Attack-Resilient Hierarchical Data Aggregation in Sensor Networks

Sankardas Roy
Center for Secure Information Systems
George Mason University
sroy1@gmu.edu

Sanjeev Setia
Department of Computer Science
George Mason University
setia@cs.gmu.edu

Sushil Jajodia
Center for Secure Information Systems
George Mason University
jajodia@gmu.edu

ABSTRACT

In a large sensor network, in-network data aggregation, i.e., combining partial results at intermediate nodes during message routing, significantly reduces the amount of communication and hence the energy consumed. Recently several researchers have proposed robust aggregation frameworks, which combine multi-path routing schemes with duplicate-insensitive algorithms, to accurately compute aggregates (e.g., Sum, Count, Average) in spite of message losses resulting from node and transmission failures. However, these aggregation frameworks have been designed without security in mind. Given the lack of hardware support for tamper-resistance and the unattended nature of sensor nodes, sensor networks are highly vulnerable to node compromises. We show that even if a few compromised nodes contribute false sub-aggregate values, this results in large errors in the aggregate computed at the root of the hierarchy. We present modifications to the aggregation algorithms that guard against such attacks, i.e., we present algorithms for resilient hierarchical data aggregation despite the presence of compromised nodes in the aggregation hierarchy. We evaluate the performance and costs of our approach via both analysis and simulation. Our results show that our approach is scalable and efficient.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; D.4.6 [**Operating Systems**]: Security and Protection—*Cryptographic controls*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection

General Terms

Algorithms, Design, Security

Keywords

Sensor Network Security, Data Aggregation, Hierarchical Aggregation, Synopsis Diffusion, Attack-Resilient

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.
SASN'06, October 30, 2006, Alexandria, Virginia, USA.
Copyright 2006 ACM 1-59593-554-1/06/0010 ...$5.00.

1. INTRODUCTION

In large sensor networks, computing aggregates *in-network*, i.e., combining partial results at intermediate nodes during message routing, significantly reduces the amount of communication and hence the energy consumed [11, 23]. An approach used by several data acquisition systems for sensor networks is to construct a spanning tree rooted at the querying node, and then perform in-network aggregation along the tree. Partial results propagate level-by-level up the tree, with each node awaiting messages from all its children before sending a new partial result to its parent.

Tree-based aggregation approaches, however, are not resilient to communication losses resulting from node and transmission failures, which are relatively common in sensor networks [11, 22, 23]. Because each communication failure loses an entire subtree of readings, a large fraction of sensor readings are potentially unaccounted for at the querying node, leading to a significant error in the query answer. To address this problem, researchers have proposed the use of multi-path routing techniques for forwarding sub-aggregates [11]. For aggregates such as Min and Max which are duplicate-insensitive, this approach provides a fault-tolerant solution. For duplicate-sensitive aggregates such as Count and Sum, however, multi-path routing leads to double-counting of sensor readings, resulting in an incorrect aggregate being computed.

Recently researchers [3, 12, 14] have presented clever algorithms to solve the double-counting problem associated with multi-path approaches. A robust and scalable aggregation framework called *Synopsis Diffusion* has been proposed for computing duplicate- sensitive aggregates such as Count and Sum. There are two primary elements of this approach - the use of a ring-based topology instead of a tree-based topology for organizing the nodes in the aggregation hierarchy, and the use of duplicate-insensitive algorithms for computing aggregates based on Flajolet and Martin's algorithm for counting distinct elements in a multi-set [5].

As presented, the Synopsis Diffusion aggregation framework does not include any provisions for security. Although we can easily prevent *unauthorized* nodes from launching attacks by augmenting the aggregation framework with authentication and encryption protocols [15, 24], *compromised* nodes present an entirely new set of security challenges. The lack of tamper-resistance and the unattended nature of many networks renders sensor nodes highly vulnerable to compromise. Standard authentication mechanisms cannot prevent a compromised node from launching attacks since all its keys are also compromised. In this paper, we present novel mechanisms for making the synopsis diffusion aggregation framework resilient to attacks launched by compromised nodes.

We present counter-measures against attacks in which a compromised node attempts to change the aggregate value computed at the root of the hierarchy. In particular, we focus on an attack in which

a sensor node that is not a leaf node in the aggregation hierarchy relays a false sub-aggregate value to its parents. We refer to this attack as *the falsified sub-aggregate attack*.

We show that if the synopsis diffusion approach is used to compute aggregates such as Count and Sum, an adversary can use the falsified sub-aggregate attack to cause the answer computed at the base station in response to a query to differ from the true value by an arbitrary amount. Moreover, we show that this attack can be launched with a high rate of success, even if only one or a small number of nodes are compromised.

We present an approach in which the synopsis diffusion aggregation frameork is augmented with a set of countermeasures that mitigate the effect of the falsified sub-aggregate attack. In our approach, a *subset* of the total number of nodes in the network include an authentication code (MAC) along with their response to a query. These MACs are propagated to the base station along with the partial results that are computed at each level in the hierarchy. By verifying these MACs, the base station can estimate the accuracy of the final aggregate value it computes, and can filter out the effect of any false sub-aggregates contributed by compromised nodes. Thus, our approach can be used in conjunction with synopsis diffusion to compute basic aggregates such as Count and Sum despite the presence of compromised nodes in the aggregation hierarchy.

