Noname manuscript No.
(will be inserted by the editor)

Jun Han · Abhishek Shah · Mark Luk · Adrian Perrig

Don't Sweat Your Privacy

Using Humidity to Detect Human Presence

Received: June 8, 2007 / Accepted: July 2, 2007

Abstract Sensor nodes are increasingly deployed in many environments. Most of these nodes feature onboard sensor chips to measure environmental data such as humidity, temperature and light. In this paper, we show that seemingly innocuous and non-sensitive data such as humidity measurements can disclose private information such as human presence. We conduct several experiments using Telos motes running TinyOS to justify our claims. research to investigate mechanisms to prevent the leakage of private information.

1 Introduction

Sensor networks are generally deployed to measure some characteristics about a particular environment of interest. The data they gather can then be analyzed to extract important information regarding the occurrence of events in that environment. Some well-known applications of sensor networks include surveillance of critical infrastructure, tracking of environmental pollutants, measurement of traffic flows, and climate sensing and control in office buildings and homes.

Sensor networks are tools for collecting information, and an adversary can gain access to sensitive information either by accessing stored sensor data or by querying or eavesdropping on the network. Since sensor networks communicate over a wireless medium, even a remote adversary can eavesdrop and gain access to the data collected by the network. The need for privacy of data is evident in applications where sensor networks are deployed to collect personally identifiable information, such as sensing the location of people in buildings for disaster preparedness. However, in some environments, an adversary can use seemingly innocuous data to derive sensitive information other than the data monitored. In this paper, we discuss one such instance of this problem. Specifically, we show how seemingly innocuous data such as humidity measurements can be used to determine human presence or absence in a room. We show this because humidity data is not considered to be privacy-sensitive today. Hence, to reduce cost, the sensor networks monitoring humidity data will likely to be unprotected, and the data collected throughout such system might be shared freely without regard to privacy concerns. The present work, however, overturns this conventional wisdom by demonstrating that humidity data, is in fact, privacy-sensitive, since it yields information about human presence. We conduct several experiments using Moteiv Telos motes running TinyOS and the results from these experiments justify our claims.

It may be argued that an adversary could collect such personal information directly through site surveillance. However, as prior work points out, the main privacy problem posed by sensor networks is not that they facilitate the collection of information that would otherwise be impossible, but that sensor networks aggravate the privacy problem by making important information easily available through remote access [6]. Hence, an adversary can gather information in a low-risk, anonymous manner without being physically present to maintain surveillance. As the results from the experiments in our paper indicate, given a room with a setup of sensor nodes that measure humidity, a remote adversary can determine human presence or absence in that room by *only* using the humidity readings from the sensor nodes deployed in that room.

We note that our system is not a substitute for a human activity/motion detector system. Rather, it serves as a demonstration for inferring privacy-sensitive personal information such as human presence by only using humidity mea-

This research was supported in part by CyLab at Carnegie Mellon under grant DAAD19-02-1-0389 from the Army Research Office, and grant CCF-0424422 from the National Science Foundation, and a grant from Bosch. The views and conclusions contained here are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of ARO, Bosch, CMU, NSF, or the U.S. Government or any of its agencies.

Jun Han
E-mail: junhanece@gmail.com

Abhihek Shah
E-mail: abhishekjain.itbhu@gmail.com

Mark Luk
E-mail: mluk@ece.cmu.edu

Adrian Perrig
E-mail: adrian@ece.cmu.edu

surements. However, we envision that with further work, we would be able to improve our current prototype to deduce human activities such as speaking or drinking, and in some cases even breathing patterns.

2 System Description

Before we explain the details of our system, we first give a brief overview to summarize the main ideas in our approach. In our system, we deploy a sensor node in proximity to a user in a room. This sensor node performs humidity measurements and reports the readings to a data collection server. The humidity readings are then processed at the server, and based on the dynamics of the humidity data, we are able to detect human presence and absence.

Our system consists of the following three phases: (a) data acquisition, (b) data calibration, and (c) detection algorithm. We now proceed to the detailed description of each of these phases.

2.1 Data Acquisition

For our experiments, we use the Sensirion SHT15 [13] humidity sensor mounted on a Moteiv Telos mote[12,9] that is placed within a distance of one meter from the subject.

