Misinformation Theory: Foundations and Applications

Giacomino Veltri
Computer Science Department
University of California, Los Angeles
gveltri@cs.ucla.edu

Miodrag Potkonjak
Computer Science Department
University of California, Los Angeles
miodrag@cs.ucla.edu

I. Abstract

Information theory had a broad, deep, and profound impact both in science and engineering. It was based on a number of new fundamental concepts. Of course, the proposed misinformation theory does not have those types of new and revolutionary concepts. In a sense, it is unlikely that any future theory related to semantic processing may have the simplicity and elegance of the theory and techniques related to syntax-only processing. However, we believe that misinformation theory can provide sound foundations and concepts to address issues related to the integrity of obtained and employed information and that it will have a broad spectrum of important applications. The theory can be generalized to address under-constrained problems by requiring that after the misinformation, none of the initial constraints be violated while the solution space is skewed to the maximal extent in accordance with proper misinformation metrics.

While the theory is generic and can be applied to many domains (e.g. information posted on the Internet and in social networks), we demonstrate it in the context of wireless embedded sensor networks. The typical question that we address is what value a node should report in such a way that will maximally skew the conclusion of the user while not being detected as an attacker and not crossing a threshold where the user starts to suspect the obtained results.

The misinformation problem is formulated using the canonical form for stating optimization problems (consisting of an objective function and constraints) and solved under a variety of assumptions about how the attacking nodes are detected, when the result is accepted as correct, protocols for information exchange, and knowledge of attacking sources. We begin by analyzing the simple scenario of single value reading and then conduct extensive simulations for trilateration and show under which conditions which type of misinforming is most effective.

II. Introduction

The birth and growth of the Internet is often considered the technological revolution. Furthermore, modern times are referred to as the *information age*. Titles like this help stress the importance of information in modern society. Having accurate information is critical in many circumstances. For example, in the stock market, one can only hope to thrive if one has accurate information about the businesses and companies being invested in. Inaccurate information about businesses and companies can yield significant financial losses. The scientific community also requires accurate information; inaccurate measurements and results can yield worthless data, which in turn wastes not only money but time as well.

The importance placed on accurate information helps motivate how important it is to study misinformation theory; only when one knows the means of misinforming can one adequately formulate a strategy for defense. Thus, these are the goals of our paper: to determine the foundations of a universal and systematic way in which one can deliver inaccurate information and to identify key ways in which one can defend against receiving inaccurate information.

It is important to note that in our examples and methods, we assume that only one instance or

scenario is examined, thus making the only likely attacker the most discrepant value. Note that without loss of generality, one can adopt other metrics for the detection of the source of misinformation and when the information can be trusted. For example, one can take into account the geographical location of a sensor and the consequences of that positioning on the expected level of discrepancy.

II.A. Motivational Examples

Probably the best way to illustrate the key mechanisms, tradeoffs, and potential impact of misinformation theory is to use a conceptually simple yet illustrative example. The simplest possible example is when a user combines measurements from three different sensors in order to conclude the reading as accurate as possible. Let us denote the sources of information (sensors in a sensor network) by *A*, *B*, and *C*. Further assume that each node is sensing the same particular phenomenon and produces measurement values v_A, v_B, and v_C respectively. The final result, v_F, will be the average of these three sensor values. Finally, the attacking node is determined as the node whose readings is the furthest from the calculated average. Therefore, the equation to solve for v_F is:

$$v_A + v_B + v_C = 3v_F \quad (1)$$

The equation to determine the attacking node therefore can be written as:

$$\max\{|v_F - v_A|, |v_F - v_B|, |v_F - v_C|\} \quad (2)$$

Now, assume that *C* is the *attacking node*; i.e. *C* is the node that tries to skew its reported value in such a way to maximally affect the resulting calculation done by the user. We denote this skewed value by v_C' and the skewed final result by v_F'. Furthermore, assume that the attacking node is the last one to report its value. In addition, the attacking node is aware of the calculations used to determine the final result and to figure out if any nodes are not reporting accurately. Finally, the goal for the attacking node is to maximize the difference between the non-skewed result and the skewed result *without being detected*. The whole scenario can be abstracted using a set of equations and inequalities. Specifically, after data skewing by node *C*, the skewed result is given by the following formula:

$$v_A + v_B + v_C' = 3v_F' \quad (3)$$

To avoid detection, the attacking node must also adhere to the following constraint:

$$|v_F' - v_C'| \le \max\{|v_F' - v_A|, |v_F' - v_B|\} \quad (4)$$

Finally, the goal of the attacker can be summarized by the following objective function:

$$\max\{|v_F' - v_F|\} \quad (5)$$

From a mathematical perspective, the solution to this system of equations is unintuitive. However, if one approaches the problem from a different perspective, one can derive a solution that satisfies the above-mentioned constraints. First, assume that $v_A \le v_B$. Now, there exists two possible scenarios: v_C is closer to v_A or v_C is closer to v_B. If v_C is closer to v_A, the attacker wants to report a v_C' such that the resulting average is closer to v_B (whereas originally the resulting average was closer to v_A). Thus, the attacker would report $v_C' = 2v_B - v_A$, making v_F' be v_B. However, if v_C is closer to v_B, the resulting average is closer to v_B, and thus *C* would report a value such that the skewed average is closer to v_A. To do this, *C* would report $v_C' = 2v_A - v_B$, making v_F' be v_A. One might ask if it is possible to do better than these values, and in this example, the answer is no. The above equations guarantee that the node whose value lies between the other two is exactly the midpoint of the other two values, and thus, neither point is more likely to be the attacker. If one skews v_C further by some amount δ, the average is only skewed by $\delta/3$, and thus, skewing v_C further results in v_C being determined as the attacker, violating constraint (4).

Now, to demonstrate the potential damage an attacking node can have, we will solve this system of equations using actual numbers. Assume that $v_A = 2.0$, $v_B = 2.5$, and that $v_C = 1.5$. Clearly, one can see that $v_F = 2.0$. However, what happens to v_F' if v_C' is carefully chosen? Using the system of constraints derived for the attacker and noting the above-mentioned solutions, one can

conclude that $v_C' = 3.0^*$ and that the resulting $v_F' = 2.5$. Thus, in this simple example, one can see that the difference between the actual result (2.0) and the skewed result (2.5) is quite significant, and therefore one must pay attention to attackers can pose significant threats.

The next motivational example we present is more complicated than the simple average example presented earlier. This problem deals with the idea of navigation through a sensor network. Navigation makes use of not only Euclidean equations but also the laws of physics, thus making the system of constraints more difficult. Note that although this problem contains an element of time, only one measurement is being made and thus does not violate our assumptions that we examine only one scenario.

