Resilient Aggregation in Sensor Networks

David Wagner[*]
University of California at Berkeley

ABSTRACT

This paper studies security for data aggregation in sensor networks. Current aggregation schemes were designed without security in mind and there are easy attacks against them. We examine several approaches for making these aggregation schemes more resilient against certain attacks, and we propose a mathematical framework for formally evaluating their security.

Categories and Subject Descriptors

D.4.6 [**Operating Systems**]: Security and Protection

General Terms

Security

Keywords

aggregation, sensor networks, node capture attack, multiparty computation, robust statistics, average, mean, median

1. INTRODUCTION

The goal of this paper is to examine secure data aggregation in sensor networks. Sensor networks have been proposed for scientific data collection, environmental monitoring, building health monitoring, burglar and fire alarm systems, and many other applications involving distributed interaction with the physical environment. Many of these applications involve a distributed system of sensors measuring the environment from many vantage points and then somehow aggregating the collected data to form a global summary view that can be acted upon. Consequently, data aggregation can be viewed as an important building block in sensor networks. Unfortunately, even though security has been identified as a major challenge for sensor networks [2, 11], current proposals for data aggregation protocols have not been designed with security in mind, and consequently they are vulnerable to easy attacks. In this paper, we undertake an in-depth study of security for data aggregation in sensor networks.

First, we show that existing proposals for data aggregation are subject to attack (Sections 3 and 4). When a single sensor node can be captured, compromised, or spoofed, an attacker can often manipulate the result of the aggregation operation without limit, gaining complete control over the computed aggregate. This is undesirable. For instance, we show that any protocol that computes the average, sum, minimum, or maximum function is insecure against malicious data, no matter how these functions are computed.

In response to this threat, we introduce a theoretical framework for modeling the security of data aggregation. This model insists that the aggregation function must be resilient in the presence of arbitrary changes to a small subset of sensor observations, and thus we coin the term *resilient aggregation* to refer to schemes that satisfy this condition. We formalize this condition precisely (Section 5) and characterize which functions achieve the resilience condition (Section 6). For instance, we show that the median is a more robust alternative to the average. Many of our conclusions are consistent with intuition; the value of our mathematical framework is that one can place this intuition on a firm foundation, and one can analyze systems that are too complex for intuition to provide sufficient guidance.

Finally, we introduce several techniques and principles for achieving resilient aggregation in new protocols (Section 7). For instance, we show that outlier elimination (trimming) is a powerful aggregation technique that provides inherent robustness against attack.

This paper makes three scientific contributions:

- The paper describes attacks on standard schemes for data aggregation and introduces the problem of securing aggregation in the presence of malicious or spoofed data. The attacks are quite obvious, but the crisp problem statement we give has not appeared in print before.
- The paper proposes a mathematical theory of security for aggregation. This theory lets us quantify, in a principled way, the robustness of an aggregation operator against malicious data. The paper draws novel connections to statistical estimation theory and to the field of robust statistics.
- The paper identifies techniques for aggregation that provide robustness against attack. The techniques are

[*] daw@cs.berkeley.edu. Supported by DARPA NEST contract F33615-01-C-1895.

SASN'04, October 25, 2004, Washington, DC, USA.

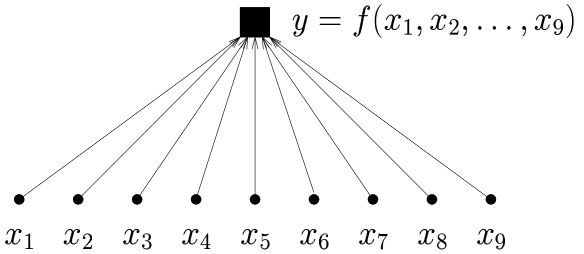

Figure 1: An abstract sensor network architecture, with inessential underlying physical structures abstracted away. We have n sensor nodes (the small circles), each with a separate secure channel to a single trusted base station (the large solid square). The ith sensor sends measurement x_i to the base station, and the base station uses the function f to compute the aggregate y. In this picture, $n = 9$.

not novel, but the analysis of their performance in this setting is new. This should provide helpful guidance to sensor network implementors in selecting appropriate aggregation operators.

2. BACKGROUND

The architecture. A sensor network is a distributed system designed for interacting with the physical environment. A sensor network might contain hundreds or thousands of tiny, low-cost, low-power sensor nodes. Many architectures also use more powerful base stations, which are in a one-to-many association with sensor nodes. Often, one forms a tree with a base station at the root and sensor nodes at the leaves. An aggregation transaction begins by broadcasting the query down the tree from the base station to the leaves. Then, the sensor nodes measure their environment and send their measurement back up the tree to the base station. Finally, the base station performs an aggregation computation to obtain the aggregate. Thus, sensor nodes act as data sources, and the base station acts as a sink.

An abstract model. In this paper, we will abstract away some inessential features. We consider only a single base station and n associated sensor nodes. At some point, each sensor node takes a measurement and reports the observed value x_i to the base station. The base station's goal is to compute an aggregate value y that summarizes the sensor readings $x_1, \ldots, x_n$ using the aggregation function f; thus, $y = f(x_1, \ldots, x_n)$. Our attacks and defenses ignore the inner workings of the specific aggregation and communication protocols used, such as how data is routed. Instead, we focus only on the function f computed by the base station. Consequently, we ignore the structure of the multi-hop network, assuming only that each sensor node has a separate link to the base station. We depict this abstracted architecture in Figure 1.

Threat model. We will assume that each sensor node has a secure channel back to the base station for reporting data measurements. Moreover, we assume these secure channels are independent: capture of one sensor node might compromise the contents of that node's channel to the base station, but it will not reveal anything about other nodes' channels. For instance, each sensor might share a per-node symmetric key with the base station and use this key to encrypt its data. As a consequence of these assumptions, we do not need to worry about spoofing or interception of data in transit. This leaves only the question of whether the endpoints are trustworthy or not.

The main threat we will consider is that of malicious data. One way that malicious data can be injected is through node compromise or node capture attacks: if a sensor node is captured, reverse-engineered, or otherwise comes under adversarial control, we can no longer trust its measurements. Alternatively, if an adversary is able to fool the sensor's measuring element—perhaps by subjecting it to unusual temperature, lighting, or other spoofed environmental conditions, for instance—then the sensor node's measurement is compromised. Thus, measurements can often be compromised even if nodes are protected by tamper-resistant packaging.

We will use a Byzantine fault model, where the adversary can inject arbitrarily chosen malicious data readings at a few sensors. Of course, compromised nodes may behave in arbitrarily malicious ways, which means that measurements from compromised nodes are under the complete control of the adversary. We conservatively assume that all compromised nodes collude, or are under the control of a single attacker. An archetypical attack involves compromised nodes reporting bogus measurements in an attempt to skew the computed aggregate.

Compromise of sensor nodes is indeed a real threat in real sensor networks. Because sensor nodes must be low-cost, we often cannot afford to mount them in physical packaging that provides a high level of tamper resistance. Because sensor nodes must be deployed into the environment, we cannot provide physical security or control access to them. And, because sensor nodes must be deployed in large numbers, the adversary is afforded many opportunities to compromise a sensor node.