The Moteiv Telos is a popular mote architecture in the sensor network research community. It features the 8MHz TI MSP430 micro-controller, a 16-bit RISC processor with 10 Kbytes of SRAM, a 48Kbytes flash ROM, and a 12-bit Analog/Digital Converter with multiple input channels. It also carries a variety of sensors that include the Hamamatsu light sensors and Sensirion temperature and humidity sensors[1] . Telos motes run TinyOS, a real time operating system that is light weight and is specially designed for sensor nodes that have limited resources.

Sensirion SHT 15 is a high precision humidity sensor that uses the CMOS process and outputs digital values using its internal 12-bit A/D converter. It has a typical resolution of 0.03% Relative Humidity (RH) , and its humidity and temperature accuracies are ± 2.0 (%RH), and ± 0.3 (at 25° Celsius).

We use a small TinyOS application written in nesC (the programming language for TinyOS) to obtain the sensor readings and transmit them to the PC. The application samples humidity and temperature data every 500 milliseconds from the SHT15. The readings are then transferred to the UART, which is MSP430's universal synchronous/asynchronous receiver/transmitter (USART) set in an asynchronous mode. This allows us to transfer the data from the Telos mote to the server via a USB connection.

The data transferred from the Telos mote is read at the serial port on the server. To process this data, we implement a real-time analysis script written in MATLAB. When an event is triggered at the serial port, the script executes a callback function to process and graph the raw data in real time.

[1] Moteiv Telos motes carry a Sensirion SHT11 humidity sensor, which is slightly less accurate than SHT15. For our experiments, we replaced the onboard SHT11 sensor with SHT15 to get better results.

2.2 Data Calibration

In order to process the received data, it must first be calibrated to the standard units: Relative Humidity (*RH*) for humidity and degree Celsius for temperature. We use well-known standard techniques to perform data calibration for humidity and temperature [13]. For the sake of completeness, we briefly discuss them here.

We use Equation 1 to calibrate the raw temperature readings obtained from the sensor node.

$$C = D_1 + D_2 \cdot t \tag{1}$$

In the above equation, D_1 and D_2 are temperature conversion coefficients equivalent to -39.6 and 0.01 respectively, and t is the raw temperature reading from the sensor. To calibrate the raw humidity readings, we use Equation 2 given below.

$$RH = (C - 25) \cdot (T_1 + T_2 \cdot s) + h \tag{2}$$

In the above equation, C is the calibrated temperature in degrees Celsius, T_1 and T_2 are the temperature compensation coefficients equivalent to 0.01 and 0.00008 respectively, s is the raw humidity reading from the sensor, and h is the temperature-uncompensated humidity value given by:

$$h = K_1 + K_2 \cdot s + K_3 \cdot s^2 \tag{3}$$

where K_1, K_2, and K_3 are the humidity conversion coefficients equivalent to -4, 0.0405, and -2.8×10^{-6} respectively.

2.3 Detection Algorithm

Our detection algorithm determines human presence or absence based on the dynamics of the calibrated humidity data obtained from the previous phase. We now proceed to describe the detection algorithm. Later, we use results from an experiment to reason our methodology.

Algorithm

First, we apply a high pass filter to the calibrated humidity data obtained in the second phase of Data Calibration, which is equivalent to the first order discrete derivative of the input data. This will detect the changes in the original data. Next, we set a threshold value T over the filtered humidity data. Finally, we set a sliding window of size n for the data samples from the high pass filtered data. At any point of time, we evaluate the samples in the current sliding window to check if at least m of these samples exceed the threshold value T. If the check succeeds, then the system infers that a human is present. This decision holds true until some point of time,

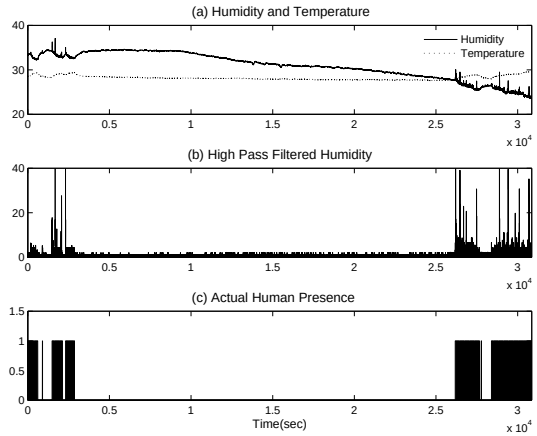

Fig. 1 Experiment *A*. (a) shows the Humidity and Temperature readings in RH and degree Celsius units respectively. (b) shows the resultant data when the high pass filter is applied over the humidity readings. (c) shows actual human presence.