The communication overhead of our approach depends upon the number of contributing nodes which send a MAC to the base station. We evaluate the performance and costs of our approach via both analysis and simulation. We show that our approach is scalable since the number of contributing nodes (and hence the average communication overhead) do not increase with network size. To further reduce the communication overhead, we describe a variation of our basic approach that trades communication costs for latency.

2. BACKGROUND: SYNOPSIS DIFFUSION FOR ROBUST AGGREGATION

In this section, we provide a brief overview of the synopsis diffusion approach for robust aggregation [3, 14]. Figure 1 illustrates how the synopsis diffusion approach uses a rings topology for aggregation.

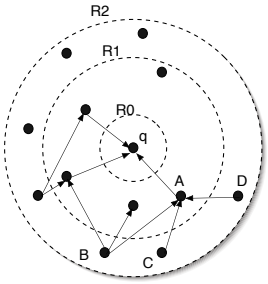

Figure 1: Synopsis Diffusion over a rings topology

In the query distribution phase, nodes form a set of rings around the querying node q based on their distance in hops from q. During the subsequent query aggregation period, starting in the outermost ring each node generates a local synopsis $s = SG(v)$ where v is the sensor reading relevant to the query, and broadcasts it. ($SG()$ is the synopsis generation function.) A node in ring R_i will receive broadcasts from all the nodes in its range in ring R_{i+1}. It will then combine its own local synopsis with the synopses received from its children using a synopsis fusion function $SF()$, and then broadcast the updated synopsis. Thus, the fused synopses propagate level-by-level until they reach the querying node, who first combines the received synopses with its local synopsis using $SF()$ and then uses the synopsis evaluation function $SE()$ to translate the final synopsis to the answer to the query.

The functions $SG()$, $SF()$, and $SE()$ depend upon the target aggregation function, e.g. Count, Sum, etc. We now describe the duplicate-insensitive synopsis diffusion algorithms for the Count aggregate, i.e., the total number of nodes in the sensor network, and the Sum aggregate, i.e., the sum of the sensor readings of the nodes in the network. These algorithms are based on Flajolet and Martin's well-known probablistic algorithm for counting the number of distinct elements in a multi-set[5].

2.1 COUNT

In this algorithm, each node generates a local synopsis which is a bit vector ls of length $k > \log n$, where n is an upper bound on the nodes in the network. To generate its local synopsis, each node executes the function $CT(X,k)$ given below, where X is the node's identifier and k is the length of ls in bits. $CT()$ can be interpreted as a coin-tossing experiment (with a cryptographic hash function $h()$, modeled as a random oracle whose output is 0 or 1, simulating a fair coin-toss), which returns the number of coin tosses until the first heads occurs or $k+1$ if k tosses have occurred with no heads occurring. In the local synopsis ls of node X, a single bit i is set to 1, where i is the output of $CT(X,k)$. Thus ls is a bitmap of the form $0^{i-1}1\cdots$ with probability 2^{-i}.

Algorithm 1 $CT(X,k)$

```
i=1;
while i < k+1 AND h(X,i) = 0 do
    i = i+1;
end while
return i;
```

The synopsis fusion function $SF()$ is simply the bitwise Boolean OR of the synopses being combined. Each node fuses its local synopsis ls with the synopses it receives from its children by computing the bit-wise OR of all the synopses. Let S denote the final synopsis computed by the querying node by combining all the synopses received from its children and its local synopsis. We observe that S will be a bitmap of length k of the form $1^{r-1}0\cdots$. The querying node can estimate Count from S via the synopsis evaluation function $SE()$: if r is the lowest-order bit in S that is 0, the count of nodes in the network is $2^{r-1}/0.7735$. The synopsis evaluation function $SE()$ is based on Property 2 below. Intuitively, the number of sensor nodes is proportional to 2^{r-1} since no node has set the rth bit while computing $CT(X,k)$.

We now present a few important properties of the final synopsis S computed at the querying node that have been derived in [5, 3], and that we will find useful in the rest of this paper. Let $S[i], 1 \le i \le k$ denote the ith bit of S, where bits are numbered starting at the left.

Property 1 For $i < \log_2 n - 2\log_2\log_2 n$, $S[i] = 1$ with probability ≈ 1. For $i \ge \frac{3}{2}\log_2 n$, $S[i] = 0$ with probability ≈ 1.

This result implies that for a network of n nodes, we expect that S has an initial prefix of all ones and a suffix of all zeros, while only the bits around $S[\log_2 n]$ exhibit much variation. This provides an estimate of the number of bits, k, required for a node's local synopsis. In practice, $k = \log_2 n + 4$ bits are sufficient to represent S with high probability [5]. This result also indicates that the length of the prefix of all ones in S can be used to estimate n. Let $r =$

$\min\{i|S[i]=0\}$, i.e., r is the location of the leftmost zero in S. Then $R = r-1$ is a random variable representing the length of the prefix of all ones in the sketch. The following results hold for R.

Property 2 The expected value of R, $E(R) \approx \log_2(\phi n)$ where the constant ϕ is approximately 0.7735.

This result implies that R can be used for an unbiased estimator of $\log_2(\phi n)$, and it is the basis for the synopsis evaluation function $SE()$ which estimates n as $2^R/\phi$.

Property 3 The variance of R, denoted as $\sigma^2_{R_n}$, satisfies

$$\sigma^2_{R_n} = \sigma^2_{R_\infty} + Q(\log_2 n) + o(1),$$

where constant σ_{R_∞} is approximately 1.1213 and $Q(x)$ is a periodic function with mean value 0 and period 1.

