SIA: Secure Information Aggregation in Sensor Networks*

Bartosz Przydatek
Carnegie Mellon University
Pittsburgh, PA 15213, USA
bartosz@cmu.edu

Dawn Song
Carnegie Mellon University
Pittsburgh, PA 15213, USA
dawnsong@cmu.edu

Adrian Perrig
Carnegie Mellon University
Pittsburgh, PA 15213, USA
perrig@cmu.edu

ABSTRACT

Sensor networks promise viable solutions to many monitoring problems. However, the practical deployment of sensor networks faces many challenges imposed by real-world demands. Sensor nodes often have limited computation and communication resources and battery power. Moreover, in many applications sensors are deployed in open environments, and hence are vulnerable to physical attacks, potentially compromising the sensor's cryptographic keys.

One of the basic and indispensable functionalities of sensor networks is the ability to answer queries over the data acquired by the sensors. The resource constraints and security issues make designing mechanisms for information aggregation in large sensor networks particularly challenging.

In this paper, we propose a novel framework for secure information aggregation in large sensor networks. In our framework certain nodes in the sensor network, called aggregators, help aggregating information requested by a query, which substantially reduces the communication overhead. By constructing efficient random sampling mechanisms and interactive proofs, we enable the user to verify that the answer given by the aggregator is a good approximation of the true value even when the aggregator and a fraction of the sensor nodes are corrupted. In particular, we present efficient protocols for secure computation of the median and the average of the measurements, for the estimation of the network size, and for finding the minimum and maximum sensor reading. Our protocols require only sublinear communication between the aggregator and the user. To the best of our knowledge, this paper is the first on secure information aggregation in sensor networks that can handle a malicious aggregator and sensor nodes.

*This research was supported in part by the Center for Computer and Communications Security at Carnegie Mellon under grant DAAD19-02-1-0389 from the Army Research Office, by NSF Aladdin Center grants CCR-0122581 and CCR-0058982, and by gifts from Bosch. The views and conclusions contained here are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of NSF, ARO, Bosch, Carnegie Mellon University, or the U.S. Government or any of its agencies.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SenSys'03, November 5–7, 2003, Los Angeles, California, USA.
Copyright 2003 ACM 1-58113-707-9/03/0011 ...$5.00.

Categories and Subject Descriptors

C.2.2 [**Computer – Communication Networks**]: Distributed Systems; F.2 [**Analysis of Algorithms and Problem Complexity**]: Miscellaneous

General Terms

algorithms, reliability, security

Keywords

sensor networks, information aggregation, security, approximate interactive proofs

1. INTRODUCTION

Sensor networks are becoming increasingly popular to provide economical solutions to many challenging problems such as real-time traffic monitoring, wildfire tracking, wildlife monitoring, or building safety monitoring. In sensor networks, thousands of sensor nodes collectively monitor an area. These large sensor networks generate a substantial amount of data, yet the sensor nodes often have limited resources, such as computation power, memory, storage, communication, and most importantly, battery energy. The large scale of sensor networks and the resource constraints make it an important challenge to design and develop efficient information processing and aggregation techniques to make effective use of the data. Given a query, it may be unnecessary and inefficient to return all raw data collected from each sensor—instead, information should be processed and aggregated within the network and only processed and aggregated information is returned [13, 16]. In such a setting, certain nodes in the sensor network, called *aggregators*, collect the raw information from the sensors, process it locally, and reply to the aggregate queries of a remote user. However, information aggregation in sensor networks is made even more challenging by the fact that the sensor nodes and aggregators deployed in hostile environments may be compromised due to physical tampering. Therefore, the processing and aggregation mechanisms need to be resilient against attacks where the aggregator and a fraction of the sensor nodes may be compromised.

Previous work in data aggregation assumes that every node is honest [21, 10, 13, 16], with the exception of [15] (cf. Sec. 1.1). In this paper, we address the problem of how to enable *secure* information aggregation, such that the user accepts the data with high probability if the aggregated result is within a desired bound, but that the user detects cheating with high probability and rejects the result if it is outside of the bound.