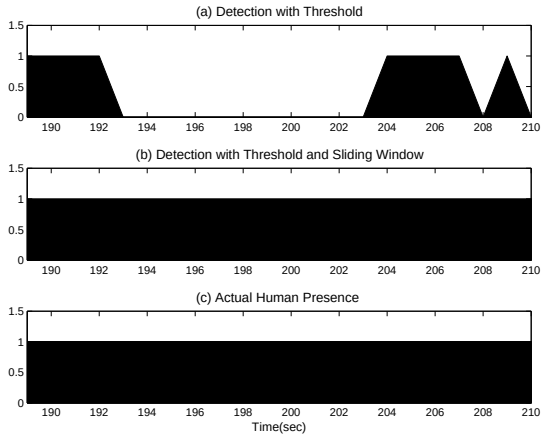

Fig. 2 These figures compare the decision on human presence (with respect to actual human presence) made by the system that uses a thresholding mechanism and one that uses a sliding window mechanism in addition to the threshold, during the interval $t = 189$ to $t = 210$ in experiment A

when the above check fails. At this point, the system decides that the human is absent.

Defintion 1 *Event E_1 is the case where the system is able to successfully detect human presence for a given sample when the human is present. We define* ***Detection Rate*** *as the ratio of the total number of samples when event E_1 occurs to the total number of samples when the subject is present.*

Defintion 2 *Event E_2 is the case where the system detects human presence for a given sample when the human is absent. We define* ***False Positive Rate*** *as the ratio of the total number of samples when event E_2 occurs to the total number of samples when the subject is absent.*

Discussion

Our methodology is guided by experimental results. To provide a better understanding of the reasoning behind our detection algorithm, we consider an example experiment. We conducted an experiment for a period of over eight hours. The subject in the experiment was a male. The total number of data samples obtained during the course of the experiment were 30855. Of these, there were 5756 samples when the subject was present and 25098 samples when the subject was absent. From now on, we shall refer to this experiment as *A*.

Figure 1(a) illustrates the calibrated humidity and temperature data obtained during experiment *A*. From inspection, the changes in humidity represented by jitters in Figure /reffig:one(a) directly correlate to human presence illustrated in Figure 1(c).

In light of the above observation, we apply a high pass filter to the calibrated humidity data. The resultant data is illustrated in Figure 1(b). Next, we set a threshold value T over the filtered humidity data. Ideally, we would want to achieve a high correlation between human presence and the filtered data.

One possible way to determine human presence is to check at any point of time whether the filtered humidity data exceeds the threshold T. Let us consider the time interval $t_1 = 189$ to $t_2 = 210$ during experiment *A*. Figure 2(a) illustrates the decision on human presence made by such a system during this time interval.

Observe that the system determines human absence in this time period, while the subject was actually present. Due to the occurrence of such events throughout the course of the experiment, the overall detection rate obtained by using such a methodology is considerably low. We have observed such occurrences when the subject is either temporarily idle or further away from the sensor node. Specifically, we found that the detection rate for this system in experiment *A* is 42.74%, when the threshold T was set to 1.3. We defer until later the explanation of the choice of the threshold value.

We solve the above problem by using the sliding window mechanism described earlier. Figure 2(b) illustrates the decision on human presence made by this system during the time interval $[t_1, t_2]$ in experiment *A*. Observe that the decision directly correlates to actual human presence ash illustrated in Figure 2(c).

3 Experimental Results

We conducted various experiments for different time-periods on different subjects. For each experiment, we placed a sensor node beneath the desk of the subject. In order to determine actual human presence, We preferr this simple method since it is the least intrusive way of obtaining actual presence for a given subject.

For each experiment, we analyzed the collected data by varying the system parameters – T, n and m in order to obtain the best trade-off between the Detection Rate and the False Positive Rate. Specifically, we chose the system parameters in the following manner: (a) $T \in [1.0, 1.3]$, (b) $n \in$

Experiment	Detection Rate	False Positive Rate
A	95.59	2.67
B	92.25	28.33
C	91.80	29.70

Table 1 Final Results. This table summarizes the results from three different experiments.