It is our goal to develop a general and systematic way of skewing data in a sensor network, and thus, we demonstrate the wider applicability of our general model of skewing data to the problem of navigation in a sensor network. Although similar to atomic trilateration, navigation contains many different elements and aspects that make the problem more complex and interesting to study from a data skewing perspective.

The problem of navigation in a sensor network is described as follows. A node whose location is unknown travels through a sensor network and encounters different sensors at different times. These sensors, like the sensors in atomic trilateration, are capable of measuring the distance between themselves and the target node. However, in order to increase measurement accuracy and extract more data from the network, the target node is further equipped with sensors to measure velocity and acceleration. It is important to note that although we focus on velocity and acceleration, the target node can contain any type of sensor that aids in increasing the accuracy of the measurements. Thus, using the distance measurements from different nodes throughout the course of travel through the sensor network, and the additional measurements from onboard sensors, the target node is able to calculate such attributes like position and trajectory.

To further clarify the problem of navigation, we now discuss in detail the variables that are used in the canonical formulation of our problem, which consists of an objective function and constraints. First, assume that there exists three sensors that are encountered at time $t = 0$: A, B, and C, with positions (x_A, y_A), (x_B, y_B), and (x_C, y_C) respectively. In addition, there exists two more sensors, D and E, which are encountered at time $t = ?t$ and have positions (x_D, y_D) and (x_E, y_E) respectively. Furthermore, assume that node E is the attacking node. The target node will have position (x_0, y_0) at time $t = 0$ and (x_1, y_1) at time $t = ?t$. Finally, the target node measures the velocity vector (v_{x0}, v_{y0}) at time $t = 0$ and (v_{x1}, v_{y1}) at $t = ?t$ and the acceleration vector (a_{x0}, a_{y0}) at time $t = 0$ and (a_{x0}, a_{y0}) at time $t = ?t$. Before deriving the constraints and objective function for the canonical form, we must also associate error variables denoted by $?$ to represent errors associated with particular measurements.

These variables are combined using a system of constraints and an objective function to determine the position of the target node. It is important to note that although the velocity and acceleration constraints are redundant, they are still critical because they limit the amount by which the attacker can skew the data. The system of constraints will contain two different types: the first type is due to Euclidean geometry and the second type is due to the physical laws of nature. The first set of constraints states that the position of the target node at time $t = 0$ is related to the distance measured to each node at that time:

$$\begin{aligned} D_A &? \sqrt{(x_A - x_0 - v_{x0}t - \tfrac{1}{2}a_{x0}t^2)^2 + (y_A - y_0 - v_{y0}t - \tfrac{1}{2}a_{y0}t^2)^2} \\ D_B &? \sqrt{(x_B - x_0 - v_{x0}t - \tfrac{1}{2}a_{x0}t^2)^2 + (y_B - y_0 - v_{y0}t - \tfrac{1}{2}a_{y0}t^2)^2} \\ D_C &? \sqrt{(x_C - x_0 - v_{x0}t - \tfrac{1}{2}a_{x0}t^2)^2 + (y_C - y_0 - v_{y0}t - \tfrac{1}{2}a_{y0}t^2)^2} \end{aligned} \quad (6)$$

* In actuality, the final result is not 3.0 due to the strict inequality of the detection constraint, but is in fact some number infinitesimally smaller. However, for purposes of this example, 3.0 will suffice.

Similarly, the next set of constraints relate the position of the target node at time $t = \Delta t$ to the distance measured to each node at time $t = \Delta t$:

$$D_D \approx \sqrt{(x_D - x_1 - \Delta_{Dx1})^2 + (y_D - y_1 - \Delta_{Dy1})^2}$$
$$D_E \approx \sqrt{(x_E - x_1 - \Delta_{Ex1})^2 + (y_E - y_1 - \Delta_{Ey1})^2} \quad (7)$$

The last set of constraints relates the position at time $t = \Delta t$ to the position at time $t = 0$ by using classical physics:

$$x_1 = x_0 + (v_{x0} + \Delta_{vx0})\Delta t + \frac{1}{2}(a_{x0} + \Delta_{ax0})\Delta t^2$$
$$y_1 = y_0 + (v_{y0} + \Delta_{vy0})\Delta t + \frac{1}{2}(a_{y0} + \Delta_{ay0})\Delta t^2 \quad (8)$$

Finally, because multiple solutions exist to this problem, each solution must be evaluated according to some particular objective function. Typically, this function will be to minimize some function of the discrepancy variables.

$$Min\ f(\Delta_{Ax0}, \Delta_{Ay0}, \Delta_{Bx0}, \Delta_{By0}, \Delta_{Cx0}, \Delta_{Cy0}, \Delta_{Dx1}, \Delta_{Dy1}, \Delta_{Ex1}, \Delta_{Ey1}, \Delta_{vx0}, \Delta_{vy0}, \Delta_{ax0}, \Delta_{ax1}) \quad (9)$$

From this system of constraints and an objective function, we are able to calculate the value of (x_1, y_1), and can use this value to maximally skew the result. To formulate the skewed result, we follow the generic algorithm outlined above, with the first step being to calculate the non-skewed result. The second step is to calculate the skewed result by replacing the variables and values reported by the attacker and replacing them:

$$D_D \approx \sqrt{(x_D - x_1^* - \Delta^*_{Dx1})^2 + (y_D - y_1^* - \Delta^*_{Dy1})^2}$$
$$D_E^* \approx \sqrt{(x_E - x_1^* - \Delta^*_{Ex1})^2 + (y_E - y_1^* - \Delta^*_{Ey1})^2} \quad (10)$$
$$x_1^* = x_0 + (v_{x0} + \Delta_{vx0})\Delta t + \frac{1}{2}(a_{x0} + \Delta_{ax0})\Delta t^2$$
$$y_1^* = y_0 + (v_{y0} + \Delta_{vy0})\Delta t + \frac{1}{2}(a_{y0} + \Delta_{ay0})\Delta t^2$$

Finally, the goal of the attacker is to maximize the difference between the original solution (x_1, y_1) and the skewed solution (x_1^*, y_1^*). This can be done by maximizing the Euclidean distance between the two by using the objective function:

$$Max \sqrt{(x_1 - x_1^*)^2 + (y_1 - y_1^*)^2} \quad (11)$$

II.B. Objectives and Technical Contributions

Our goal is to study and develop misinformation theory – the theory by which data can be skewed and the means by which one can protect oneself from skewed data. In addition we hope to demonstrate the significance of misinformation theory through our simulation results. Finally, we present additional simulation results that demonstrate the effectiveness of different means of defense. Our main technological contributions are our generic algorithm for determining how to effectively skew data and our generic algorithm for protecting oneself from potentially skewed data.