This property implies that the standard deviation of R is approximately 1.1213, i.e., the estimates of n derived from R will often be off by a factor of two or more in either direction. To reduce the standard deviation of R, Flajolet et al [5] proposed an algorithm named PCSA, where m synopses are computed in parallel and the new estimator ($\bar{R}$) is the average of all individual R's of these synopses. For PCSA, the standard error in the estimate of n, i.e., σ_n/n, is equal to $0.78/\sqrt{m}$ [5].

Property 4 In a network of n nodes, the expected number of nodes that will have the ith bit of their local synopsis $ls[i] = 1$ is $n/2^i$. This result implies that the expected number of nodes that contribute a 1 to the ith bit of S and the bits to the right of the ith bit in S (i.e., bits j, where $i \le j \le k$) is $n/2^{i-1}$.

2.2 SUM

Considine et al. [3] extended the Count algorithm described above for computing the Sum aggregate. The synopsis generation function $SG()$ for Sum is a modification of that for Count while the fusion function $SF()$ and the evaluation function $SE()$ for Sum are identical to those for Count.

To generate its local synopsis for a sensor reading v, a node X invokes the function $CT()$ v times[1] and ORs the results. As a result, the local synopsis of a node is a bitmap of length $k = \log_2 u_s + 4$ where u_s is an upper bound on the value of Sum aggregate. Unlike the local synopsis of a node for Count, more than one bit in the local synopsis of a node for Sum will be equal to 1. Count can be considered as a special case of Sum where each node's sensor reading is equal to one unit.

Considine et al. [3] proposed an optimized version of $SG()$ for Sum to make it suitable for a low-end sensor node, even if the sensed value v is high. Moreover, they showed that Properties 1-4 described above for Count also hold for Sum (with appropriate modifications). Similarly, as in the case of Count, the PCSA algorithm can be used to reduce the standard deviation of the estimate for Sum.

3. ATTACKS ON SYNOPSIS DIFFUSION

The Synopsis Diffusion aggregation framework does not include any provisions for security; as a result, it is vulnerable to many attacks that can be launched by unauthorized or compromised nodes. To prevent *unauthorized nodes* from eavesdropping on or participating in communications between legitimate nodes, we can augment the aggregation framework with any one of several recently proposed authentication and encryption protocols [15, 24]. However, *compromised nodes* pose an entirely new set of security challenges.

Sensor nodes are often deployed in unattended environments, so they are vulnerable to physical tampering. Current sensor nodes lack hardware support for tamper-resistance. Consequently, it is relatively easy for an adversary to compromise a node without being detected. The adversary can obtain confidential information (e.g., cryptographic keys) from the compromised sensor and reprogram it with malicious code.

A compromised node can be used to launch multiple attacks against the sensor application. These attacks include jamming at physical or link layer, other denial of service attacks like flooding, route disruption, message dropping, message modification, false data injection and many others. Standard authentication mechanisms cannot prevent these insider attacks since the adversary knows all the keying material possessed by the compromised nodes.

In this paper, we focus on defending against an important subclass of these insider attacks which can potentially corrupt the final result of the aggregation query. Below we describe these attacks in the context of the Count and Sum aggregates.

A compromised node M can corrupt the aggregate value computed at the root (i.e., the sink) of the hierarchical aggregation framework in three ways. First, M can simply drop aggregation messages that it is supposed to relay towards the sink. If M is located at a relatively high position in the aggregation hierarchy, this has the effect of omitting a large fraction of the set of sensor readings being aggregated. Second, M can falsify its own sensor reading with the goal of influencing the aggregate value. Third, M can falsify the sub-aggregate which M is supposed to compute based on the messages received from M's child nodes.

The effect of the first attack in which a node intentionally drops aggregation messages is no different from the effect of transmission and node failures, which are common in sensor networks [7]. The synopsis diffusion approach employs multi-path routing for addressing these failures, and thus it also addresses message losses due to compromised nodes [3, 12, 14]. We refer to the second attack in which a sensor intentionally falsifies its own reading as *the falsified local value attack*. This attack is similar to the behavior of nodes with faulty sensors and can be addressed by well-studied approaches for fault tolerance such as majority voting and reputation-based frameworks [10, 6]. The third attack, however, in which a node falsifies the aggregate value it is relaying to its parents in the hierarchy is much more difficult to address, and is the main focus of this paper. We refer to this attack as *the falsified sub-aggregate attack*.

The Falsified Sub-Aggregate Attack Since the sink estimates the aggregate based on the lowest-order bit r that is 0 in the final fused synopsis, a compromised node would need to falsify its own fused synopsis such that it would affect the value of r. It can accomplish this quite easily by simply inserting ones in one or more bits in positions j, where $r \le j \le k$, in its own fused synopsis which it broadcasts to its parents. Note that the compromised node does not need to know the true value of r; it can simply set some higher-order bits to 1 in the hope that this will affect the value of r computed by the sink. Since the synopsis fusion function is a bitwise Boolean OR, the resulting synopsis computed at the sink will reflect the contributions of the compromised node.

Let r' be the lowest-order bit that is 0 in the corrupted synopsis, whereas r is the lowest-order bit that is 0 in the correct synopsis. Then the sink's estimate of the aggregate will be larger than the correct estimate by a factor of $2^{r'-r}$. It is easy to see that, with the above technique, the compromised node can inject a large amount of error in the final estimate of the sink.