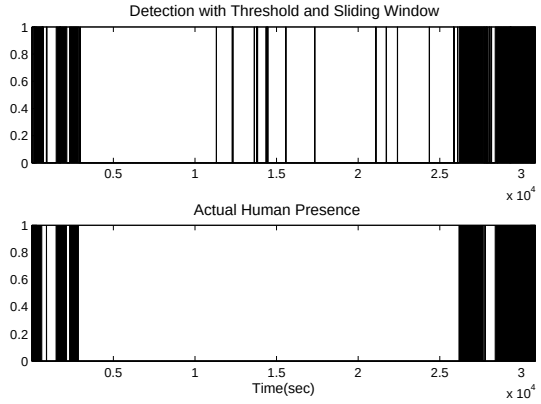

Fig. 3 Final Decision for Experiment *A*. These figures compare the final decision on human presence and absence made by our system with actual human presence for experiment *A*.

$[10,30]$, and (c) $m \in [1,5]$. In general,we found that $T \geq 1.0$ was a good choice to remove the noise in the filtered humidity data. Similarly, when *n* and *m* were chosen from the above value sets, better results were obtained. We now proceed to the description of the experiments and the obtained results. Later, we discuss how the system parameters influence the Detection Rate and the False Positive Rate.

We restrict our discussion to a few experiments to conserve limited space. Table 1 summarizes the results from three different experiments. Recall experiment *A* (c.f. Section 2.3) that was carried out on a male subject for a period of over 8 hours. From a total of 30854 samples, the subject was present during 5756 samples while he was absent during the remaining 25098 samples. By varying the system parameters, we found that the Detection Rate varies from 47.78% to 99.58%, while the False Positive Rate varies from 0.01% to 98.22%. We achieve a reasonable trade-off with 95.59% Detection Rate and 2.67% False Positive Rate, when the system parameters T, *n*, and *m* are set to 1.2, 30 and 3 respectively. Fig 3 compares the decision on human presence and absence made by the system relative to actual human presence for experiment *A*.

Next, we conducted experiment *B* on a second male subject for a period of over 14 hours. From a total of 51337 samples, the subject was present during 7663 samples and absent during the remaining 43674 samples. Again, we varied the system parameters while evaluating the results. When T, *n*, and *m* are set to 1.2, 30, and 1 respectively, we achieve a trade-off with 92.25% Detection Rate and 28.33% False Positive Rate.

Finally, we conducted experiment *C* on a female subject for a period of over 5 hours. From a total of 19566 samples, the subject was present during 13516 samples and absent during the remaining 6050 samples. When the system parameters are set to 1.2, 10, and 3 respectively, we obtain a trade-off with 91.80% and 29.70% as the Detection Rate and False Positive Rate respectively.

We note that the False Positive Rate is relatively high in Experiment B. This is primarily a result of the subject not being careful with recording actual human presence. We observed that the subject was around his desk many times while he actually recorded that he was away from his desk, thus appearing as present to the system, while the "actual human presence" was set to absent. This can be easily verified by changing the system parameters, T, *n* and *m* to 1.2, 30, and 4, respectively. The Detection Rate and the False Positive Rate reduce to 50.99% and 3.14%. Similarly, we note that the False Positive Rate is relatively high in experiment C. After analyzing the experimental data, we conclude that this is due to the fact that the number of samples when the subject was absent is relatively low – 6050 in total, therefore the resultant ratio is somewhat misleading. This can be compared to the number of samples the subjects were absent in experiments A and B, which were 43,674 and 25,098 samples respectively. Hence, even a low number of incorrectly asserted samples during the period when subject C was absent would lead to a high False Positive Rate.

Remark 1 We note that the sliding window mechanism can enable the system to use a lower threshold value T in order to capture more humidity changes, thus increasing the detection rate, while maintaining the number of false positives to a reasonably low value.

System Parameters vs Detection Rate and False Positive Rate. We now discuss the influence of the system parameters on the Detection Rate and False Positive Rate for a given experiment.

Remark 2 We note that decreasing T and *m* and increasing *n* results in higher values of the Detection rate and False Positive Rate. Similarly, increasing T and *m* and decreasing *n* results in lower values of the Detection Rate and False Positive Rate.

We now consider each system parameter individually and discuss its influence on the Detection Rate and False Positive Rate.