II.C. Paper Organization

This paper is organized as follows. In Section III we present the means by which we will test and evaluate our algorithm. Section IV presents a generalized overview of misinformation theory. Section V discusses our method by which one can skew data and presents simulation results that demonstrate the effectiveness of our method. In Section VI we discuss the means by which one can protect oneself from skewed data and provide data showing how effective each method is. In Section VII we present related work and we conclude in Section VIII, which is followed by our references.

III. Preliminaries

In this section we discuss the means by which we will test and evaluate misinformation theory. Misinformation theory will be applied to sensor networks, and in particular, the problem of atomic trilateration [Sli02].

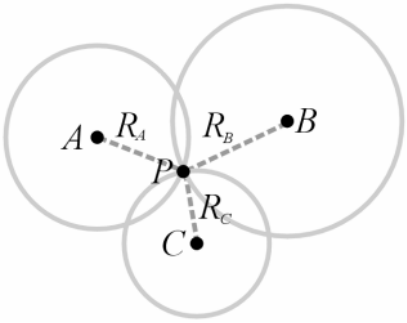

Figure 1: This figure demonstrates a geometric interpretation of atomic trilateration.

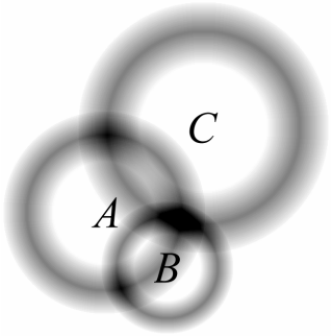

Figure 2: This figure demonstrates how errors can affect the problem of atomic trilateration.

III.A. Sensor Networks

We use sensor networks as a testing ground to demonstrate the applicability and accuracy of misinformation theory. Recently, sensor networks have been receiving a significant amount of attention from the research community. This new focus on sensor networks can be attributed to their revolutionary nature; sensor networks are revolutionary in that they represent a fundamental paradigm shift and provide a new means with which one can create exciting applications. One key aspect of sensor networks is its infrastructure. Unlike traditional networks (such as cellular phone networks), sensor networks often have no fixed infrastructure. By not having a fixed infrastructure, sensor networks are easier to deploy and can be adapted to particular situations much more rapidly. In addition, combining inexpensive yet power-efficient reliable sensors in a wireless ad-hoc network with limited computation and communication resources provides a new field of engineering research.

III.B. Atomic Trilateration

One key problem related to sensor networks is that of location discovery. Most significant applications of sensor networks assume that a node is aware of its own location; however, this is not a trivial matter. When dealing with sensor networks containing 100, 1000, or more nodes, manually inputting sensor location information becomes tedious and time consuming, and thus, an algorithmic approach to location discovery is needed.

The algorithmic approach to location discovery that we use throughout this paper is that of atomic trilateration. With respect to a two-dimensional sensor network, atomic trilateration is the means by which a sensor in a network can determine its position by using the positions of and distances to at least three other nodes of known location. From these positions and distances, a node that is trying to determine its location can create a system of equations. To further specify this problem, we assume that a node P is attempting to determine its actual location (x_F, y_F) by contacting nodes whose location is known, A, B, and C, with positions (x_A, y_A), (x_B, y_B), and (x_C, y_C), and distances (radii) R_A, R_B, and R_C respectively. With the information obtained from the other nodes in the network, the node P is able to determine the following system of constraints for atomic trilateration to determine the final result:

$$\begin{aligned} \sqrt{(x_A - x_F)^2 + (y_A - y_F)^2} &= R_A \\ \sqrt{(x_B - x_F)^2 + (y_B - y_F)^2} &= R_B \\ \sqrt{(x_C - x_F)^2 + (y_C - y_F)^2} &= R_C \end{aligned} \qquad (12)$$

When examining this system of nonlinear equations, it is difficult to determine whether or not a solution exists. However, if one looks at this system of equations from a geometric perspective, one can see that (in non-pathological cases) exactly one solution exists (see Figure 1).

In reality, there exist errors in measurement, and these errors can have a significant impact on the determination of the querying node's position. Our model of error in measurements with sensor networks is defined as having a Gaussian distribution with the standard

deviation increasing as the measured distance increases [Rap60]. Assume that $(?_{Fx}, ?_{Fy})$ is the error associated with the calculated result. Further assume that $?_A$, $?_B$, and $?_C$ are the errors associated with the measurements from nodes *A*, *B*, and *C* respectively. When applying these errors to the original atomic trilateration system of constraints, we receive the following set of constraints:

$$\left|\sqrt{(?x_A - ?x_F - ?_{Fx})^2 + (?y_A - ?y_F - ?_{Fy})^2} - R_A\right| \le ?_A$$
$$\left|\sqrt{(?x_B - ?x_F - ?_{Fx})^2 + (?y_B - ?y_F - ?_{Fy})^2} - R_B\right| \le ?_B \qquad (13)$$
$$\left|\sqrt{(?x_C - ?x_F - ?_{Fx})^2 + (?y_C - ?y_F - ?_{Fy})^2} - R_C\right| \le ?_C$$

Again, from this system of equations, it is difficult to visualize a solution. When analyzing this system of equations from a geometric approach, it becomes easier to visualize a solution. This can be seen in Figure 2. As one can see, in this scenario there exists not a single solution but an area within which the actual solution exists. In order to differentiate between and judge the possible solutions, it is necessary to evaluate each solution with respect to an objective function. We use the following objective function (which minimizes the sum of squares of the errors $?_A$, $?_B$, and $?_C$):

$$\min: \quad ?_A^2 + ?_B^2 + ?_C^2 \qquad (14)$$

It is possible to use an optimization algorithm to solve this system of constraints with respect to the objective function. For our purposes, we will use a grid-based optimization algorithm that works as follows. The first step of this algorithm is to determine the area in which the solution exists. Then, an $n \times n$ grid is superimposed over this area and each grid square is checked to see if it satisfies all constraints. If the grid square satisfies all constraints and is optimal with respect to the objective function, it is selected. We assume that if *n* is large enough, the entire grid square will either satisfy or not satisfy the constraint, and thus, only the midpoint of the grid square is checked. An $n \times n$ grid is superimposed over the selected grid square and the algorithm is repeated until some user-defined search limit is reached. The final selected grid square is reported to contain the solution.