We also observe that even a single node can launch this attack with a high rate of success because the use of multi-path routing in the synopsis diffusion approach makes it highly likely that the falsified synopsis will be propagated to the base station. If p is the

[1]Each sensor reading is assumed to be an integer

packet loss rate and if each node has κ parents in the aggregation hierarchy then the probability of success for this attack is $(1-p^{\kappa})^h$, where the compromised node is h hops away from the sink. As an example, if $p = 0.2$, $\kappa = 3$, and $h = 5$ then the probability that the attack will succeed is 96%.

On the other hand, it is very hard to launch an attack which results in the aggregate estimated at the sink being lower than the true estimate. This is because setting a bit in the falsified synopsis to 0 has no effect if there is another node X that contributes a 1 to the same position in the fused synopsis. To make this attack a success the attacker has to compromise all the possible paths from node X to the sink so that X's 1 cannot reach the sink, which is hard to achieve. If there is more than one node which contributes to the same bit then it is even harder. As an example, in Count algorithm, half of the nodes are likely to contribute to the leftmost bit of the synopsis, one-fourth nodes of contribute to the second bit, and so on. There are bits in the synopsis to which only one or two nodes contribute but it is very hard to predict in advance which nodes will be contributing to these particular bits if the sink broadcasts along the query request a random seed to be used with the hash function in the synopsis generation phase. Hence, we can safely assume that this attack is extremely difficult to launch. In the rest of this paper, we restrict our discussion to the previous attack where the goal of the attacker is only to increase the estimate.

4. PROBLEM DESCRIPTION & ASSUMPTIONS

4.1 Problem Description

In a sensor network where some fraction of the nodes are potentially compromised, there are three sources that contribute to the error in the sink's estimate of the aggregate being computed: (i) error due to packet losses, (ii) error due to the approximation algorithm used, e.g., Flajolet and Martin's probabilistic algorithm [5], and (iii) error injected by compromised nodes.

The first two types of error are already addressed by the synposis diffusion aggregation framework. Our paper is complementary to this previous work; our objective is to filter out the third type of error. In particular, we aim to make the synopsis diffusion approach resilient to *the falsified local value attack* and *the falsified sub-aggregate attack*, i.e., to enable the sink to get the "true" estimate of the aggregate being computed despite the presence of compromised nodes in the aggregation hierarchy. By "true" estimate we mean the estimate of the aggregate which the sink would compute if there were no compromised nodes.

4.2 Assumptions

We now discuss our assumptions with respect to the sensor network and the adversary.

System Assumptions We assume that the base station is located at the center of the sensor network, and nodes are deployed around the base station. However, our approach for attack-resilient aggregation does not depend upon this assumption. We assume that sensor nodes are similar to the current generation of sensor nodes, e.g., Mica2 motes [13], in their computational and communication capabilities and power resources, while the sink is a laptop class device supplied with long-lasting power.

We assume that the sink has an estimate of the upper bound on the value of the Count aggregate. If the sink does not have any further knowledge, the upper bound of Count can be set to the total number of nodes deployed. We also assume that there exists an upper bound on the value of a sensor reading. The upper bound of Sum can be conservatively set to be equal to product of the upper bound of Count and the upper bound of a sensor reading. Previous works on the synopsis diffusion approach [3, 14] have made the same assumptions regarding the upper bounds for Count and Sum; these bounds provide an estimate of the length of the synopsis.

Security Assumptions We assume that the sink cannot be compromised and it uses a protocol such as μTesla [15]) to authenticate its broadcast messages. We also assume that each node shares a pairwise key with the sink, which is used to authenticate the messages it sends to the sink.

We assume that the adversary can compromise sensor nodes without being detected. If a node is compromised, all the information it holds will also be compromised. We use a Byzantine fault model, where the adversary can inject malicious messages into the network through the compromised nodes. We conservatively assume that all compromised nodes can collude, or are under the control of a single attacker.

Notations The following notations are used in the description of our attack-resilient aggregation algorithms.

- BS refers to the base station, i.e., the sink. X is the identifier of a the sensor node whereas M represents a compromised node.
- K_X is the pair-wise key X shares with the sink.
- $m1|m2$ denotes the concatenation of two message fields $m1$ and $m2$.
- $MAC(K,m)$ is the message authentication code (MAC) of the message m generated using the key K.
- $X \rightarrow Y : m$ denotes a one-hop delivery of message m from X to Y, while $X \rightarrow * : m$ denotes that X broadcasts message m to all of its one-hop neighbors, and $X \rightarrow\rightarrow * : m$ denotes that X broadcasts message m to all nodes in the network.

5. ATTACK-RESILIENT AGGREGATION: THE BASIC APPROACH

In this section, we present an attack-resilient approach for computing the Count and Sum aggregates. In this approach we assume that the BS has an estimate of the lower bound and the upper bound of the aggregates. We will see that this approach is scalable only if the ratio of the upper bound to the lower bound is small. Despite this limitation, we discuss this approach in detail because it provides the background and motivation for our extended approach, which is discussed in Section 6. We first present the main idea underlying the basic approach and then present the detailed protocol for securing Count and Sum.

5.1 The Main Idea

In our approach, nodes execute the synopsis diffusion aggregation algorithm as specified in [3, 14]. However, a subset of the nodes include along with their synopses a message authentication code (MAC) that can be used by the sink to verify the validity of their contribution to the aggregate function.