Of course, the adversary's capabilities are not unlimited. Some cost or luck will be required for each node that the adversary wishes to compromise. Therefore, we should assume that the adversary can compromise only a limited number of sensor nodes: perhaps one or two or three, but not half of the network. We require that security degrade gracefully as the number of compromised nodes grows. In this respect, our main hope is for "safety in numbers": if we can build a network that is robust in the presence of a few malicious or captured sensor nodes, we will be in great shape.

In this paper, our analysis will assume that base stations remain trustworthy and unassailable. This assumption seems plausible: base stations are rarer and hence we can spend more on them, so it may be feasible to enclose them in high-quality tamper resistant enclosures or to place them in physically secure, access-controlled locations.

Security goals. We focus on *integrity.* The adversary should not be able to affect the result of the aggregation operation, as computed at the base station. This should remain true even in the presence of a few compromised sensor nodes.

Unfortunately, perfect integrity is rarely attainable. By manipulating the readings at a few compromised sensor nodes, the attacker can usually affect the computed aggregate, even if only negligibly, no matter how clever the protocol. Therefore, we relax our goal slightly and ask for *approximate integrity*: the adversary should have only a limited influence on the result of the aggregation computation. In other words, if y denotes the result in the absence of an attacker and y^* the result after an attacker intervenes, then we wish $|y^* - y|$ to be bounded, preferably by some small value. Notice that if $|y^* - y|$ is negligible compared to the random noise in y, the attacker has gained little. Because measurements of the physical world are inherently noisy, we expect approximate integrity to form an adequate security goal for most applications.

We separate the specification of desired aggregation functionality from the protocol used to implement or achieve this functionality. We assume the functionality is given as a function f; given sensor readings $x_1, \ldots, x_n$, we wish to compute the aggregate $y = f(x_1, \ldots, x_n)$. A valid protocol might use any mechanism whatsoever to compute this aggregate. See Figure 1 for an abstract view.

The central question for secure aggregation is as follows:

> *Question.* Which aggregation functionalities can be securely and meaningfully computed, in the presence of a few compromised nodes?

The following sections are directed at answering this central question.

Caveats. We do not consider confidentiality, availability, or performance in this paper. We do not consider in-network aggregation; in our model, trusted base stations are the only aggregation points.

Connections to secure multi-party computation. The central question for secure aggregation carries some connection to the area of secure multi-party computation. It is known that any functionality that can be computed with the help of a trusted third party can also be computed without it, using generic multi-party computation. The connection to sensor networks is natural: we may think of aggregation in sensor networks as an instance of multi-party computation *with* a trusted third party, where the functionality is given by the aggregation function f and where the base station plays the role of the trusted third party.

The question of which aggregation functionalities f can be meaningfully computed in sensor networks now corresponds to the following question about multi-party computation: Which functionalities f can be meaningfully computed by a protocol for secure multi-party computation, when some parties might behave maliciously by submitting bogus inputs? In the general case, this question does not seem to have been considered in the literature on generic multi-party computation. The literature has focused on demonstrating that anything that can be computed with a trusted third party can be computed without a trusted third party. However, whether or not we assume that we have a trusted third party, there is no guarantee that what we want to compute is meaningful in the presence of malicious participants. For instance, if a group of cryptographers wishes to learn their average salary without disclosing anything else, there is a potential problem: whether or not we use a trusted third party, a single malicious cryptographer can cause the computed result to deviate tremendously from the correct result.

To put another way, the difference between this work and previous research on generic multi-party computation is the difference between specification and implementation. Prior work has shown that any functionality you can specify, you can implement; we focus on asking whether a specified functionality is meaningful in its own right.

Applications to system robustness. Even in the absence of an adversary, resilient aggregation functions may have utility in improving robustness against random faults. A single corrupted measurement should not cause large errors in the computed aggregate. Standard aggregates, such as the average, cannot achieve this goal (see Section 3); a single malfunctioning sensor returning random results can skew the average by an unbounded amount.

Of course, any aggregate that is resilient against malicious attack will also be resilient against random failures. Consequently, resilient aggregation may also be of independent interest for its use in improving overall system reliability.

3. ATTACKS ON EXISTING AGGREGATION PRIMITIVES

An example. To give the flavor of the kind of attack possible on naive aggregation protocols, let us consider a whimsical example. Imagine a large building with a control network that regulates inside temperatures by measuring the temperature in each room, computing the building-wide average, and deciding whether to turn on the air conditioning or not according to whether the average temperature exceeds some threshold.

Now imagine a building occupant who prefers climates much cooler than the system is programmed to provide. It is easy to see that she can manipulate the system, simply by holding her cigarette lighter under a single sensor. Why does this work? The key observation is that she has artificially increased this sensor's temperature reading by hundreds of degrees, which will often have the effect of raising the building-wide average temperature above the threshold, because huge changes to a single sensor reading can cause noticeable changes to the average. This maliciously-skewed average can thereby trigger the air conditioning unit into turning on when it otherwise would not have.

We see that a single attacker has succeeded in hijacking control of the building's climate merely by fooling a single sensor. This means that our hypothetical climate control network is fragile against malicious attack. Even though in this example the operation of the air conditioning unit is probably not particularly security-critical, this illustrates a general problem that can have more serious consequences in some real systems. This is a failure mode that we would prefer to avoid.

Some functionalities are inherently insecure. Earlier, we suggested separating the choice of aggregation functionality (specified by the function f) from the way our protocol computes this aggregate. For instance, in the building-control example above, we had $f(x_1, \ldots, x_n) = (x_1 + \cdots + x_n)/n$. Notice that our example attack did not depend in any way on how we compute this function. In other words, it is the specification itself that was fundamentally faulty, not the implementation.

This is a point of such general applicability that it bears repeating. Some functions simply cannot be computed securely in the presence of compromised nodes, no matter what protocol we use. Because a single node can exert total control over the building-wide average temperature, the average temperature is not a meaningful quantity when nodes are compromised.

In the remainder of this section, we will present attacks on several popular aggregation functions. These attacks are straightforward and obvious upon inspection, once one recognizes the importance of resilience; however, we have not seen these attacks described before in the sensor network literature.

The average is insecure. We saw earlier that the average, given by $f(x_1, \ldots, x_n) = (x_1 + \cdots + x_n)/n$, is insecure in the presence of a single malicious sensor node. Say that sensor node 1 is compromised. Then by substituting the fake reading x_1^* in place of the real measurement x_1, the average is changed from $y = f(x_1, \ldots, x_n)$ to $y^* = f(x_1^*, x_2, \ldots, n) = y + (x_1^* - x_1)/n$. Since the attacker can choose x_1^* freely, the attacker can exert complete control over the result. For instance, if the attacker wants to artificially add a bias δ to the average, then he can set $x_1^* = x_1 + \delta n$, and the average will be successfully altered from y to $y^* = y + \delta$. Consequently, the average is not a meaningful aggregate in the presence of even a single compromised node.

The sum is insecure. Similarly, the sum $f(x_1, \ldots, x_n) = x_1 + \cdots + x_n$ is not meaningful in the presence of one or more compromised nodes. The attacker can freely increase or decrease this value without limit.