IV. Foundations

Our main purpose in this paper is to demonstrate the importance of data skewing. In order to show its importance, we must first discuss the ways by which an attacker can skew data, the means by which the attacker can skew data, and the assumptions required to do so. In this section, we present the different methods for skewing data, how to achieve them, and what assumptions must be made.

Although there are potentially many ways to skew data, we have determined two that we find to be of most significance. The first (and simpler) means of skewing data is to exploit naturally occurring errors in the system. By doing this, the attacker hopes to skew the result yet still have its reported value appear statistically correct. The second (and more difficult) means of skewing data is to attempt to force the resulting calculation to an incorrect solution. This scenario can only be realized when the system itself contains multiple possible solutions. The attacker can then determine for itself the most likely solution and report a value such that a different solution results.

To skew data for a particular problem, we assume that the problem can be represented in the canonical form consisting of a set of constraints and an objective function. Writing a problem in this canonical form can be difficult depending on the nature of the particular problem. However, this step is crucial because once this step is completed, it is possible to use any optimization method to solve the problem. In Section III, We have demonstrated how the problem of trilateration in a wireless sensor network can be written in the canonical form.

Before we can discuss the mechanisms that we have developed for an attacker to skew data, we must first discuss the assumptions that we hold. These assumptions involve how the attacker wishes to skew data, which nodes are capable of receiving data, what data they can receive, and what prior knowledge about the existing system that an attacking node has. For all systems we analyze, we assume that all nodes are close enough to be able to communicate amongst each other. In addition, we assume that the goal of the attacker is to maximize a function of the original solution and skewed solution subject to the constraint that the calculating node must not suspect the attacking node's data as being faulty. The function to be maximized should be representative of some sort of "distance" between the original solution and the skewed solution. The remaining assumptions deal with what prior information the attacking node has about the system. For example, we assume that the attacking node is aware of the procedure used to determine the final result, that the attacking node is capable of receiving values reported by other nodes, that the attacker is aware of the characteristics (i.e. error probability and so forth) of other nodes, and finally, what the procedure is being used to detect attacking nodes. When determining what information is available to the attacking node, it is important to make a corresponding assumption – when the attacking node reports its value. If an attacker reports its value before other nodes, it must rely upon assumptions about other nodes' values in order to skew data.

With these assumptions, it is possible to derive a generic means of skewing data in the system. Because the attacker is aware of what procedure is being used to calculate the final result, the attacker can determine a system of constraints and an objective function that can produce the non-skewed result. With this system of equations, the attacker (after calculating the original solution) can modify this system of equations to utilize its skewed value, thus producing a skewed result. The attacker must also add a constraint to this system of equations that prevents its value from being detected as skewed. This is possible due to the assumption stating that the attacker is aware of what mechanism is being used to determine skewed values. Finally, the attacker must optimize this system of constraints according to an objective function. The generic objective function is to maximize the difference between the original and skewed solutions. A more formal algorithm describing this procedure can be seen in Figure 3.

Determine a system of constraints and an objective function for the original problem.

Solve this system using any optimization method, and let the resulting solution be S.

Modify the system of constraints from Step 1 to use the attacker's skewed reported value and produce a skewed result S'.

Add to the constraints determined in Step 3 the Detection Constraint.

Solve this new system of equations using any optimization method with the objective function maximize(f(S, S')), *where f is representative of the distance in the solution space between* S *and* S'.

Figure 3: Formal algorithm describing the procedure used by an attacker to maximally skew a computation without being detected.

We have also developed two means of defense against these types of data skewing. The first means of defense deals with the method of data skewing in which the attacker attempts to hide in naturally occurring errors in the system. To reduce data skewing from this method, one can over-constrain the problem. By adding additional constraints, the range in which a solution can exist will potentially diminish. With respect to atomic trilateration, this method of defense corresponds to querying additional sensors in the network.

The second means of defense attempts to prevent the attacker from being able to skew the final result towards a different (yet still valid) solution. The key behind doing this is to limit the amount of information the attacker has at its disposal. With respect to atomic trilateration in a sensor network, this method of defense corresponds to varying the order in which sensors are queried in the network.

V. Attack Protocols

In this section, we clarify two primary means by which an attacker can skew data by demonstrating how they can be applied to the problem of atomic trilateration in a wireless sensor network. The first means is to exploit the naturally occurring errors in a measurement from a sensor in the network. By doing so, the attacker hopes that its value will not appear statistically significant when compared to the final value calculated.

When a problem contains more than one possible solution, an attacker can have a more significant impact on the final value calculated by trying to manipulate it towards an alternate solution. By moving the final value calculated towards an alternate solution, the attacker not only has the potential to seriously impact the final value calculated but also is able to remain undetected because the final value calculated is a valid solution as well.

How does one determine which method is best in a particular scenario? This question is inherently answered by the canonical form in which we assume misinformation problems to be stated. If multiple solutions exist for a particular problem, solving the problem (stated in canonical form) with an appropriate optimization algorithm will automatically select a valid solution (due to the constraints listed in the canonical form) and it will be as skewed as possible (due to the objective function).

To promote further understanding of misinformation theory, we will demonstrate how the generic procedure of attacking derived in Section IV can be applied to the problem of atomic trilateration in a sensor network. Recall from Section III the canonical form for the problem of atomic trilateration (the system of constraints (13) and objective function (14)).

Now, assume that node C is the attacking node, and that it will report a distance (radius) of R_C' to produce a skewed result of (x_F', y_F'). From this system of constraints and an objective function to calculate the original, non-skewed result (x_F, y_F), the attacker is now able to calculate the non-skewed results and modify this system to produce a skewed result. The first step in doing so is to modify the

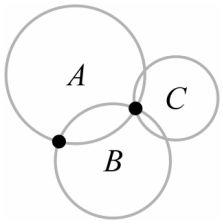

Figure 4: In this figure, the two dots represent the possible solutions before the attacking node (*C*) reports its value. As one can see, the attacking node's radius determines which of the two possible solutions is correct.

constraints to utilize the skewed reported value (R_C') instead of the non-skewed value (R_C) thus producing a skewed result (x_F', y_F'):

$$\begin{aligned}
&\left|\sqrt{(x_A - x_F' - \varepsilon_{Fx})^2 + (y_A - y_F' - \varepsilon_{Fy})^2} - R_A\right| \le \varepsilon_A \\
&\left|\sqrt{(x_B - x_F' - \varepsilon_{Fx})^2 + (y_B' - y_F - \varepsilon_{Fy})^2} - R_B\right| \le \varepsilon_B \\
&\left|\sqrt{(x_C - x_F' - \varepsilon_{Fx})^2 + (y_C - y_F' - \varepsilon_{Fy})^2} - R_C'\right| \le \varepsilon_C
\end{aligned} \tag{15}$$