An attacker can perform a wide variety of attacks. For example, once the attacker compromised the base station or the aggregators,

the attacker could perform a denial-of-service attack and stop responding to any queries. Since we assume that a compromised node is under the full control of the attacker, there is nothing to prevent the attacker from mounting such denial-of-service attacks. However, in this paper we focus on another type of attack that we call *stealthy attack*. In a stealthy attack, the attacker's goal is to make the user accept false aggregation results, which are significantly different from the true results determined by the measured values, while not being detected by the user. In particular, we want to guarantee that if the user accepts a reported aggregation result from the aggregators, then the reported result is "close" to the true aggregation value with high probability; otherwise, if the reported value is significantly different from the true value due to the misbehavior of the compromised aggregators and/or the sensors, the user will detect the corruption and reject the reported aggregation result with high probability. We stress that in the considered model the corrupted sensors and aggregators may deviate from the protocol in an arbitrarily malicious way, and our goal is to prevent the user from accepting incorrect results.

More precisely, we propose the approach of *aggregate-commit-prove*: in our setting, the aggregators not only perform the aggregation tasks, but also *prove* that they perform these tasks correctly. Specifically, to prevent the aggregators from cheating, we use cryptographic techniques of *commitments*, and construct efficient random sampling mechanisms and interactive proofs, which enable the user to verify that the answer given by the aggregators is a good approximation of the true value even when the aggregators and/or a fraction of the sensor nodes may be corrupted.

In this paper we present the following contributions:

- We introduce the problem of *secure* information aggregation in sensor networks, analyze the attack model and security requirements.
- We propose the *aggregate-commit-prove* framework for designing secure information aggregation protocols (Section 3).
- We put forward concrete protocols for securely computing the median (Section 4), securely finding the minimum and maximum values (Section 5), securely estimating (counting) the number of distinct elements (and the network size) (Section 6), and securely computing the average of measurements (Section 7). Our protocols require only sublinear communication overhead between the aggregator and the user.
- We propose the approach of *forward secure authentication* to ensure that even if an attacker corrupts a sensor node at a point in time, it will not be able to change any previous readings the sensor has recorded locally (Section 8).

1.1 Related work

Previous work in sensor network data aggregation has mainly focused on how to aggregate information assuming every node is honest [21, 10, 13, 16]. Hu and Evans have studied the problem of information aggregation if one node is compromised [15], but their protocol may be vulnerable if a parent and a child node in their hierarchy are compromised.

Ergün *et al.* [12] studied the problem of approximate interactive proofs, where a prover (the aggregator) proves to a verifier (the home server) that the input data has some property. However, in their model both the prover and the verifier can access the input data, and the task of the prover is to assist the verifier, so that the verifier doesn't have to read the entire input. Some of their solutions can be implemented directly in our model by simulating verifier's access to the input: whenever verifier should read a part of the input, he asks the prover to deliver the desired part. However, in many cases the *locations* of the desired parts should be hidden from the prover, hence a more expensive simulation is needed, e.g., using a private information retrieval protocol [7, 18].

2. PROBLEM STATEMENT: SECURE INFORMATION AGGREGATION

2.1 Problem Setting

We consider the setting where a large number of sensors are deployed in some area distant from a *home server*. Sensors perform measurements and the home server would like to query statistics of the measured values. However, sensors are usually simple, low-powered devices which can communicate only within small range of their location, and so they cannot report the measurements directly to the distant home server [17]. Thus, a resources-enhanced *base station* is often used as an intermediary between the home server and the sensor nodes.