1. ***T***. Given a $< n,m >$ value pair, both Detection Rate and False Positive Rate tend to decrease when the threshold T is increased. This is true because as the threshold increases, there will be a decrease in the number of samples that exceed the threshold, thus resulting in lower values of the Detection Rate and False Positive Rate.
2. ***n***. Given a $< T,m >$ value pair, both Detection Rate and False Positive Rate tend to increase when the sliding window size *n* is increased. This can be explained in the following manner. As the sliding window size increases, the probability that *m* samples within the sliding window exceed the threshold increases. This results in

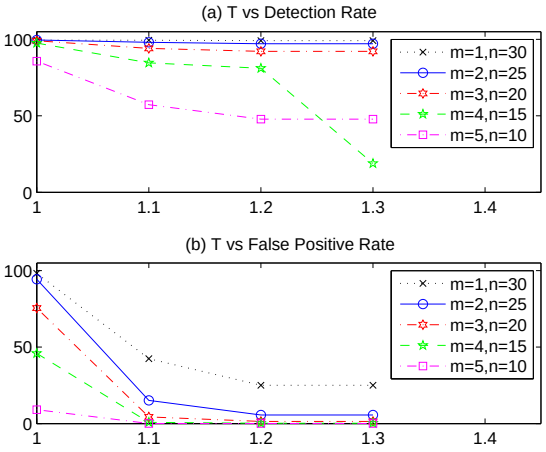

Fig. 4 ***T*** **vs Detection Rate and False Positive Rate.** These graphs show the variation of the Detection Rate and the False Positive Rate with respect to the threshold T. For simplicity of presentation, we restrict the graphs to specific values of the system parameters.

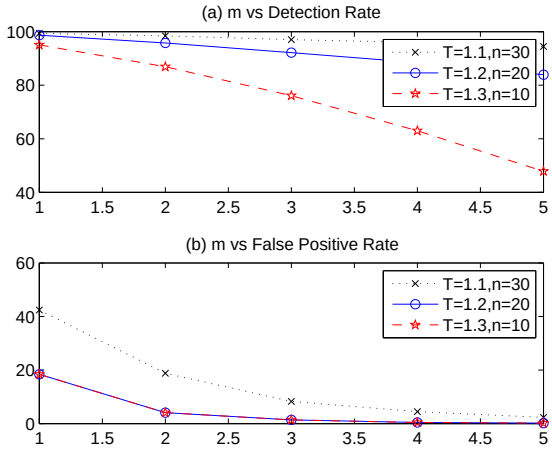

Fig. 6 ***m*** **vs Detection Rate and False Positive Rate.** These graphs show the variation of the Detection Rate and the False Positive Rate with respect to m. For simplicity of presentation, we restrict the graphs to specific values of the system parameters.

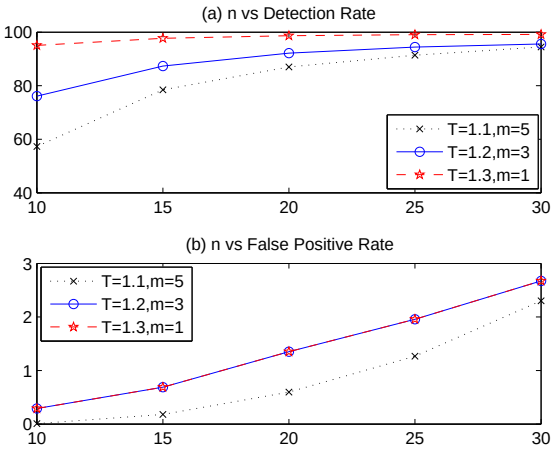

Fig. 5 ***n*** **vs Detection Rate and False Positive Rate.** These graphs show the variation of the Detection Rate and the False Positive Rate with respect to the sliding window size n. For simplicity of presentation, we restrict the graphs to specific values of the system parameters.

higher values of the Detection Rate and False Positive Rate.

3. ***m***. Given a $< T, n >$ value pair, both Detection Rate and False Positive Rate tend to decrease when m is increased. This is true because as m increases, the probability that m samples within a constant size sliding window exceed the threshold decreases, thus yielding lower values of the Detection Rate and False Positive Rate.

Figure 4, 5 and 6 illustrate the variation of the Detection Rate and False Positive Rate with the system parameters for experiment A.

Finally, we note that in our current prototype, we tune the system parameters manually to obtain a reasonable trade-off in the results. Ideally, we would want an optimal range for the system parameters to derive an upper bound on the results. In an advanced system, these parameters could be derived via a learning phase or on the basis of empirical data collected by a large number of experiments. Also, while conducting our experiments in real-time, we were able to notice sharp changes in the humidity measurements when the subject was engaged in activities such as talking or drinking a hot beverage. Furthermore, in some cases when the sensor node was within reasonable proximity to the subject, the humidity measurements were able to reflect the breathing pattern of the subject. We believe that with further work, it may be possible to improve our current prototype to deduce such human activities with reasonable accuracy.