At this point, the attacker must also ensure that its reported value is not discovered to be skewed. To ensure this, the attacker must add a detection constraint to its list of constraints. This is possible because it is assumed that the attacker is aware of the procedure used to determine if a particular value has been skewed. In our example, we assume that a value is detected as skewed if it is determined to be the most discrepant from the final result. Thus, the detection constraint can be written:

$$\left|\sqrt{(x_C - x_F')^2 + (y_C - y_F')^2}\right| \le \max\left(\left|\sqrt{(x_A - x_F')^2 + (y_A - y_F')^2}\right|, \left|\sqrt{(x_B - x_F')^2 + (y_B' - y_F')^2}\right|\right) \tag{16}$$

Again, because there exists many solutions to this system of constraints, the attacker must be guided by some objective function to choose the best possible solution. Intuitively, the objective function should maximize the difference between the non-skewed result and the skewed result. This is the root-mean-

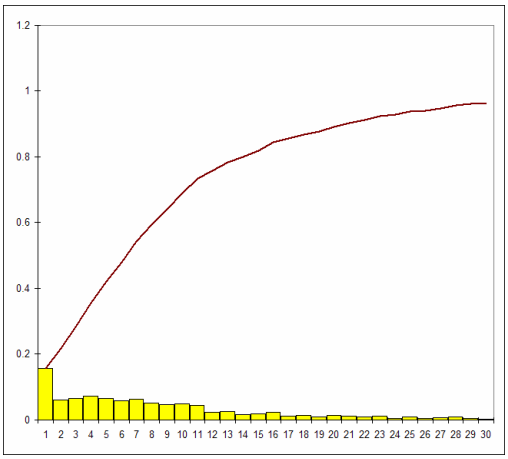

Figure 5: This figure shows a histogram of ratios with bin 1 containing values from 0 to 1, bin 2 containing values from 1 to 2, and so forth. The two lines represent the cumulative probability function.

square function, and can be written mathematically as:

$$\max: \left| \sqrt{(x_F - x_F')^2 + (y_F - y_F')^2} \right| \qquad (17)$$

Now that we have determined an appropriate system of constraints and an objective function to determine the skewed and non-skewed results, we must solve these systems using an optimization method.

The method for determining the actual solution can be modified to produce a skewed value for the attacker and a skewed solution. To do so, note that before the attacking node (*C*) reports its value there exists two possible valid solutions to the system of equations. This can be seen in Figure 4. Intuitively, node *C* simply needs to report a value that skews the result towards the incorrect solution. To do this, node *C* calculates the intersection of the circles defined by nodes *A* and *B* and their corresponding radii. It then, using its own correct value, calculates the correct area in which the solution exists. Next, it superimposes an $n \times n$ grid over the area where the correct solution *does not exist*. The attacking node then calculates for each grid square the objective function and checks to see if it adheres to all constraints. If the grid square does adhere to all constraints and its objective function value is maximum, an $n \times n$ grid is superimposed inside of it and the process continues, until a user-specified number of iterations has been reached.

V.A. Attack Simulation Results

We ran several simulations that determine the effectiveness of this method for skewing data and when it is easier for an attacker to skew data. The goal of our first simulation was to show how much damage an attacking node could do when a node *P* is trying to determine its position using atomic trilateration and distances and positions of three other nodes (*A*, *B*, and *C*) of known location. The value used to determine the effectiveness of our algorithm is the *skewed/non-skewed ratio*. This value is defined as being the distance from the skewed solution to the actual value divided by the distance from the non-skewed solution to the actual value. If this ratio is larger than one, it implies that the attacker was able to do more damage to the final result by skewing its data than by not skewing it. If the ratio is less than one, the attacker actually did less damage to the final result by skewing its data than by not skewing it. In other words, the skewed solution was in fact *closer* to the actual value than the non-skewed solution. The results of our simulation can be seen in Figure 5. In this graph, we show a histogram of the skewed/non-skewed ratio. The bins for the histogram are [0, 1], [1, 2], and so forth. In addition, we plot the cumulative probability function in red (the increasing line) for the skewed-non-skewed distance ratio. This line represents the probability of calculating a ratio that lies in a bin that is less than or equal to a particular bin. The decreasing line in the graph is equal to one minus the cumulative probability function (i.e. it is the probability that a particular value will land in a bin larger than the current bin). The intersection of these two lines represents the value at which half the time the result is above and half the time the result is below (the median). Thus, in Figure 5, one can see that the median skewed/non-skewed ratio is near 6 (5.822581). In other words, the attacker, when skewing its result, is on average able to produce a result that is approximately six times farther away than the non-skewed solution.

The results of the first simulation demonstrate the importance of data skewing using a sensor network as the target platform. However, there exist times when achieving particular a skewed/non-skewed ratio is more difficult. Thus, in our next simulation, we attempt to determine when it is easier for an attacker to significantly skew data and when it is not (again, using a sensor network as the target platform). In this simulation, we are assuming that a node *P* is trying to calculate its location by using distances and locations of three other nodes – *A*, *B*, and *C*. In this simulation, node *A*'s location was fixed at (0, 0). Node *B*'s location was fixed at (1, 0). Node *C*'s location was placed somewhere on the semicircle defined by the circle with node *A* located in the center and node *B* on the circle itself. The angle between *AB* and *AC* was varied from 1 to 179 and we measured the skewed-non-skewed distance ratio. In doing so, we hope to determine the orientation of nodes that yields the maximum potential for skewing data. In all circumstances, node *P* was randomly placed inside the triangle formed by nodes *A*, *B*, and *C*. For each triangle orientation of nodes *A*, *B*, and *C*, node *P* was randomly placed inside 10,000 times and the skewed/non-skewed ratio was measured. The results can be seen in Figure 6.

VI. Defense Protocols

In this section we discuss two basic methods of defending oneself from an attacker. The first line of defense can be mathematically described as over-constraining the system. By adding more constraints to the system, the attacking node has less room by which it can skew the resulting calculated solution. Furthermore, adding constraints has the potential to reduce the number of possible solutions to a system of equations, which thus can provide a significant amount of protection.

The second method for defending oneself from an attacker is to vary the order in which nodes are queried. By varying the order, the attacking node does not always receive complete information about the network and thus must estimate any remaining values. In order to skew the data effectively and reduce the chance of detection, the attacking node must, when the query order is varied, determine some level of certainty that it will not be detected. The amount that the attacker can skew the result is dependent upon this threshold of detection, which we demonstrate later.

