99.2% to classify 5s interval, then TPR decreases to 91.9%, even drops to 76.5% for interval 10s and 20s, respectively. The drop is due to the packet transmission interval becoming larger than the RALC window. Thus, TPR increases with a larger RALC window size (30s), significantly improved the classification accuracy, as showed in the same figure with red color.

Packet Transmission Interval over Various Protocols: In Figure 4(b), we plot the results from determining packet intervals across four protocols using 30s window size. The average accuracy to classify one of the instance from all of the 12 combinations of 4 protocol and 3 intervals is 87.5%.

Packet Size: We vary the data packet size sent with CTP and LQI from 10 to 50 to 100 bytes. The dataset includes a total 6 distinct types of instances, which are the combination of two protocols and three packet sizes. Figure 4(c) shows the average accuracy to classify one instance from 6 combinations is 82.8%.

6 Conclusions

In this paper, we demonstrated that energy instrumentation can be a powerful tool to study and reveal information about the network, protocol, or workload. We designed features for classification and analysis based on energy instrumentation. We found that the feature called Radio Awake Length Counter is especially

versatile in revealing information across protocols and application workloads, such as 97% accurate for classify protocols, and average 87.5% accurate for classify workloads. Furthermore, another feature named Radio Awake Overlapped Counter could reveal the parent node for each node, even to disclose the actual network topology with 98% accuracy. Our extensive experimental results performed on three different testbeds over 100 test cases suggest that our proposed features are robust across the testbeds, frequency bands.

Acknowledgments

This work was partially supported by the National Science Foundation under grant no. IIS-1111507.

References

1. Dang, T., Bulusu, N., Feng, W.C., Park, S.: Dhv: A code consistency maintenance protocol for multi-hop wireless sensor networks. In: Wireless Sensor Networks, pp. 327–342 (2009)
2. Doddavenkatappa, M., Chan, M.C., Ananda, A.L.: Indriya: A low-cost, 3d wireless sensor network testbed. In: Testbeds and Research Infrastructure. Development of Networks and Communities, pp. 302–316. Springer (2012)
3. Dunkels, A., Eriksson, J., Finne, N., Tsiftes, N.: Powertrace: Network-level power profiling for low-power wireless networks (2011)
4. Dunkels, A., Osterlind, F., Tsiftes, N., He, Z.: Software-based on-line energy estimation for sensor nodes. In: EmNets. pp. 28–32 (2007)
5. Gnawali, O., Fonseca, R., Jamieson, K., Moss, D., Levis, P.: Collection tree protocol. In: ACM SenSys (2009)
6. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. ACM SIGKDD Explorations Newsletter 11(1), 10–18 (2009)
7. Li, Q., Han, D., Gnawali, O., Sommer, P., Kusy, B.: Twonet: Large-scale wireless sensor network testbed with dual-radio nodes. In: ACM SenSys (2013)
8. Lim, R., Ferrari, F., Zimmerling, M., Walser, C., Sommer, P., Beutel, J.: Flocklab: A testbed for distributed, synchronized tracing and profiling of wireless embedded systems. In: ACM/IEEE IPSN (2013)
9. Molina-Markham, A., Shenoy, P., Fu, K., Cecchet, E., Irwin, D.: Private memoirs of a smart meter. In: ACM BuildSys (2010)
10. Pang, R., Allman, M., Paxson, V., Lee, J.: The devil and packet trace anonymization. ACM SIGCOMM Computer Communication Review 36(1), 29–38 (2006)
11. Stathopoulos, T., Mclntire, D., Kaiser, W.J.: The energy endoscope: Real-time detailed energy accounting for wireless sensor nodes. In: ACM/IEEE IPSN (2008)
12. TinyOS: The drip protocol. `https://github.com/tinyos/tinyos-main/tree/master/tos/lib/net/drip`
13. TinyOS: Multihoplqi collection protocol. `https://github.com/tinyos/tinyos-main/tree/master/tos/lib/net/lqi`
14. Woehrle, M., Lampka, K., Thiele, L.: Exploiting timed automata for conformance testing of power measurements. In: Formal Modeling and Analysis of Timed Systems, pp. 275–290. Springer (2009)