We assume that certain nodes in the network would perform the aggregation task. In the rest of the paper, we refer to the node that performs the aggregation task the *aggregator*. The base station is a natural candidate to perform the aggregation task, due to its enhanced computation and communication power. However, the issue of deciding which nodes are the aggregators is out of the scope of this paper and we simply assume that there exist some aggregators in the network at a given time. Moreover, some sensor networks may have multiple aggregators (For example, in TAG [21], each non-leaf node is an aggregator). For simplicity, in most of this paper, we only consider the case of a single aggregator. Nevertheless, our techniques can be extended to multiple aggregators, as we discuss in Section 9.

2.2 Key Setup And Communication Model

We assume that each sensor has a unique identifier and shares a separate secret cryptographic key with the home server and with the aggregator [25]. The keys enable message authentication, and encryption if data confidentiality is required. Note that the home server and the aggregator do not need to store $O(n)$ keys [25] — instead each of them stores simply a master key K_B and K_A (for the home server and the aggregator, respectively), and each sensor node stores the shared keys $\mathrm{MAC}_{K_B}(\text{node ID})$ and $\mathrm{MAC}_{K_A}(\text{node ID})$, where MAC is a secure message authentication code that is used here as a pseudo-random function. A specific secure instantiation of MAC is the HMAC construction by Bellare *et al.* [5]. Thus, given a node ID, the home server (or the aggregator) can compute its shared key with the sensor node by using its master key and hence authenticate the sensor node's message.

Since some sensors may be corrupted (cf. Section 2.3), the network may become partitioned by the corrupted sensors, in which case some uncorrupted sensors are able to communicate with each other and/or with the aggregator only via routes through corrupted sensors. When this happens, the corrupted sensors could always play a denial-of-service attack and cut off the communication between two partitioned sensor networks and simply claim to one partition that the other partition is not reachable. In such a case an aggregator may at best be able to compute aggregated information for one partition. Therefore, for simplicity, we assume that the uncorrupted sensors form a connected component containing the aggregator, meaning that the set of uncorrupted sensors can reach each other via paths composed of only uncorrupted sensors.

We furthermore assume that the home server and base station have a mechanism to broadcast authentic messages (e.g. queries)

into the network, such that each sensor node can verify the authenticity of the message, for example using the TESLA broadcast authentication protocol [24, 25].

2.3 Attack Model And Security Goals

We consider a setting with a polynomially bounded attacker, which can corrupt some of the sensors as well as the aggregator. Actions of a corrupted device are totally determined by the adversary. In particular, the adversary can arbitrarily change the measured values reported by a corrupted sensor. However, we assume that the adversary can corrupt at most a (small) fraction of all the sensors.

An attacker can perform a wide variety of attacks. For example, a corrupted aggregator could report some significantly biased or fictive values (possibly totally independent of the measured values), instead of the real aggregates, and so provide the home server with false information. Since in many applications the information received by the home server provides a basis for critical decisions, false information could have catastrophic implications. However, we do not want to limit ourselves to just a few *specific* selected adversarial strategies. Instead, we assume that the adversary can misbehave in any *arbitrary* way, and the only limitations we put on the adversary are its computational resources (polynomial in the security parameter) and the fraction of nodes that it can corrupt. In particular, we assume the Byzantine fault model [20] where a compromised node is under the full control of the attacker.

In this setting, we focus on *stealthy attacks*, where the attacker's goal is to make the home server accept false aggregation results, which are significantly different from the true results determined by the measured values, while not being detected by the home server. In this context, denial-of-service attacks such as not responding to queries clearly indicates to the home server that something is wrong and therefore is not a stealthy attack.

Our security goal is to prevent stealthy attacks. In particular, we want to guarantee that if the home server accepts a reported aggregation result from the aggregators, then the reported result is "close" to the true aggregation value with high probability; otherwise, if the reported value is significantly different from the true value due to the misbehavior of the corrupted aggregators and/or the sensors, the home server will detect the corruption and reject the reported aggregation result with high probability.

2.4 Efficiency vs. Accuracy Tradeoff

The problems discussed in this paper have a straightforward (but unfortunately very inefficient) solution: the aggregator forwards to the home server all data and authentication information from each sensor. Given all the data, the home server can verify authenticity of each data item, and answer all the statistical queries locally.