The count can be secured. A related primitive is the count, which is like the sum, except that each node contributes 0 or 1 to the total. If an incautious implementation forgets to check that each node's value is in the set $\{0, 1\}$, then it will be susceptible to the same attack as the sum. However, if properly implemented, we know of no serious attacks on the count. An attacker with control over k compromised nodes can only change the count by at most k, and hence if the number of compromised nodes is limited, this may be acceptable.

The minimum is insecure. Consider computing the minimum of the sensor readings, $f(x_1, \ldots, x_n) = \min(x_1, \ldots, x_n)$, and suppose that sensor 1 is compromised. The attacker can only increase the minimum if x_1 is the unique smallest sensor reading, and even then, the minimum is raised to $\min(x_1^*, x_2, \ldots, x_n)$, which cannot exceed $\min(x_2, \ldots, x_n)$. Thus, the attacker has little capacity to increase the computed aggregate. However, the attacker can freely reduce the computed minimum value, simply by choosing x_1^* to be a very small (or even negative) value. So long as the attacker's desired outcome is smaller than the correct outcome, the attacker has complete control. Therefore, we consider that the minimum is not resilient against false sensor readings. In the presence of a single compromised node, the minimum is not a meaningful aggregate to compute.

The maximum is insecure. By symmetry, the maximum is also not meaningful in the presence of a compromised node.

4. VULNERABILITIES OF EXISTING SYSTEMS

Many of today's sensor network systems are susceptible to the attacks described above. A few examples should to serve to illustrate the point.

TinyDB. TinyDB is a database-centric interface to sensor networks, where data aggregation is expressed with a SQL-like query language [8, 5]. The database consists of a table, where each sensor provides one row in the table, and each row provides values for several attributes (e.g., temperature, humidity, light, acceleration). Queries compute an aggregate of the listed attribute, or of a simple boolean or arithmetic expression of a single attribute. TinyDB supports five conventional aggregates: the minimum, maximum, average, count, and sum. Also, TinyDB supports temporal aggregates: a sliding window of previous sensor readings is kept, and then the minimum, maximum, average, count, or sum can be applied to this recent history.

Based on the above discussion, we see that TinyDB is insecure in the presence of compromised nodes when the minimum, maximum, average, or sum aggregates (or their corresponding temporal aggregation variants) are used. The count (or sliding window count) is the only primitive that is resilient to malicious data.

Other systems. TinyDB is not alone. For example, the Cougar sensor database system supports SQL-like queries with aggregation operators, much like TinyDB [15]. Likewise, the SensorWare project proposes efficient algorithms for computing the average, maximum, and minimum, as well as for building approximate contour maps [1]. These examples are representative of the work in the field [16]. The popularity of aggregates like the average and the maximum is no accident: they are both useful and easy to compute.

We emphasize that our attacks are not intended as a criticism of these systems, as these systems were not designed for security against compromised nodes. Nonetheless, this motivates the search for aggregation primitives with better security properties.

5. A THEORY OF RESILIENT AGGREGATION

After these attacks on certain aggregation operations, it is natural to ask for a way to reason about secure aggregation. We develop such a framework in this section. Two key landmarks are worth watching for in the exposition that follows:

1. We propose that aggregation primitives be viewed as nothing more than statistical estimators. This allows us to exploit the power of statistics and build on the existing literature on statistical estimation theory.

2. We argue that the resilient aggregration problem has close connections to the field of robust statistics, which was developed to deal with noisy and error-prone data. This correspondence looks fruitful, because it allows us to borrow clever ideas that have been developed in that field and adapt them to solve our problems.

Before we work out the details, it will be helpful to digress for a moment to give some essential mathematical background on the classical estimation theory.

Estimation theory. Succintly put, the estimation problem is this: Given a sequence of observations $x_1, \dots, x_n$ from a known parameterized distribution $p(X \mid \theta)$, where θ is a hidden parameter, the goal is to estimate θ as accurately as possible.

In more detail, let θ denote a parameter, whose distribution is not specified. A parameterized distribution $p(X \mid \theta)$ is a family of distributions, one for each possible value of θ. For instance, $\mathcal{N}(\theta, 1)$, the Gaussian distribution with mean θ and variance 1, is a distribution with parameter θ.

Next, let $X_1, \dots, X_n$ denote n random variables that are distributed according to $p(X \mid \theta)$ and that are conditionally independent given θ. In other words, imagine a referee secretly choosing a value θ, then making n independent draws from the distribution with this fixed value of θ as the common parameter and letting $X_1, \dots, X_n$ denote the result from the n draws.

An estimator is an algorithm $f : \mathbb{R}^n \to \mathbb{R}$, where $f(x_1, \dots, x_n)$ is intended as an estimate of some real-valued function of θ. For simplicity of exposition, in what follows we will assume that θ is real-valued and that we wish to estimate θ itself. Next, we define the random variable $\hat{\Theta} \overset{\text{def}}{=} f(X_1, \dots, X_n)$. Then, for our purposes[1], the relevant figure of merit is

$$\text{root-mean-square (r.m.s.) error (at } \theta\text{):} \quad \text{rms}(f) \overset{\text{def}}{=} \mathbb{E}[(\hat{\Theta} - \theta)^2 \mid \theta]^{1/2}$$

Note that $\text{rms}(f)$ is a function of θ, the underlying parameter; we compare functions pointwise. Also, an unbiased estimator is one for which $\mathbb{E}[\hat{\Theta} \mid \theta] = \theta$ for all θ.

The r.m.s. error is a good measure of the inaccuracy or "spread" of our estimator, and a reasonable intuition would be to think of the r.m.s. error as representing a "typical" value of the error term $|\hat{\Theta} - \theta|$. For an unbiased estimator, the r.m.s. error is exactly the standard deviation of the random variable $\hat{\Theta}$, so our usual intuitions about Gaussian distributions will often be helpful in thinking about the r.m.s. error metric. We usually rely on the r.m.s. error to characterize the quality of our estimator, and we often approximate the r.m.s. error by its asymptotic behavior as $n \to \infty$.

Application to aggregation. As others have noted before [10, 9], statistical estimation theory is a powerful way to think about data aggregation. One may think of the parameter θ as reflecting some interesting underlying quantity in the physical environment. However, θ cannot be directly observed. Instead, our only way to gain information on θ is by taking n sensor measurements, which are modelled by the random variables $X_1, \dots, X_n$. The aggregation function f can then be viewed as an estimator of θ.

For instance, θ might represent the average temperature across the New York City metropolitan area, and X_i the temperature reading at the ith sensor. It is natural that θ is a parameter (since the average temperature can vary from day to day) and that X_i is random but dependent on θ (since sensor readings are noisy, but correlated to θ). We might take $f(x_1, \dots, x_n) \overset{\text{def}}{=} (x_1 + \dots + x_n)/n$, so that our estimator is the average of all sensor readings. Then, if we have a model of the distribution $p(X_i \mid \theta)$, we can measure the quality of the average as an aggregation function by calculating the r.m.s. error of f as an estimator of θ. This seems like a reasonable view; when the user asks for the average of sensor readings, usually it is not that the average is of any inherent interest of its own right, but rather that this aggregate is an intuitive way to estimate some hidden parameter of interest.