4 Related Works

To the best of our knowledge, no prior work has been done to demonstrate the use of humidity measurements to determine human presence. Previous research on privacy issues in sensor networks has either concentrated on data confidentiality or transactional confidentiality. Data confidentiality implies secrecy of the messages being communicated in the network, while transactional confidentiality implies countermeasures against an adversary who conducts traffic analysis to gather additional information.

Data confidentiality in sensor networks has been extensively studied in the past. The authors in [11, 1, 5] provide a nice summary of available literature discussing this topic. The well-known way to achieve data confidentiality is data encryption. In order to perform encryption, the nodes in a sensor network utilize a key distribution protocol [4, 8, 3, 15] to establish keys.

The authors in [7, 10] discuss the transactional confidentiality associated with routing of messages within a sensor

network. These papers address the problem of an adversary determining source location by violating transactional confidentiality and discuss possible solutions to this problem using altered routing algorithms for sensor networks. In [14], refinements are made on the solutions discussed in [7, 10]. The problem of sink location information confidentiality is addressed in [2]. The authors discuss two main examples where transactional confidentiality can be breached in a sensor network: message routing and message sending rate. Countermeasures against an adversary who can perform traffic analysis to gather such information are given.

5 Conclusion and Future Work

We show that seemingly innocuous and non-sensitive data such as humidity measurements can disclose private information such as human presence or absence. We conduct several experiments using Telos motes running TinyOS and the results from these experiments justify our claims.

As discussed earlier, we were able to notice clear correlation between the changes in humidity measurements with the activities that the subject was engaged in, such as talking, drinking hot beverages, and breathing patterns in some cases. With further research, we hope to improve our current prototype to be able to determine these various activities with reasonable accuracy.

We note that it is worthwhile to investigate potential areas where our system could have a positive impact. Given that sensor nodes capable of humidity measurements are very inexpensive and widely used, it may be feasible to use our system for applications such as automation of climate control in buildings, and infant (or patient) monitoring.

References

1. C.Karlof and D.Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. In *IEEE International Symposium on Advances in Wireless Communications*, 2002.
2. J. Deng, R. Han, and S. Mishra. Decorrelating wireless sensor network traffic to inhibit traffic analysis attacks. In *Elsevier Pervasive and Mobile Computing Journal, Special Issue on Security in Wireless Mobile Computing Systems*, 2006.
3. W. Du, J. Deng, Y. Han, and P. Varshney. A pairwise key predistribution scheme for wireless sensor networks. In *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, pages 42–51, October 2003.
4. L. Eschenauer and V. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of Conference on Computer and Communication Security*, pages 41–47, November 2002.
5. E.Shi and A.Perrig. Designing secure sensor networks. *IEEE Wireless Communications*, 2004.
6. H.Chan and A.Perrig. Security and privacy in sensor networks. *IEEE Computer Magazine*, October 2003.
7. P. Kamat, Y. Zhang, and W. Trappe. Enhancing source-location privacy in sensor network routing. In *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems*, 2005.
8. D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, pages 52–61, October 2003.
9. Moteiv Corp. *Telos (Rev B) Datasheet, http://www.moteiv.com*, Dec 2004.
10. C. Ozturk, Y. Zhang, and W. Trappe. Source-location privacy in energy-constrained sensor network routing. In *Proceedings of the 2nd ACM workshop on Security of Ad hoc and Sensor Networks*, 2004.
11. A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. In *Proceedings of Conference on Mobile Computing and Networks (MobiCom)*, July 2001.
12. J. Polastre, R. Szewczyk, and D. Culler. Telos: Enabling ultra-low power wireless research. In *Special track on Platform Tools and Design Methods for Network Embedded Sensors (SPOTS), Fourth International Symposium on Information Processing in Sensor Networks (IPSN'05)*, April 2005.
13. Sensirion. *High precision humidity sensor SHT15 Datasheet, http://www.sensirion.com*.
14. Y. Xi, L. Schwiebert, and W. Shi. Preserving source location privacy in monitoring-based wireless sensor networks. In *Proceedings of the 2nd International Workshop on Security in Systems and Networks*, 2006.
15. S. Zhu, S. Setia, and S. Jajodia. LEAP: Efficient security mechanisms for large-scale distributed sensor networks. In *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, October 2003.