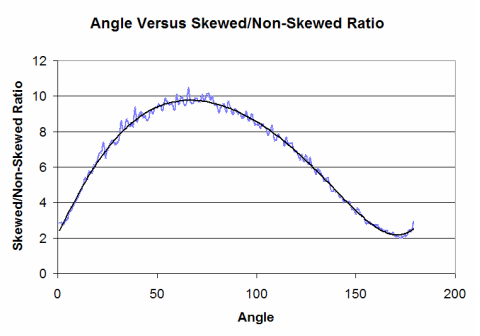

Figure 6: This figure shows how the skewed/non-skewed ratio changes as the angle formed by the nodes varies from 1 to 179 degrees.

VI.A. Over-Constrain

A node can defend itself from attacking nodes using several different means. One simple means of defense is to over-constrain the problem. In doing so, a node hopes that it will receive enough non-skewed responses to not only accurately calculate a solution but also to reduce the possibilities by which an attacking node can skew data.

We will illustrate this idea of over-constraining a problem by showing how it applies to the example of atomic trilateration – the situation in which a node determines its location through distances to and positions of other nodes. Assume that node *P* is trying to determine its own position using nodes *A*, *B*, *C*, and *D* with positions (x_A, y_A), (x_B, y_B), (x_C, y_C), (x_D, y_D) and radii (measured distances to node *P*) R_A, R_B, R_C, and R_D. Assume that node *D* is trying to skew the final result, (x_F, y_F), such that the distance between the skewed result, (x_F', y_F'), and the non-skewed result is maximal. Also assume that node *D* is the last node to reports its value.

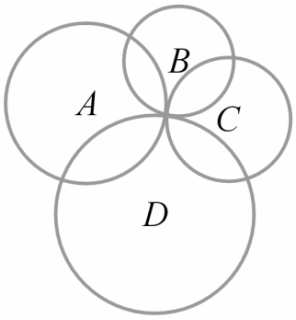

Figure 7: This figure demonstrates how increasing the number of sensors (and thus constraints) reduces the attackers ability to skew the final result.

Since node *D* is the last node queried for its value, *D* is able to overhear the values reported by nodes *A*, *B*, and *C*. With this information, node *D* is able to determine the following constraints to determine (x_F, y_F):

$$\left|\sqrt{(x_A - x_F)^2 + (y_A - y_F)^2} - R_A\right| \le \varepsilon_A$$
$$\left|\sqrt{(x_B - x_F)^2 + (y_B - y_F)^2} - R_B\right| \le \varepsilon_B$$
$$\left|\sqrt{(x_C - x_F)^2 + (y_C - y_F)^2} - R_C\right| \le \varepsilon_C \quad (18)$$
$$\left|\sqrt{(x_D - x_F)^2 + (y_D - y_F)^2} - R_D\right| \le \varepsilon_D$$

Note that there are two paradigms by which an attacker can skew its data. The first paradigm is that the attacker attempts to skew data by manipulating its reported by hiding within existing errors in measurement. The second paradigm by which the attacker can skew its data is to force the calculated result to different (yet still mathematically valid) solution. However, when one over-constrains the system, the number of alternate solutions decreases, and thus, the attacker's ability to skew data diminishes significantly. In addition, because more readings are being used in the calculation of the final answer, the attacker's ability to exploit errors in measurement is reduced (due to having to stay within acceptable margins of these additional readings).

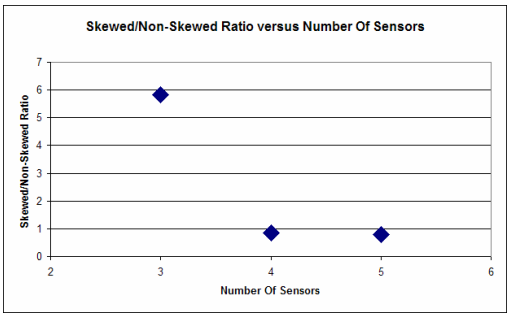

Figure 8: This figure demonstrates the trend of the skewed/non-skewed ratio as the number of sensors increases.

From a geometric perspective, over-constraining the problem of atomic trilateration results in more intersecting circles, and when using four readings in two-dimensional space, the four radii (for a non-pathological placement of sensors) intersect at only one point. Thus, if a node trying to position itself uses only one extra sensor (assuming that the system contains only one liar), the node is able to greatly enhance its approximation of its actual position. This can be seen in Figure 4.

VI.B. Over-Constrain Simulation Results

We have run several simulations demonstrating how increasing the number of constraints provides greater assurance that the calculated result is accurate. To demonstrate this increase in accuracy, we use 3-sensor, 4-sensor, and 5-sensor networks. For each type of network, we randomly generate 1000 examples and record the skewed/non-skewed ratio.

Figure 8 is a plot of the number of sensors used in trilateration versus the corresponding skewed/non-skewed ratio. Each ratio in this measurement was determined by calculating the median value over 1000 randomly generated scenarios. As one can see, when the number of sensors is increased from 3 to 4, the skewed/non-skewed ratio decreases significantly. However, further increasing the number of sensors used in atomic trilateration does not have as significant an impact. This figure demonstrates two concepts. First, over-constraining the problem decreases the attacker's ability to skew its data effectively.

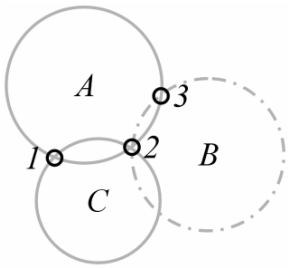

Figure 9: In this figure, point 2 represents where the actual result lies and what node *C* is trying to approximate using method 1. Point 3 represents an alternate solution if node *B*'s value were known, and what node *C* is trying to approximate using method 2.

Second, further over-constraining the problem does not yield significant benefits.

VI.C. Query Order

One method to defend oneself against an attacking node is to vary the order in which nodes are queried for data. By doing this, the attacking node does not know when it will be selected to report its value, and thus, it is not able to know in advance what information from the other nodes it can rely upon. Therefore, the attacking node's ability to skew data is limited, and if the attacker does not skew the data properly, its likeliness for detection increases.

To further illustrate this idea, we will demonstrate how this problem applies to atomic trilateration. In this example, assume that node *P* is trying to determine its own position using nodes *A*, *B*, and *C* with positions (x_A, y_A), (x_B, y_B), and (x_C, y_C) and radii (measured distances to node *P*) R_A, R_B, and R_C. Assume that *P* queries node *A* first, followed by node *C*, and then node *B*. Further assume that node *C* is trying to skew the final result, (x_F, y_F), such that the distance between the skewed result, (x_F', y_F'), and the non-skewed result is maximal.