Resilient estimators and resilient aggregation. A k-node attack A is an algorithm that is allowed to change up to k of the values $X_1, \dots, X_n$ before the estimator is applied. In particular, the attack A is specified by a function $\tau_A : \mathbb{R}^n \to \mathbb{R}^n$ with the property that the vectors x and $\tau_A(x)$ never differ at more than k positions.

We can define the r.m.s. error associated with A by

$$\text{rms}^*(f, A) \overset{\text{def}}{=} \mathbb{E}[(\hat{\Theta}^* - \theta)^2 \mid \theta]^{1/2} \quad \text{where } \hat{\Theta}^* \overset{\text{def}}{=} f(\tau_A(X_1, \dots, X_n)).$$

To explain, $\hat{\Theta}^*$ is a random variable that represents the aggregate calculated at the base station in the presence of the k-node attack A, and $\text{rms}^*(f, A)$ is a measure of the inaccuracy of the aggregate after A's intrusion. If $\text{rms}^*(f, A) \gg \text{rms}(f)$, then the attack has succeeded in noticeably affecting the operation of the sensor network. If $\text{rms}^*(f, A) \approx \text{rms}(f)$, the attack has had little or no effect. We define

$$\text{rms}^*(f, k) \overset{\text{def}}{=} \max\{\text{rms}^*(f, A) : A \text{ is a } k\text{-node attack}\},$$

so that $\text{rms}^*(f, k)$ denotes the r.m.s. error of the most powerful k-node attack possible. Note that $\text{rms}^*(f, 0) = \text{rms}(f)$.

Roughly speaking, we think of an aggregation function f as an instance of the *resilient aggregation* paradigm if $\text{rms}^*(f, k)$ grows slowly as a function of k. More precisely:

Definition 1. *We say that an aggregation function f is (k, α)-resilient (with respect to a parameterized distribution $p(X_i \mid \theta)$) if $\text{rms}^*(f, k) \le \alpha \cdot \text{rms}(f)$ for the estimator f.*

The intuition is that the (k, α)-resilient functions, for small values of α, are the ones that can be computed meaningfully and securely in the presence of up to k compromised or malicious nodes. This is justified by the following result.

Informal result 1. *Let f be an unbiased estimator, i.e., an aggregation function satisfying $\mathbb{E}[f(X_1, \dots, X_n) \mid \theta] = \theta$ for all θ. Let σ denote the standard deviation of $f(X_1, \dots, X_n)$*

[1] The relation to standard estimation theory is as follows. In classical estimation theory, there are three standard metrics:

$$\begin{aligned} \text{mean square error at } \theta: &\quad \text{MSE}(f) \overset{\text{def}}{=} \mathbb{E}[(\hat{\Theta} - \theta)^2 \mid \theta] \\ \text{variance at } \theta: &\quad \text{V}(f) \overset{\text{def}}{=} \text{Var}[\hat{\Theta} \mid \theta] \\ \text{bias at } \theta: &\quad \text{Bias}(f) \overset{\text{def}}{=} \mathbb{E}[\hat{\Theta} \mid \theta] - \theta \end{aligned}$$

A minimal-variance unbiased estimator is an estimator where $\text{MSE}(f)$ is minimal among all unbiased estimators. Note that $\text{MSE}(f)$, and the other metrics, are functions of θ. We compare functions pointwise when testing for minimality, i.e., $m \le m'$ iff $m(\theta) \le m'(\theta)$ for all θ. The name "minimal-variance unbiased estimator" is justified by the following basic identity: $\text{MSE}(f) = \text{V}(f) + \text{Bias}(f)^2$. Consequently, the unbiased estimator with minimal variance is also the unbiased estimator with minimal mean square error (or r.m.s. error, for that matter).

Notice that $\text{rms}(f) = \text{MSE}(f)^{1/2}$, so the r.m.s. error is closely connected to the mean square error. In this paper, we introduced the r.m.s. metric because it is intuitive, its units are more convenient, and it summarizes in one quantity the figure of merit we care most about.

in the absence of compromised nodes. If f is (k, α)-resilient, then it can be computed at a base station with bias on the order of $\pm\alpha \cdot \sigma$ or less. Conversely, if f is not (k, α)-resilient, then no matter how f is computed, the adversary can skew the result by more than $\pm\alpha \cdot \sigma$ at least some of the time.

PROOF (INFORMAL). Since f is an unbiased estimator, we have

$$\mathrm{rms}(f) = \mathbb{E}[(\hat{\Theta} - \theta)^2 \mid \theta]^{1/2} = \mathrm{StdDev}[\hat{\Theta} \mid \theta] = \sigma.$$

If f is (k, α)-resilient, then $\mathrm{rms}^*(f, k) \leq \alpha \cdot \mathrm{rms}(f) = \alpha \cdot \sigma$. In this case, the standard deviation of the aggregate (after being disturbed by compromised nodes) is at most $\alpha \cdot \sigma$, so the "typical" error term is on the order of $\alpha \cdot \sigma$. Conversely, if f is not (k, α)-resilient, then there is a k-node attack A so that $\mathrm{rms}^*(f, A) > \alpha \cdot \sigma$. For reasonable distributions, the magnitude of the actual observed error is at least $\mathrm{rms}^*(f, A)$ some non-negligible fraction of the time, from which the informal result follows. □

With the above results, we are now ready to propose an answer to the central question of secure aggregation: Which functionalities can be meaningfully computed, in the presence of compromised nodes?

> *Answer.* The aggregation functionalities that can be securely and meaningfully computed, in the presence of k compromised nodes, are exactly the functions that are (k, α)-resilient for some α that is not too large.

Attack models. The above definitions are framed in terms of what one might call *omniscient* attacks, i.e., attacks where the adversary first observes all n sensor readings, then chooses k sensors to corrupt and picks k bogus values for these sensors to report in place of their actual measurements. Hence omniscient attackers are limited only by the number of sensor readings they can change. In particular, omniscient attackers can dynamically choose which sensors to corrupt and can eavesdrop or predict all other sensor readings. This seems like an impractical threat model.

It would be more realistic to consider *myopic attacks*, where the adversary can only observe and affect measurements from some set of k sensors. In an adaptive attack, the attacker can choose a sensor to corrupt, learn the reading at that sensor, choose a replacement value based on his observation, and then choose another sensor to corrupt (possibly based on what he has learned up to this point) and repeat this up to k times. A weaker threat model is the static attack, where the attacker must choose k sensors in advance to corrupt; then the attacker learns those k readings and subsequently can choose k replacements values based on the observed values. Of course, in all cases we assume the parameterized distribution $p(X_i \mid \theta)$ is known in advance to the adversary. We could also consider many other refinements to the definition.

Myopic attackers are a more reasonable threat model than omniscient attackers, and myopic attacks correspond more closely to standard threat models used, for instance, in the cryptographic literature on secure multi-party computation. However, in all the examples treated in this paper, the difference in power between omniscient and myopic attacks turns out to be negligible. Because omniscient attacks permit cleaner definitions, for simplicity, we will stick with them.