The key observations behind the design of our approach are that

- In order to derive the correct estimate from the final synopsis (say S) computed at the sink, we need only to figure out the correct lowest order bit (say r) in S that is 0.
- The number of nodes contributing a 1 to bit j decreases exponentially as we move from the lowest order bit ($j = 1$) to higher order bits of the synopsis. For example, in the case

of Count, on average, half the nodes in the network will contribute [2] to the leftmost bit of the synopsis, one-fourth of the nodes contribute to the second bit of the synposis, and so on.

Thus, we expect that only a small number of nodes will contribute to the bits around the expected value of r. Each such node includes along with its response to an aggregation query a MAC computed using a pairwise key shared exclusively with sink. We demonstrate that these MACs enable the sink to filter out the contributions of the falsified sub-aggregates injected by the compromised nodes to the final aggregate.

For our scheme to work, two issues need to be addressed. First, since the the value of r is not known before the execution of the query, we need to specify a criterion whereby a node can determine if it needs to include a MAC along with its synopsis. Second, this criterion should be designed so that the number of such nodes who include a MAC is minimized.

In our basic approach, we assume that the BS has an estimate of the lower bound and the upper bound of Count which are denoted by l_c and u_c respectively. Based upon these bounds, the BS knows that bit r will lie between a and b, which are the bit positions in the synopsis S corresponding to l_c and u_c respectively, i.e., $a = \lceil \log_2(\phi l_c) \rceil$ and $b = \lfloor \log_2(\phi u_c) \rfloor$ (by Property 2 in Section 2). Thus, there is no need for the BS to verify the bits to the left of a; only nodes contributing to bits in the range a to b need to prove to the BS that their contribution to the synopsis S is valid. We refer to the collection of bits in the range a to b in synopsis S as the *synopsis-edge* as shown in Figure 2. It is easy to see that the length of the synopsis-edge is $(\lfloor log_2(\frac{u_c}{l_c}) \rfloor + 1)$ bits. If we denote the number of nodes contributing to the synopsis-edge by η, then, by Property 4 in Section 2, $\eta \leq (\frac{u_c}{2^a} + \ldots + \frac{u_c}{2^b}) \approx \frac{1}{\phi} \cdot (\frac{2u_c}{l_c} - 1)$.

The upper bound for Count (u_c) can be set to the total number of nodes deployed. The lower bound for Count (l_c) can be guessed depending on the the energy reserve of the sensor nodes and rate of energy expenditure. As an example, if 2000 nodes are deployed then $u_c = 2000$ and $l_c = 1000$ may be a safe estimate at the time of the Count query's execution. For this example, the length of the *synopsis-edge* is $\frac{u_c}{l_c} = 2$ and the expected number of nodes contributing to synopsis-edge is less than 3.87.

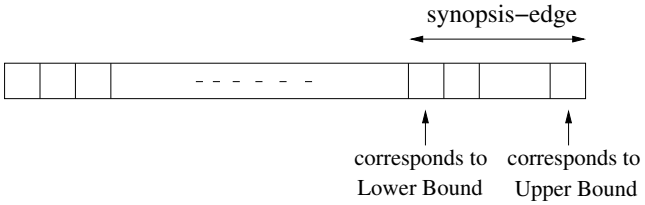

Figure 2: Securing Count synopsis. To securely compute Count synopsis, the base station needs to verify only bits in the synopsis-edge.

For the ease of presentation, we present the basic approach assuming that only one synopsis is computed. We can easily extend this approach to compute m synopses in parallel as in algorithm PCSA.

5.2 Securing Count

To compute the Count aggregate securely, we extend the original Count algorithm discussed in Section 2 as follows. For the sake of completeness, we first briefly describe the query dissemination phase, and then we present the aggregation procedure in detail.

In the query dissemination phase, the BS broadcasts the name of the aggregation function, a random number (*Seed*) and the bit positions of the start and the end of the synopsis-edge, which are specified by a and b respectively. Each node will use the random number, *Seed*, as an input to the hash function in the synopsis generation procedure. In more concrete terms, a query packet that the BS broadcasts is as follows:

$$BS \rightarrow\rightarrow * : F_{agg}, Seed, a, b, s, t, h$$

where F_{agg} is the name of the aggregation function (i.e. 'Count'), s denotes the time when the aggregation phase will start, t represents the duration of one *round* i.e. $t = \frac{T}{h}$, where h is the total number of hops and T is the duration of the aggregation phase (also called *epoch*). Note that, as in the original Count algorithm discussed in Section 2, the epoch is sub-divided into a series of *rounds*, one for each hop, starting from the farthest hop. μTesla [15] can be used for authenticating the broadcast packet.

In the aggregation phase, each node executes the synopsis generation function $SG()$ and the synopsis fusion function $SF()$ for Count as discussed in Section 2. In addition, each node checks whether it contributes to the synopsis-edge, and if so, it generates a MAC and forwards the MAC along with its fused synopsis. Specifically, if node X contributes to bit i in the synopsis-edge, it generates a MAC, $M = MAC(K_X, m)$ over the message m whose format is $[X|i|Seed]$, where *Seed* is the random number which was disseminated in the query distribution phase. Each node X forwards to its parents its fused synopsis along with the set of MACs ($\mathcal{M}$) it received from its child nodes and its own MAC if it generated one. The format of the message a node X forwards to its parents is as follows:

$$X \rightarrow * : S_l | \mathcal{M} ,$$

where S_l is the fused synopsis computed by X. If the message does not fit into one packet, node X breaks it into several packets and forwards them. In Appendix A, we formally describe the algorithm (SecureCount) executed by each node in response to an aggregation query.