Since *C* is queried second, it is only able to overhear the radius reported by node *A*. However, because *C* is also able to determine its own radius to point *P*, *C* is able to narrow down the list of possibilities of where *P* can be to two regions, which are defined by the intersection of the two radii of nodes *A* and *C*. Using our generic method for skewing data, we will derive a system of equations to formalize this problem of when the attacker reports second in atomic trilateration. Because

Estimate the solution using the known reported values.

Determine upper and lower bounds for the skewed reported value.

For each value between the upper and lower bounds (separated by intervals of 1 / G):

> *Calculate N scenarios by randomly choosing the remaining values.*
>
> *For each scenario, calculate the resulting solution and determine the suspected attacking node.*
>
> *If the attacking node is unsuspected in at least R ? N scenarios, record this solution if it is maximally offset from the estimated solution.*

Figure 10: Formal algorithm describing the Monte Carlo procedure used by an attacker to maximally skew a computation while maintaining a certain probability of detection.

node *C* is only able to determine its value and the value reported by node *A*, node *C* is only able to determine two constraints to define (x_F, y_F), which are similar to (15) except that the reading from node *B* is not present.

Node *C* does not know node *B*'s radius or position because these values have not been reported by node *B* yet. Because this system is under constrained, node *C* is not able to uniquely determine a value for (x_F, y_F), and thus might not be able to skew the data accurately.

Because of a lack of information (in this case, the value reported by node *B*), the attacker must make an educated guess as to where the final result will be (and when using method 2, where an alternate result will be). To do this, we propose a Monte Carlo simulation method. The Monte Carlo method depends on several parameters: the number of Monte Carlo tries per reported value (*N*), the success rate of the attacker (*R*), and the granularity of the reported values (*G*). The Monte Carlo method works as

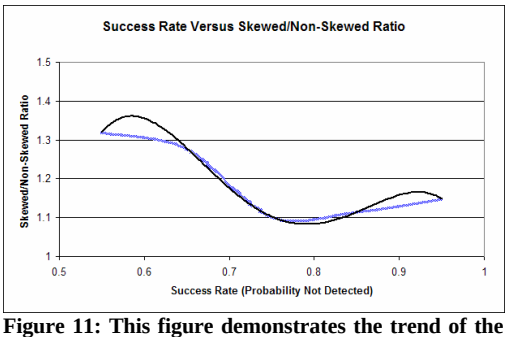

Figure 11: This figure demonstrates the trend of the skewed/non-skewed ratio as the success (non-detection) rate of the attacker increases.

follows. The attacking node must first estimate the final solution. Then, the attacking node must determine a set of boundaries within which it can report. Next, for each value within the boundaries (with each value occurring at intervals of 1 / *G*), the attacker will create *N* scenarios by randomly choosing the values to be reported by the remaining nodes *N* times. For each scenario, the attacker will calculate the resulting solution and determine if it is suspected to be skewing its data. If the number of scenarios in which the attacker is unsuspected is larger than *R* ? *N*, the current value is remembered. Finally, the attacker chooses from each remembered value the value that is maximally offset from its original estimated value. This Monte Carlo method is more formally stated in Figure 10.

This Monte Carlo method of defense can be applied to atomic trilateration in the following way. Node *C* must first store the two intersection areas where it believes the actual solution lies. Then, node *C* sets its skewed reported radius to the minimal radius, which can be determined by calculating the distance from node *A* to node *C* and subtracting from that the measured radius reported by node *A*. This minimal radius corresponds to the scenario where the radii from nodes *A* and *C* intersect at exactly one point between nodes *A* and *C*. Using this radius, node *C* then randomly chooses *N* locations and radii for node *B*. The radii are determined by randomly choosing from the two areas where the solution may lie and adding a random error. For each scenario, node *C* then calculates the final solution and checks to see which node is determined to be the attacker. This is repeated *N* times, and if node *C* is not suspected in *R* ? *N* Monte Carlo simulations, node *C*'s current reported radius is stored. Node *C*'s radius is incremented by 1 / *G* (where *G* is the granularity of measurements) and the Monte Carlo procedure is repeated until the maximum radius is reached, which is determined by calculating the distance from node *A* to node *C* and adding to that the measured radius reported by node *A*. This radius corresponds to the scenario when the circles from nodes *A* and *C* intersect at exactly one point that is not between nodes *A* and *C*. From each value of node *C*'s radius that is unsuspected at least *R* ? *N* times node *C* selects the value that maximally offsets the calculated solution. The offset to a solution is calculated by determining the distance between the calculated solution and the closer of the two initial solution estimations.

VI.D. Query Order Simulation Results

We ran several Monte Carlo simulations, with *N* being equal to 100, *R* varying from 0.65 to 0.95, and *G* being 100. These simulations were designed to validate the intuitive idea that as the Monte Carlo success rate increases, the ability to skew data decreases. In our Monte Carlo method simulations, we show how the attacker has less potential to skew its data and thus the effectiveness of the Monte Carlo method. As one can see from Figure 11, as the attacker's success rate (probability of not being detected) increases, its ability to skew data decreases.

VI.E. Comparison

From Figures 8 and 11, one can see that both over-constraining the problem and varying the query order can significantly reduce the attacker's potential to skew data. However, judging from the skewed/non-skewed ratios determined by each method, it appears that over-constraining the problem actually causes an attacker (when attacking) to skew the result closer to the actual solution, and thus, is a better method for defending oneself from attackers. However, the tradeoff associated with this approach is that it is more costly and, in the context of sensor networks, requires not only more sensors but also more communication, which uses more energy and thus reduces the lifetime of the network. On the other hand, varying the query order is a simple mechanism that achieves significant results without increasing the size of the

network or reducing its lifetime through excess communications.

VII. Related Work

Misinformation theory and application is a new topic and therefore there are no directly related efforts. From a more general perspective, information theory, fault-tolerance, and in particular fault-tolerance in distributed systems, security and cryptography in sensor networks, and several other efforts relate to the treatment of skewed data in the computer science theory community.

Shannon started Information theory [Sha48]. The standard references include [Ash65] and [Cov91]. It is interesting to note that Shannon also introduced the first set of principles for obfuscation of information [Sha49] that form the basis for building cryptographic systems. Although no work has had as much impact and broad importance and applicability as that done by Shannon, we hope to build upon the theory presented by Shannon with our misinformation theory.

Location discovery is an important problem that occurs frequently in wireless embedded sensor networks. The works done by [Bul00], [Doh01], and [Bul02] help demonstrate the importance of this problem and illustrate how trilateration is a common technique used for location discovery.