However, we assume that the communication between the aggregator and the home server is expensive, hence the trivial solution of sending all the data is very inefficient, and our goal is to reduce the amount of communication between the the aggregator and the home server. On the other hand, communicating just the result of a query is in many cases (e.g., for count, min/max, average, or median queries) very efficient, but it does not give the guarantee of correctness. Moreover, for all the problems studied in this paper we can show that in order to prove that the reported aggregation result is exact (with zero probability of error), we need at least linear communication complexity (linear in the size of the network), i.e., we cannot do much better than sending all the data to the aggregator. If we are willing to accept (a small) non-zero probability of error, then theoretically general methods based on PCP techniques could be applied [2, 19]. However, such methods would be very inefficient in practice. Hence, in order to achieve practical sublinear communication complexity, we need to relax the accuracy requirements and accept approximative results.

Depending on the function f being computed by the aggregator, various notions of approximations are useful. Let f be a function of $a_1, \ldots, a_n$ into real numbers, and let $y = f(a_1, \ldots, a_n)$. We say that $\tilde{y}$ is a *multiplicative ε-approximation* of y (or just ε-*approximation*) if $(1-\varepsilon)y \leq \tilde{y} \leq (1+\varepsilon)y$. We say that $\tilde{y}$ is an *additive ε-approximation* of y if $y - \varepsilon \leq \tilde{y} \leq y + \varepsilon$.

The difference between y and $\tilde{y}$ can be caused by various factors:

(1) Some sensors may be compromised and report wrong values that will affect the aggregation result. If a corrupted sensor simply reports a wrong value,[1] it may be difficult to detect the misbehavior since such a detection may require application/semantics specific knowledge. However, depending on the aggregation function, assuming that at most a certain number of sensors are compromised, we can calculate the bound on how much deviation from the correct result these corrupted sensors can cause.

(2) In some scenarios, when the aggregator uses sampling techniques to calculate the aggregation result, the sampling technique will introduce some estimation error. We can bound the estimation error by adjusting the number of required samples.

(3) The aggregator may be compromised, and may try to cheat by reporting wrong aggregation values. Without security mechanisms, a corrupted aggregator can lie about the aggregation result and report wrong values that are very far from the true result.

Because the errors caused by the above three factors can be upper bounded additively, and computing the bounds for factors (1) and (2) is relatively straightforward, in the rest of the paper we mainly focus on describing new techniques preventing the attacks of the third kind (corrupted aggregator) and compute the corresponding bounds. In particular, we propose efficient interactive proofs for verification of the accuracy of the results reported by the aggregator.

In addition to the approximation error ε, which describes the quality of a reported result, we also use a parameter δ, which upper bounds the probability of not detecting a cheating aggregator (i.e., an aggregator reporting a result not within ε bounds). Formally, we call such a protocol an (ε, δ)-*approximation*, meaning that the protocol finds an ε-approximation with probability at least $1-\delta$, and runs in time polynomial in $1/\varepsilon$ and $1/(1-\delta)$. As long as δ is bounded away from 1 by some constant fraction, the actual value of δ is not essential, since by repeating a protocol $O(\log 1/\delta)$ times, we can make the probability arbitrarily small of being effectively cheated, assuming the independence of each trial.

2.5 Notation and Conventions

In the remainder of this paper, n denotes the number of sensors, $S_1, \ldots, S_n$, $\mathcal{A}$ denotes the aggregator, and $\mathcal{B}$ the home server. We consider scenarios where the values measured by the sensors are from some totally ordered set, and we denote by a_i the value reported by sensor S_i. In fact, without loss of generality, we assume that the values a_i are integers from $[m] = \{1, \ldots, m\}$.

For the complexity analysis, we assume that each element and each hash value can be accessed in 1 step, and sending it costs 1

[1] However, a faulty value with a correct authentication tag, since we assume that node compromise also results in key compromise.