Robust statistics. Robust statistics is the study of statistics when data is noisy or error-prone. Some authors suggest that robust statistics is generally concerned with two kinds of robustness:

1. Tolerating small errors in a large fraction of observations. For instance, perhaps many devices have small calibration errors or are otherwise slightly noisy.

2. Tolerating large errors in a small fraction of observations. For instance, if we imagine a scientist writing down observed values by hand, occasionally she may misplace the decimal point. A few gross errors like this should not render our statistical estimator useless. Thus, a classic task of robust statistics is to develop estimators that tolerate a small fraction of arbitrarily contaminated observations.

Resilient aggregation is concerned with the latter challenge; we wish to build aggregation operators that cannot be manipulated too much by a few malicious or compromised nodes.

There is a healthy literature on the field of robust statistics [6, 3, 13, 14]. For instance, it is well-known in the literature that the median is a more robust replacement for the average. Our work has been greatly inspired by many classic results in robust statistics.

One useful concept from robust statistics is the *breakdown point*, defined as[2]

$$\epsilon^* \overset{\text{def}}{=} \sup\{k/n : \mathrm{rms}^*(f, k) < \infty\}.$$

Strictly speaking, this definition gives the breakdown point $\epsilon^*(n)$ at n, but in practice we usually take the limit as $n \to \infty$. One small word of caution is in order: the breakdown point is only informative when the estimator is unbounded.

The relevance of the breakdown point to sensor networks is that the breakdown point indicates the fraction ϵ^* of nodes that can be captured before security breaks down. If an ϵ^* fraction of nodes are compromised, then the r.m.s. error becomes unbounded, and the adversary can drive the output of the aggregation operation to take on any value he would like. For instance, we can easily verify that the average has breakdown point $\epsilon^* = 0$, because any single compromised node can be used to skew the average by any desired amount. Consequently, the breakdown point is one measure of the security of an aggregation function against data spoofing attacks.

6. SATISFACTORY AND UNSATISFACTORY AGGREGATION FUNCTIONS

We next apply the theoretical framework sketched above to analyze the resilience of a number of natural aggregation functions. See Table 1 for a concise summary of our results.

Data model. Unfortunately, the resilience of a function f depends not only on the choice of f, but also on the parametrized distribution $p(X_i \mid \theta)$. In other words, we will need some model for how the data from the sensors is distributed. In practice, the exact distribution may vary from application to application, but to make the analysis

[2] Note: If $S \subseteq \mathbb{R}$ is a set, $\sup S$ is the smallest real number larger than or equal to every element of S. If S is finite, $\sup S$ is just the largest element in S.

aggregate (f)	error ($\mathrm{rms}(f)$)	resilience (α)	(ϵ^*)	security level
minimum	—	∞	0	insecure
maximum	—	∞	0	insecure
sum	$\sqrt{n} \cdot \sigma$	∞	0	insecure
average	$\sigma/\sqrt{n}$	∞	0	insecure
$[l, u]$-truncated average	$\sigma/\sqrt{n}$ or larger	$1 + (u - l)/\sigma \cdot k/\sqrt{n}$	—	problematic
5%-trimmed average	$(1 + \epsilon) \cdot \sigma/\sqrt{n}$	if $k < 0.05n$: $1 + 6.278\, k/n$	0.05	better
		if $k > 0.05n$: ∞		
median	$1.253\sigma/\sqrt{n}$	if $k < n/2$: $\sim \sqrt{1 + 0.101k^2}$	0.5	much better
		if $k > n/2$: ∞		
count	$\sqrt{n\theta(1 - \theta)}$	$1 + O(k/\sqrt{n})$	—	acceptable

Table 1: Summary of results. This table shows several possible aggregation functions, their root-mean-square error term in the absence of attacks, their resilience against k-node attack, and their breakdown point (ϵ^*). The smaller the resilience α is, the greater the security. Note that the 5%-trimmed average performs well as long as $k < 0.05n$, and the median is even better: it degrades gracefully for $k < 0.5n$. The other functions perform less well.

tractable, it seems reasonable to focus on two canonical distributions.

- For continuous data, we assume that the X_i come from n independent and identically distributed (i.i.d.) random variables with the Gaussian distribution $\mathcal{N}(\theta, \sigma^2)$ of mean θ and variance σ^2, where σ is fixed in advance and θ represents the hidden parameter to be estimated. It will be useful to let φ and Φ denote the probability density and cumulative probability functions for the standard normal distribution $\mathcal{N}(0, 1)$, i.e., $\varphi(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$ and $\Phi(x) = \int_{-\infty}^{x} \varphi(t)\, dt$.
- For 0/1-valued data, let us assume that the X_i come from n i.i.d. r.v.'s with the Bernoulli distribution, i.e., $X_i \sim \mathrm{Bernoulli}(\theta)$, or equivalently, $\Pr[X_i = 1 \mid \theta] = \theta$, where again θ represents the hidden parameter to be estimated.

It seems reasonable to expect that these two distributions will form a reasonable model of sensor readings in many practical applications.

Estimators of location. In practice, the most common goal of aggregation is to build an estimate of location, such as the mean or mode or median. A good estimate of location f should satisfy several properties. It should be *location equivariant*, so that $f(x_1 + c, \ldots, x_n + c) = f(x_1, \ldots, x_n) + c$ for any constant c. It should be *scale equivariant*, so that $f(cx_1, \ldots, cx_n) = c \cdot f(x_1, \ldots, x_n)$ for any constant c. Also, it should have permutation symmetry, so that permuting the inputs to f does not change its output, i.e., $f(x_{\pi(1)}, \ldots, x_{\pi(n)}) = f(x_1, \ldots, x_n)$ for all permutations π on $\{1, \ldots, n\}$. Finally, it should be robust. We will study the robustness of several popular estimates of location.

The average and sum. It is easy to verify that the sum $f(x_1, \ldots, x_n) = x_1 + \cdots + x_n$ is not $(1, \alpha)$-resilient for any constant α. For instance, for any fixed α, we can consider a 1-node attack[3] A that replaces the first reading x_1 with $x_1^* = x_1 + 2\alpha \cdot \mathrm{rms}(f)$; this attack has error term $\mathrm{rms}^*(f, A) = 2\alpha \cdot \mathrm{rms}(f)$, hence f is not $(1, \alpha)$-resilient. Similarly, the average is not $(1, \alpha)$-resilient for any constant α. Put another way, the average and sum have breakdown point $\epsilon^* = 0$. These points justify our claim in Section 3 that the average and sum cannot be meaningfully computed in the presence of a malicious sensor node.

The minimum and maximum. One can also show that the minimum is not $(1, \alpha)$-resilient for any constant α: simply consider the 1-node attack that replaces x_1 with $x_1^* = x_1 - 2\alpha \cdot \mathrm{rms}(\min)$. Likewise, the maximum is not $(1, \alpha)$-resilient for any α.

The count. In contrast, the count behaves much better. Take $f(x_1, \ldots, x_n) = x_1 + \cdots + x_n$ for $X_i \sim \mathrm{Bernoulli}(\theta)$ (i.i.d. r.v.'s). We have $\mathrm{rms}(f) = \sqrt{n\theta(1 - \theta)}$. For a k-node attack, a simple argument shows that $|\hat{\Theta}^* - \hat{\Theta}| \le k$ always, so $\mathrm{rms}^*(f, k) \le \sqrt{k^2 + \mathrm{rms}(f)^2} \le k + \mathrm{rms}(f)$.