Fault tolerance studies techniques on how to handle misinformation that is introduced randomly. There exists a vast quantity of literature on this topic ([And81], [Jal94], [Lyu95]). Most related fault tolerance work is one done in the distributed systems community [Lyn96], and in particular the Byzantine generals problem [Lam82] and [Lam83]. Another related approach is Rabin's scheme for efficient dispersal of information for security, load balancing and fault tolerance. In a sense, the most closely related fault tolerance work is that done by Marzullo in a series of papers ([Mar90] and [Mar97]). These papers propose techniques on how to aggregate information from faulty sensors. The main difference between our work and that of Marzullo is that Marzullo determines bounds within which a solution may exist. We, on the other hand, hope to push the reported solution to a different (yet still valid) solution. Finally, in the theoretical computer science community, several versions of how to play the twenty questions game when one of players is reporting incorrectly was addressed from several different points viewpoints ([Dha92] and [Amb99]). Furthermore, Koushanfar et al. studied fault tolerance techniques for sensor networks [Kou02]. Although the addition of constraints can restrict the amount by which one can skew data, because errors will continue to exist in the network, it will still be possibly to maximally skew data even with additional constraints, although the extent may become significantly lessened.

Two techniques for facilitating security and privacy in sensor networks have been proposed. One uses cryptographic techniques [Per01] and one uses system-building techniques [Cor02]. Good starting points for sensor network research include [Ten00], [Pot00] and [Est00], and good starting points for semantic misinformation can be seen in from [Lib94] and [Sch00].

VIII. Conclusion

We have developed the first quantitative misinformation scheme for sensor networks. The important aspects of this task are identified and abstracted so that the problem can be posed as either an instance of nonlinear programming or as a Monte Carlo-based optimization. A number of attacks and defense schemes have been proposed and analyzed. We demonstrated the effectiveness of this approach with techniques and algorithms on two canonical tasks in sensor networks: single value averaging and atomic trilateration.

IX. References

[Amb99] A. Ambainis, S. A. Bloch, and D. L. Schweizer, *"Playing Twenty Questions with a Procrastinator,"* In Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, 1999.

[And81] T. Anderson and P. A. Lee, *"Fault Tolerance, Principles and Practice,"* Prentice/Hall International, 1981.

[Ash65] R. Ash, *"Information Theory,"* Interscience Publishers, 1965.

[Bir85] K. P. Birman, *"Replication and Fault-Tolerance in the Isis System,"* Proceedings of the Tenth ACM Symposium on Operating Systems Principles, 1985.

[Bul00] N. Buluso, J. Heidemann and D. Estrin, *"GPS-Less Low Cost Outdoor Localization For Very Small Devices,"* IEE Personal Communications, 2000.

[Bul02] N. Bulusu, J. Heidemann, and T.Tran, *"Self-Configuring Localization Systems: Design and Experimental Evaluation,"* ACM Transactions on Embedded Computing Systems, 2002.

[Cor02] M. Corner and B. Noble, *"Zero-Interaction Authentication,"* Conference on Mobile Computing and Networking, 2002.

[Cov91] T. Cover and J. Thomas, *"Elements of Information Theory,"* John Wiley & Sons, 1991.

[Dha92] A. Dhagat, P. Gacs, and P. Winkler, *"On Playing 'Twenty Questions' with a Liar,"* In Proceedings of the Third Annual ACM-SIAM Symposium on Discrete Algorithms, 1992.

[Doh01] L. Doherty, K. Pister, and L. El Ghaoui, *"Convex Optimizatoin Methods for Sensor Node Position Estimation,"* IEEE INFOCOM, 2001.

[Est00] D. Estrin, R. Govindan, J. Heidemann, *"Embedding the Internet – Introduction,"* Communications of the ACM, 2000.

[Jal94] P. Jalote, *"Fault Tolerance in Distributed Systems,"* P.T.R. Prentice Hall, 1994.

[Kou02] F. Koushanfar, M. Potkonjak, A. Sangiovanni-Vincentelli, *"Fault Tolerance in Wireless Ad Hoc Sensor Networks,"* IEEE Sensors, 2002.

[Lam82] L. Lamport, R. Shostak and M. Pease, *"The Byzantine Generals Problem,"* ACM Transactions on Programming Languages and Systems, 1982.

[Lam83] L. Lamport, *"The Weak Byzantine Generals Problem,"* J. ACM, 1983.

[Lib94] M. Libicki, *"The Mesh and the Net: Speculations on Armed Conflict in an age of Free Silicon,"* National Defense University McNair Paper 28, 1994.

[Lyn96] N. Lynch, *"Distributed Algorithms,"* Morgan Kaufman, 1996.

[Lyu95] M. R. Lyu, *"Software Fault Tolerance,"* Wiley, 1995.

[Mar90] K. Marzullo, *"Tolerating Failures Of Continuous-Valued Sensors,"* ACM Transactions on Computer Systems, 1990.

[Mar97] K. Marzullo and M. Clegg, *"Predicting Physical Processes in the Presence of Faulty Sensor Readings,"* Twenty-Seventh Annual International Symposium on Fault-Tolerant Computing, 1997.

[Per01] A. Perrig, R. Szewczyk, V. Wen, D. Cullar, and J. D. Tygar, *"SPINS: Security Protocols For Sensor Networks,"* Conference on Mobile Computing and Networking, 2001.

[Pot00] G. J. Pottie, W. J. Kaiser, *"Wireless Integrated Network Sensors,"* Communications of the ACM, 2000.

[Rab89] M. Rabin, *"Efficient Dispersal of Information For Security, Load Balancing and Fault Tolerance,"* J. ACM, 1989.

[Rap60] T. S. Rappaport, *"Wireless Communications: Principles and Practice,"* P.T.R. Prentice Hall, 1996.

[Sch00] B. Schneier, *"Semantic Attacks: The Third Wave of Network Attacks,"* Crypto-Gram Newsletter, 2000.

[Sha48] C. E. Shannon, *"A Mathematical Theory of Communication,"* Bell Systems Technical Journal, 1948.

[Sha49] C. E. Shannon, *"Communication Theory of Secrecy Systems,"* Bell Systems Technical Journal, 1949.

[Sli02] S. Slijepcevic, S. Megerian, M. Potkonjak, *"Location Errors in Wireless Embedded Sensor Networks: Sources, Models, and Effects on Applications,"* ACM Mobile Computing and Communications Review, 2002.

[Ten00] D. Tennenhouse, *"Proactive Computing,"* Communications of the ACM, 2000.