After the BS receives the MACs, it checks their validity. In particular, for each message and MAC pair $[m|MAC(K_X, m)]$ where m is $[X|i|Seed]$, the BS executes the synopsis generation function $SG()$ of X and verifies whether node X really contributes to bit i in the synopsis-edge, and then checks whether the attached MAC is valid. If any of these tests fail, the corresponding MAC is discarded.

After this verification step, the BS checks whether it has received at least one valid MAC for each bit in the synopsis-edge. The bits in the synopsis-edge for which the BS has not received a valid MAC are reset to 0. The bits at positions to the left of the synopsis-edge are set to 1. Finally, the BS computes the Count using the synopsis evaluation function $SE()$.

Security Analysis The security of our approach follows from two facts:

- The sink can independently verify the output of $SG()$ for a particular node X. This is because the output of $SG()$ depends only upon the node id X, and the random seed included in the query message.
- Each bit that is set to 1 in the synopsis edge has an associated MAC that can be verified by the sink. This MAC is computed using a pairwise key that is known only to the contributing node and the sink, thus the MAC cannot be fabricated by an attacker (as long as it is reasonably long.)

[2]For convenience, henceforth, we say that a node "contributes" to a position j in the synopsis S if bit j in its local synopsis is 1.

Although a compromised node can falsely set some bits in its fused synopsis and forward false MACs corresponding to those bits, the sink will be able to discard any false MACs. This implies that the attacker cannot falsely increase the Count. On the other hand, the attacker may attempt to decrease the Count by dropping a genuine MAC (or by corrupting a genuine MAC) sent by a contributing node, but the genuine MAC is likely to reach BS via an alternate path. If BS receives at least one valid MAC for each 1 bit in the synopsis-edge, then BS obtains the true estimate of Count as discussed below.

As discussed in Section 2, the synopsis diffusion approach uses a multi-path routing scheme to reduce the aggregation error due to packet losses resulting from node and link failures. The effect of packets being dropped by compromised nodes is simply to increase the overall packet loss rate, and this can be countered by an appropriate choice of κ, the number of parents of a node in the synopsis diffusion ring-based aggregation hierarchy. Specifically, if each node has more than κ parents, the total number of rings in the rings topology is h, and if the probability of a node being compromised is p then, on average, a contributing node's MAC will reach the BS with probability q, where

$$q \geq \frac{1}{h} \cdot \sum_{j=1}^{h} (1-p^{\kappa})^{j}$$

Here we have assumed that the contributing nodes are uniformly distributed over the rings in the hierarchy. As an example, if $p = 0.05$, $\kappa = 3$, and $h = 10$ then q is greater than 0.999, i.e., the impact of the compromised nodes on the communication error is negligible.

We also note that while deriving q we assumed that there is only one node which contributes to a particular bit in the synopsis. In reality, the expected number of nodes contributing to a bit increases exponentially as we move from the Rth bit, where R is the the length of the prefix of all ones in the synopsis S, to the lower-order bits, thereby increasing the probability that at least one MAC corresponding to a bit position reaches the sink.

Computation and Communication Overhead Each contributing node computes one MAC. The expected number of contributing nodes is $\eta = \frac{1}{\phi} \cdot (\frac{2u_c}{l_c} - 1)$, which is independent of network size. Thus, only a subset of nodes will incur any computational overhead. With respect to communication overhead, the maximum number of MACs that any node will need to forward is η. Thus this approach is scalable, and can be used in large-scale sensor networks as long as the ratio u_c/l_c is reasonably small.

5.3 Securing SUM

We can extend the approach used for making the Count aggregate resilient to compromised nodes to the Sum aggregate. To derive the synopsis-edge for Sum we need to assume upper and lower bounds for the value of a sensor reading in addition to the upper and lower bounds for the number of sensor nodes.

A node X sends to the BS a MAC, $M = MAC(K_X, m)$, only if it contributes to the synopsis-edge as in SecureCount. The format of the message m sent by a node is $[X|A|Seed|v]$, where X is the node id, *Seed* is the random seed included in the broadcast query, A represents the collection of bits in the synopsis to which X contributes, and v is X's sensed value.

Security Analysis In the case of the Sum aggregate, the attacker could falsely set some bits in its synopsis not only by using a false node id but also using a false sensor reading. Although MACs from the contributing nodes enable the BS to verify the node Ids, the BS cannot verify the sensed value of a node. A compromised node can claim to have a large sensed value close to the upper bound u_v to increase its chance of being able to contribute to the synopsis-edge.

The following theorem (whose proof can be found in Appendix B) shows that this attack's impact is limited.

Theorem 1. Let ρ be the number of compromised nodes in a network of n nodes. Let u_v and a_v denote the upper bound and the average value of the sensor reading respectively. Let S be the final synopsis computed at the sink and let R be the length of the prefix of all ones in S. Let s denote the value of the Sum aggregate. If each compromised node claims that its sensed value is equal to the upper bound u_v, and if $(\rho \cdot u_v) < s$, then the probability $\Pr[S[R+1] = 1]$ is proportional to the product of the fraction of compromised nodes in the network, ρ/n.

Note that if the compromised node contributes to the $(R+1)$th bit BS's estimate of Sum doubles. Thus, the theorem shows that for a large network, as long the fraction of compromised nodes ρ grows sub-linearly, the probability of this attack succeeding is small. For smaller networks, the probability of this attack succeeding depends upon the ratio ρ/n and on the ratio u_v/a_v. As an example, if $n = 1000$, $\rho = 25$, and $u_v/a_v = 4$, then $\Pr[S[R+1] = 1] = 0.064$.