Consequently, the count is (k, α)-resilient for $\alpha = 1 + k \cdot (n\theta(1 - \theta))^{-1/2}$. This is a rather slow-growing function of k, so for large n, the count appears to have good resilience against k-node attacks. In other words, the law of large numbers comes to our rescue: if we have a large field of sensors and we use the count as our aggregate, we can easily tolerate a small number of compromised nodes.

The median. The median is a safer replacement for the average. On inputs $x_1, \ldots, x_n$, let $x_{(1)}, \ldots, x_{(n)}$ denote the x_i-values placed in sorted order. If i is not an integer, let $x_{(i)}$ be short-hand for $\frac{1}{2} x_{(i-0.5)} + \frac{1}{2} x_{(i+0.5)}$. Then we may define the median as $f(x_1, \ldots, x_n) = \mathrm{med}_{1 \le i \le n}\, x_i = x_{(r)}$ where $r = (n + 1)/2$.

Note that a 1-node attack can only change the median to something between $x_{(r-1)}$ and $x_{(r+1)}$. If we have at least three readings, these two endpoints are sensor readings from uncompromised nodes. In general, after a k-node attack, the median will be in the interval $[x_{(r-k)}, x_{(r+k)}]$, and when $n > 2k$, the endpoints of this interval are readings from uncompromised nodes. In summary, we see that the attacker is not able to freely dictate the result of the aggregation

[3] These attacks actually do not require the assumption that $X_i \sim \mathcal{N}(\theta, \sigma^2)$; in general, it suffices merely that $\mathrm{rms}(f) < \infty$, e.g., that $f(X_1, \ldots, X_n)$ has finite variance and bias.

operation, but is constrained about how he can affect the computed median.

This intuition can be quantified. For $X_i \sim \mathcal{N}(\theta, \sigma^2)$, it is well-known that $\text{rms}(f) \approx 1.253 \cdot \sigma/\sqrt{n}$. We have calculated that, for $k \ll n$, $\text{rms}^*(f, k) = \alpha \cdot \text{rms}(f)$ where $\alpha \approx \sqrt{1 + 0.101k^2}$. Thus, f is (k, α)-resilient for this value of α. See the appendix for a justification. The interpretation is that, in the presence of an adversary, compromised nodes produce only a gradual increase in the error term. These figures also show that the "price of security" is not too high: in the absence of attack, the error term for the median is only slightly larger (by a factor of 1.253) than the error term for the average.

Also, the breakdown point of the median is $\epsilon^* = 1/2$, meaning that up to about half of the nodes may be compromised without a total breakdown of security.

7. TOOLS FOR ACHIEVING RESILIENT AGGREGATION

Next, we explore several general techniques for improving the resilience of our aggregation functions. The focus is on generality: the ideas presented here will be broadly applicable in many settings, and we characterize their benefits and limitations.

Truncation. One naive way to make an aggregation function more robust against spoofed sensor readings is to place upper and lower bounds on the acceptable range of a sensor reading. For instance, if we know that valid sensor readings will usually be in the interval $[l, u]$, then we can truncate every input to be within this range. Note, for instance, that the count can be viewed as a $[0, 1]$-truncated version of the sum. In general, given any base aggregate g, we can construct a truncated aggregator by applying g to the truncated data values.

More formally, let $\text{trunc}_{[l,u]}(x)$ be l if $x < l$, x if $l \le x \le u$, and u if $x > u$. To obtain a truncated replacement for the raw average, set

$$f(x_1, \ldots, x_n) = \frac{\text{trunc}_{[l,u]}(x_1) + \cdots + \text{trunc}_{[l,u]}(x_n)}{n}.$$

We have calculated that the $[l, u]$-truncated average is (k, α)-resilient for $\alpha \approx 1 + (u - l)k/(\sigma\sqrt{n})$. See the appendix for details.

The truncated aggregate is an improvement over the conventional aggregate, but it is not an entirely satisfactory solution. Wide intervals give the attacker a great deal of power, while narrow intervals reduce the utility of the sensor network. We can quantify this using the notion of the *dynamic range* D of an aggregate, defined as $D = (u - l)/\text{rms}(f)$. In the absence of attacks, D measures the number of possible outputs after aggregation that can be distinguished from each other. Thus, the observed aggregate conveys roughly $\lg D$ bits of information about the underlying parameter to be estimated. Also, let $D^* = (u - l)/\text{rms}^*(f, k)$ be the dynamic range in the presence of k-node attacks.

For the truncated mean, we have $D \approx (u - l)\sqrt{n}/\sigma$, so the truncated average is (k, α)-resilient for $\alpha \approx 1 + D \cdot k/n$. Also, $D^* \approx D/(1 + D \cdot k/n)$. Notice that when D is large compared to n/k, D^* is considerably smaller (indicating poor dynamic range under attack) and α is large (indicating poor security under attack). When D is small, the dynamic range is poor, meaning that the aggregate does not convey much information even in the absence of attack.

Now we are pinned between a rock and a hard place: to derive as much information as possible from our sensor network in the absence of attacks, we need D to be large; yet to make the truncated average as meaningful as possible in the presence of attacks, we need $D \cdot k/n$ to be small. This gives an unavoidable tradeoff between the resilience α and the dynamic range D. This tension suggests that we should keep looking for better techniques.

Trimming. A better choice is to ignore the highest 5% and lowest 5% (for instance) of the sensor readings, and then compute the aggregate on the remaining 90% of readings in the middle. This is known as the *trimmed mean* in the statistical literature. Intuitively, we might expect this to be fairly robust to compromised nodes, so long as no more than 5% of the sensors are compromised.

Let's work out the details. On inputs $x_1, \ldots, x_n$, let the symbols $x_{(1)}, \ldots, x_{(n)}$ represent the x_i-values in sorted order. Fix a security parameter ρ. Let g denote the underlying aggregation function. We construct a more resilient version of g by defining

$$f_\rho(x_1, \ldots, x_n) = g(x_{(\rho n)}, x_{(\rho n+1)}, \ldots, x_{(n+1-\rho n)}).$$

Trimming can be viewed as a principled, automated form of outlier elimination, where we always throw away the smallest and largest ρn observations, on the principle that they might be outliers. Assuming that $k < \rho n$, all an adversary can do is affect which subset of legitimate sensor readings are used as inputs to g; however, the adversary cannot control in any other way the inputs to g. Interestingly, the median is a special case of the above construction, obtained by taking the limit as $\rho \to 1/2$ from below. For these reasons, trimming looks intuitively promising.

The security argument can be formalized more carefully. We will analyze the case where g is the average, $g(w_1, \ldots, w_m) = (w_1 + \cdots + w_m)/m$. It turns out that $\text{rms}(f_\rho) \approx (1 + \epsilon_\rho) \cdot \sigma/\sqrt{n}$, for some constant $0 \le \epsilon_\rho \le 0.253$. We have computed $\text{rms}^*(f_\rho, k)$ for the case $k \ll \alpha n$, where ρ is arbitrary. See the appendix for details and for the general expression. For instance, when $\alpha = 0.05$, $\text{rms}(f_{0.05}) \approx (1 + \epsilon) \cdot \sigma/\sqrt{n}$ for some small ϵ, and $f_{0.05}$ is approximately $(k, 1 + 6.278\, k/n)$-resilient for $k < 0.05n$ and (k, ∞)-resilient for $k \ge 0.05n$.