The impact of the attack is further reduced if we employ the PCSA algorithm in which m independent synopses are computed and the final estimator $\bar{R}$ is calculated by averaging these m estimators. As an example, to add an error of 40% to the final Sum, the attacker needs to set the $R+1$-th bit in at least $\frac{m}{2}$ synopses. In the example above where $\Pr[S[R+1] = 1]$ is 0.064, this probability is close to zero when m is 20. This example illustrates that this attack's impact is limited when $(\frac{\rho \cdot u_v}{n \cdot a_v})$ is small.

On the other hand, when $(\frac{\rho \cdot u_v}{n \cdot a_v})$ is large, we cannot neglect the possibility that the attacker will succeed in injecting a significant error to the Sum computed at the sink. To address this scenario, we can use a scheme in which a node that contributes to the synopsis-edge needs an endorsement from at least ν neighbors attesting to the validity of its sensed value. We assume that the sensed values of one-hop neighbors are correlated so that one node can verify the reading of its neighbors. We assume that there are fewer than ν compromised nodes among the one hop neighbors of any node. Each contributing node X collects at least ν endorsements from its one-hop neighbors in the form of a MAC computed over the sensor reading using the pairwise key that the neighbor shares with the sink. Then X computes an XMAC [1] by XORing the collected MACs and X's own MAC, and sends the XMAC to the BS. (Zhu et al. [25] use an identical scheme to reduce the total size of the MACs.) We also assume that BS has the knowledge to verify if a set of nodes are one-hop neighbors, which prevents the collusion attack. (We refer to this scheme as the XMAC-based scheme.)

Computation and Communication Overhead The number of contributing nodes η is less than $\frac{1}{\phi} \cdot (\frac{2u_s}{l_s} - 1)$, where u_s and l_s are the upper bound and lower bound of Sum. As in the case of Count, η is independent of the network size and thus this approach is scalable. With respect to worst case communication overhead, a node will need to forward at most η MACs.

6. THE EXTENDED APPROACH: TRADING LATENCY FOR COMMUNICATION OVERHEAD

When the ratio (τ) of the upper bound of the aggregate to the lower bound is high, the basic approach described in the previous section is not scalable because the worst case communication cost incurred by a node is proportional to τ. In this section, we describe an approach which has lower communication costs in comparison to the basic approach at the expense of greater latency.

6.1 Protocol Overview

Our extended approach is based on the observation that the expected number of nodes that contribute to bits i, where $R < i \leq k$ in the synopsis (k is the length of the synopsis) is very small. In fact, using Property 2 and Property 4 from Section 2, we can show that expected number of nodes contributing to the Rth and higher-order bits of S is less than $2/\phi \approx 2.58$.

We use a sliding-window based approach in which the aggregation phase is divided into multiple epochs[3]. Starting at the rightmost bit k, we proceed from right to left within the synopsis S using a bit window of w bits. In each epoch, only the nodes that contribute a 1 to the bits in S corresponding to the current position of the window, send MACs to the sink that can be used to verify their contribution to S. In other words, in the first epoch, only nodes that contribute a 1 to bits k to $k-w+1$ respond to the query. In epoch two, nodes that contribute to bits between $k-w$ and $k-2w+1$ respond, and so on.

The algorithm terminates when the querying node has determined that the remaining bits of S to the left of the current window are likely to be 1 with high probability. The design of this termination criterion is the main challenge in using this approach; we discuss the termination criterion and its analytical underpinnings in detail.

Once the querying node determines that the algorithm can terminate it broadcasts a STOP message to the network to announce the end of the iterative aggregation procedure.

6.2 Protocol Operation

The operation of the protocol is similar to that of the protocol used in the basic approach with some minor differences as follows. The query message broadcast to the network includes the window size w in addition to the other parameters. As in the original synopsis diffusion algorithm [3, 14], we assume that the time is synchronized among BS and the sensor nodes. Each node computes the start and end time of the current epoch, based on the window w.

Further, although the MACs generated by nodes are sent to the BS over the course of multiple epochs, the fused synopsis computed by each node is forwarded to its parent in the first epoch. Thus, the BS can compute the aggregate at the end of the first epoch itself, although this aggregate may be erroneous in the presence of compromised nodes.

6.3 Termination Criterion

The goal of our algorithm is to find r, the lowest-order bit in S that is 0. Further, recall that S is of the form $1^{r-1}0\cdots$, where the bits at positions $i > r$ are highly likely to be 0. Thus, the intuition behind our termination criterion is simple: as we examine the bits of S moving from right to left, if we observe two consecutive 1's, i.e., if we observe the string "110", it is highly likely that the 0 is at the rth position. In fact, we can show analytically that the probability of this event is greater than 90% which follows from the following theorem.

Theorem 2. Let F denote the event that the string "$0s^l11$" where s^l represents any string of length l, $l \geq 0$ appears in a synopsis S. The probability of the event F is less than 10%. (The proof is given in the appendix.)

Further, we can take advantage of the fact that most applications will use the PCSA algorithm to reduce the approximation error in estimating $R = r-1$. Recall that in the PCSA algorithm m synopses are computed in parallel. Let R_i denote the value of R estimated from the ith synopsis. Then, according to the PCSA algorithm, the the expected value of the random variable R is estimated by averaging the individual values of R for each synopsis, i.e., $E[R] = \bar{R} = \sum_{i=1}^{i=m} R_i$.