In practice, the 5%-trimmed average only becomes more accurate than the median for large n (say, $n > 100$ or so). Therefore, in practice the 5%-trimmed average is unlikely to be any better than the median. However, one of the attractions of trimming is that it can be applied to other aggregates as well; for instance, the 5%-trimmed maximum might be a sensible replacement for the maximum.

Other estimates of location. Many other location estimators have been proposed in the robust statistics literature. Here we list a few that may be of interest.

The *shorth* is defined as the mean of the shortest subsample of $x_1, \ldots, x_n$ that contains $n/2$ of the observations; its breakdown point is $\epsilon^* = 1/2$. The *least median of squares* (LMS) is defined as the value y that minimizes $\text{med}_{1 \le i \le n}(x_i - y)^2$; it happens to be the same as the midpoint of the shortest subsample containing $n/2$ of the observations, and its breakdown point is $\epsilon^* = 1/2$. Both the shorth and the LMS converge at a rate like $n^{-1/3}$, which is abnormally slow: most other estimates, such as the average, converge at a rate like $n^{-1/2}$.

The *Hodges-Lehmann estimator* is given by $f(x_1,\ldots,x_n) = \mathrm{med}_{1\leq i<j\leq n}(x_i + x_j)/2$; it has breakdown point $\epsilon^* = 1 - \sqrt{1/2} \approx 0.293$, and its error term is only about $\pi/3 \approx 1.05$ times larger than the error term for the average. The interquartile mean is given by the mean of the first and third quartiles, i.e., $f(x_1,\ldots,x_n) = (x_{(r)} + x_{(n+1-r)})/2$, where $r = (n+1)/4$; it has breakdown point $\epsilon^* = 1/4$, and its error term is only about 1.25 times as large as the error term for the average.

The median and interquartile mean are instances of a class of functions known as of L-estimators. An aggregate is called a L-estimator if it can be expressed as a linear function of the order statistics of the sample, or in other words, f is an L-estimator if there exist constants $a_1,\ldots,a_n \in \mathbb{R}$ so that $f(x_1,\ldots,x_n) = a_1x_{(1)} + \cdots + a_nx_{(n)}$. The breakdown point is determined by the first (or last) non-zero coefficient $a_i \neq 0$; namely, $\epsilon^*(n) = \min\{i-1 : x_i \neq 0 \text{ or } x_{n+1-i} \neq 0\}/n$. There are other general classes of estimators as well, including M-estimators, GM-estimators, and R-estimators, but in general, the schemes we have described tend to be adequate, and so we omit further details about such more complicated estimators.

8. DISCUSSION

Pragmatic advice for implementors. The lesson of this work is that sensor network designers need to pay attention to security when implementing aggregation services. We have seen a number of simple measures that can be taken to improve security. Distilled into slogan form, our advice for improving the security of aggregation in sensor networks would be:

- Be aware of the threat of malicious data and of node capture attacks. Do a risk analysis before deploying aggregation services.

- Consider using resilient aggregation. For instance, try using the median or the trimmed average in place of the average.

When to use resilient aggregation. Resilient aggregation is best-suited to settings where there is plenty of redundancy in the data, so that we can cross-check sensor readings for consistency. Technology trends are tilting the scales in favor of large constellations of cheap, crude sensors, which is exactly where resilient aggregation is most appropriate, rather than small-scale deployments of a few expensive, precision-crafted devices. As a result, we expect this increasing degree of redundancy to make resilient aggregation applicable in an increasing variety of applications.

However, the techniques discussed in this paper are not always appropriate. Resilient aggregation is not a good choice in scenarios where we are looking for a needle in the haystack. For instance, multi-sensor fire alarm systems may need to trigger an alarm upon the first detection of smoke from any sensor, without waiting for corroboration from other sensors. Resilient aggregation cannot help in this case, and such systems must simply accept the possibility of successful attacks.

Open problems. One major limitation of this paper is that we did not consider in-network aggregation. There is strong evidence that performing aggregating inside the network is critical for achieving good performance [4]. Consequently, supporting in-network aggregation securely is an important open problem. Another challenge is to secure more complex aggregation operations, such as algorithms for tracking multiple objects, building contour maps, and so on.

Related work. The connection between aggregation and statistical estimation has been exploited before [10, 9], but as far as we know, no one has pointed out the relevance of robust statistics.

Song et al. also explore security for aggregation in sensor networks [12]. They, too, consider the possibility of corrupted sensors, albeit from a slightly different perspective: They consider how to reduce trust in the base station, in a scenario where a trusted outside user queries a sensor network. They focus on protocols for computing a few aggregation primitives: the median, min, max, average, and count of distinct elements. In contrast, we assume the base station is trusted, and we focus on classifying which functions can be meaningfully computed.

9. CONCLUSIONS

This paper gave a theoretical framework for evaluating data aggregation in sensor networks and its security against attack. As we saw, many of the conventional aggregates are unsuitable for use when some sensor nodes may be compromised. One of the main contributions of this paper was to provide a mathematical theory of resilient aggregation; we cast the problem in terms of statistical estimation theory, which gives a useful way to quantify the resilience of various aggregation operators and to justify our intuition in a principled way. Finally, we described a number of better methods for secure data aggregation. For instance, the median is a good summary statistic, and for general-purpose use, trimming appears to be a good way to strengthen the security of many aggregation primitives. The author's hope is that this paper will stimulate further work on the problem of secure data aggregation in sensor networks.

Acknowledgements

I thank Nikita Borisov and the anonymous reviewers for many helpful comments and suggestions.

10. REFERENCES

[1] A. Boulis, M.B. Srivastava, "Aggregation applications in resource-constrained distributed systems," Tech. report TM-UCLA-NESL-2001-11-002.

[2] Committee on Networked Systems of Embedded Computers, "Embedded, Everywhere: A Research Agenda for Networked Systems of Embedded Computers," National Academy Press, Wash., DC, 2001.

[3] F.R. Hampel, E.M. Ronchetti, P.J.Rousseeuw, W.A. Stahel, *Robust Statistics: The Approach Based on Influence Functions*, John Wiley & Sons, 1986.

[4] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, D. Ganesan, "Building Efficient Wireless Sensor Networks with Low-Level Naming," *SOSP 2001.*

[5] J.M. Hellerstein, W. Hong, S. Madden, K. Stanek, "Beyond Average: Towards Sophisticated Sensing with Queries," *IPSN 2003* (2nd International Workshop on Information Processing in Sensor Networks).

[6] P.J. Huber, *Robust Statistics*, John Wiley & Sons, 1981.

[7] C. Intanagonwiwat, R. Govindan, D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *MobiCom 2000*.

[8] S. Madden, M.J. Franklin, J.M. Hellerstein, W. Hong, "TAG: A Tiny AGgregation Service for Ad-Hoc Sensor Networks," *OSDI 2002*.

[9] R. Nowak, "Distributed EM Algorithms for Density Estimation and Clustering in Sensor Networks," *IEEE Transactions on Signal Processing*, Special Issue on Signal Processing in Networking, 2003.

[10] R. Nowak, U. Mitra, "Boundary Estimation in Sensor Networks: Theory and Methods," *2nd Intl. Workshop on Information Processing in Sensor Networks*, 2003.

[11] G.J. Pottie, W.J. Kaiser, "Wireless Integrated Sensor Networks," *Communications of the ACM*, 43(5):51–58, May 2000.

[12] B. Przydatek, D. Song, A. Perrig, "SIA: Secure Information Aggregation in Sensor Networks," *ACM SenSys 2003* (Conference on Embedded Networked Sensor Systems).

[13] P.J. Rousseeuw, A.M. Leroy, *Robust Regression and Outlier Detection*, John Wiley & Sons, 1987.

[14] R.G. Staudte, S.J. Sheather, *Robust Estimation and Testing*, John Wiley & Sons, 1990.

[15] Y. Yao, J.E. Gehrke, "Query Processing in Sensor Networks," *CIDR 2003* (First Biennial Conference on Innovative Data Systems Research), Jan 2003.

[16] J. Zhao, R. Govindan, D. Estrin, "Computing Aggregates for Monitoring Wireless Sensor Networks," *SNPA 2003* (1st IEEE Intl. Workshop on Sensor Network Protocols and Applications).

APPENDIX

A. FURTHER DETAILS

We elaborate here on some calculations from Sections 6 and 7.

The median. It is known that, when the X_i come from a Gaussian distribution $\mathcal{N}(\theta, \sigma^2)$, then the order statistics behave as follows:

$$x_{(pn)} \to \mathcal{N}\left(\theta + \frac{z_p\,\sigma}{\sqrt{n}},\ \frac{p(1-p)\sigma^2}{n\,\varphi(z_p)^2}\right) \qquad \text{as } n \to \infty,$$

where $z_p = \Phi^{-1}(p)$. Note that the median corresponds to the case $p = 1/2$. Hence, a good approximation for the standard deviation of the median is

$$\mathrm{rms}(f) \approx \frac{\sigma}{2\varphi(0)\sqrt{n}} = \sqrt{\frac{\pi}{2}} \cdot \frac{\sigma}{\sqrt{n}} \approx 1.253 \cdot \frac{\sigma}{\sqrt{n}}.$$

This is a factor of 1.253 larger than the corresponding error term for the average, so the median is only slightly worse than the average in the absence of any attack.

Likewise, we find that

$$x_{(r+k)} \to \mathcal{N}\left(\theta + \frac{z_p\,\sigma}{\sqrt{n}},\ \frac{p(1-p)\sigma^2}{n\,\varphi(z_p)^2}\right) \qquad \text{as } n \to \infty,$$

where $p = (r+k)/n$ and $r = (n+1)/2$. If $k \ll n$, then $z_p \approx k\,\varphi(0)$ and $\varphi(z_p) \approx \varphi(0)$, so

$$\begin{aligned} \mathbb{E}[(x_{(r+k)} - \theta)^2] &\approx \frac{\sigma^2}{4n\,\varphi(0)^2} + \frac{k^2\sigma^2\,\varphi(0)^2}{n} \\ &= \frac{\sigma^2}{n}\left(\frac{\pi}{2} + \frac{k^2}{2\pi}\right). \end{aligned}$$

A k-node attack can increase the median to at most $x_{(r+k)}$, so

$$\mathrm{rms}^*(f,k) \approx \sqrt{\frac{\pi}{2} + \frac{k^2}{2\pi}} \cdot \frac{\sigma}{\sqrt{n}} \approx \sqrt{1.571 + 0.159k^2} \cdot \frac{\sigma}{\sqrt{n}}.$$

Consequently, for $k \ll n$, the median is (k, α)-resilient for

$$\alpha \approx \sqrt{1 + k^2/\pi^2} \approx \sqrt{1 + 0.101k^2}.$$

The truncated average. An adversary who changes x_i to x_i^* can only increase or decrease the $[l,u]$-truncated average by at most the quantity $(u-l)/n$. Consequently,

$$|\hat{\Theta}^* - \hat{\Theta}| \leq \frac{k \cdot (u-l)}{n}$$

for every k-node attack A. (If $X_i \sim \mathcal{N}(\mu, \sigma^2)$ with $\mu \approx (l+u)/2$ and $\sigma \ll (u-l)/2$, then the above bound is too conservative by roughly a factor of two; however, it is simpler to ignore these small constant factors.) Consequently, $\mathrm{rms}^*(f,k) \leq \mathrm{rms}(f) + (u-l)k/n$. Also, $\mathrm{rms}(f)$ is roughly $\sigma/\sqrt{n}$ or larger. Thus, the $[l,u]$-truncated average is (k,α)-resilient for $\alpha \approx 1 + (u-l)k/(\sigma\sqrt{n})$.

The truncation technique applies generally to many forms of aggregation. For instance, it can also be applied to the sum, minimum, and maximum, with results comparable to those for the average. However, as discussed in the main body of this paper, truncation is not without problems.

The trimmed average. Let the notation be as in Section 7. We have $\sigma/\sqrt{n} \leq \mathrm{rms}(f_\rho) \leq 1.253 \cdot \sigma/\sqrt{n}$, since $\mathrm{rms}(f_\rho)$ is an increasing function of ρ, and as $\rho \to 0.5$, f_ρ approaches the median in the limit, while the case $\rho = 0$ is just the untrimmed average. Also, $\mathbb{E}[(x_{(n+1-\rho n)} - x_{(\rho n)})^2]^{1/2}$ is approximately $2z\sigma/\sqrt{n} + \sigma\sqrt{2\rho(1-\rho)/n}/\varphi(z)$ where $z = \Phi^{-1}(1-\rho)$. For instance, for $\rho = 0.05$, $z \approx 1.6448$, $\varphi(z) \approx 0.1031$, and thus typical values for the gap $x_{(0.95n)} - x_{(0.05n)}$ are roughly $3.290\sigma/\sqrt{n} + 2.988\sigma/\sqrt{n} = 6.278\sigma/\sqrt{n}$. A k-node attack can increase the trimmed sum $w_1 + \cdots + w_m$ by about k times the above gap; then we should divide by $(1-2\rho)n$ to obtain the change to the trimmed average $(w_1 + \cdots + w_m)/m$. In this way, for $k \ll \rho n$, we can estimate that $\mathrm{rms}^*(f_\rho, k) \approx \mathrm{rms}(f) + \sigma k(1-2\rho)^{-1}n^{-3/2}(2z + \sqrt{2\rho(1-\rho)}/\varphi(z))$ for $z = \Phi^{-1}(1-\rho)$. For instance, we have $\mathrm{rms}^*(f_{0.05}, k) \approx \mathrm{rms}(f) + 6.278\sigma k/n^{3/2}$ and $f_{0.05}$ is $(k, 1 + 6.278k/n)$-resilient, so long as $k < 0.05n$.