Although there is likely to be some variation among the R_i, we know from Property 3 in Section 2 that the variation is expected to correspond to two bit position both to the left and the right of the true value of R. This suggests that there is a high degree of correlation between the R_i for different synopses. Thus, in our window-based approach, we can increase our confidence that we have found the correct position of R, if we observe the bit pattern "11" in multiple synopses among the m that are being computed in parallel. Based on this intuition, our termination criterion consists of checking whether we have observed the string "11" in at least m' out of the m synopses.

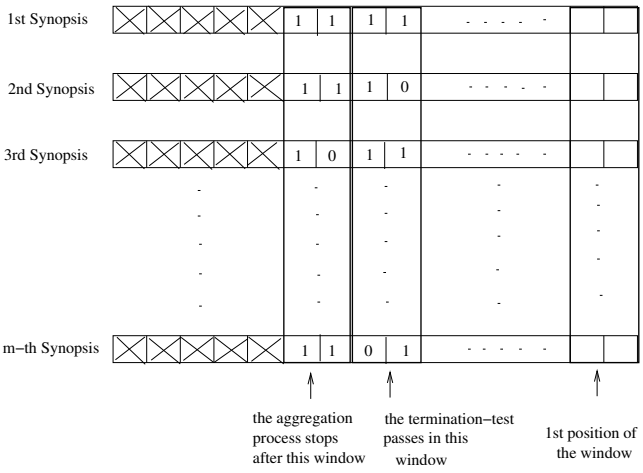

Figure 3: Each synopsis is divided into several windows of width $w = 2$ bits. After the termination criterion is satisfied, the base station broadcasts a STOP message and the aggregation phase stops after the next epoch. In each epoch, nodes which contribute to the corresponding window send a MAC to the base station. The MACs which correspond to the crossed bits are never sent.

Our goal in selecting the threshold m' is to reduce the likelihood of both a false positive, which means that the algorithm was terminated too early, and a false negative, which means that the algorithm terminated too late, i.e., after the sliding window had already crossed the true position of R. A false positive results in an over-estimate of R, whereas a false negative results in additional communication overhead. We now show that it is possible to find a suitable value for m' such that the probability of false positive and the probability of false negative are both low.

Theorem 3. Let G_i denote the event that both bit i and $(i+1)$ in a synopsis S are 1. Let λ denote the expected value of the estimator $\bar{R}$. Then, $\Pr[G_\lambda] = 0.3454$, $\Pr[G_{\lambda+1}] = 0.1315$, and $\Pr[G_{\lambda+2}] = 0.0412$.

Because of space limitations, the proof of this theorem can be found in the appendix.

If the sliding window in our algorithm is two bits wide, i.e, $w = 2$, from the definition of the false positive (FP), we get that the probability Pr[FP] is the probability that the event $G_{\lambda+2}$ occurs in m' or more synopses. Similarly, the probability of a false negative, Pr[FN] is the probability that the event G_λ occurs in fewer than m' synopses. For $m = 20$ (which is the typical value used in previous

[3]The original synopsis diffusion algorithm [3, 14] takes one epoch to complete.

work [3, 14], we find that the best value of m' is 4 in which case $\Pr[\text{FP}] = 0.0082$ and $\Pr[\text{FN}] = 0.0484$. The same approach illustrated here can be used to derive the appropriate threshold m' for other window sizes.

Figure 3 illustrates the operation of our algorithm for $w = 2$. Assume that the termination criterion is satisfied in epoch e. The BS broadcasts a STOP message which directs all nodes to terminate the aggregation phase. Note that by the time each node in the network receives the broadcast STOP message, many of the nodes will have already sent MACs corresponding to their contributions to the next epoch $e+1$ of the algorithm. Thus, the effect of the termination criterion being satisfied in epoch e message is to terminate the aggregation after epoch $e+1$.

We can take advantage of this extra epoch to further increase our level of confidence in the estimated value of R. Let the bit position of the sliding window in epoch e correspond to bits α and $\alpha+1$. Instead of estimating $R = \alpha+1$ because m' out of m synopses had both bits α and $\alpha+1$ equal to 1, we can now estimate R based on the observed value of R_i for all m synopses. Our simulations show that estimate of the aggregate computed using our extended approach is close to the estimate computed using the original synopsis diffusion [3, 14] algorithm.

Latency The number of epochs taken by our sliding window approach depends on the ratio (χ) of the upper bound of the aggregate to the actual value. If the upper bound is u and the actual value is γ, for a window of width w the number of epochs is equal to

$$(\lfloor \frac{\log_2 \frac{\phi u}{m} - \log_2 \frac{\phi \gamma}{m}}{w} \rfloor + 2) = (\lfloor \frac{\log_2 \chi}{w} \rfloor + 2)$$

Communication Overhead Theorem 3 implies that it is highly likely that the sliding window contains the position $\bar{R}$ when the termination criterion is satisfied. As discussed above, if the termination criterion is satisfied in epoch e, the aggregation completes after epoch $e+1$. Thus, by property 2 and property 4 in Section 2, if m synopses are computed in parallel, the expected number of nodes which send a MAC varies in the range of $(2.58 \times m)$ to $(5.16 \times m)$. Even if a sensor node contributes to more than one bits, it sends just one MAC validating all the bits. Note that the number of contributing nodes does not exceed this range even if the network size is increased. Our simulation results show that 85 MACs are sent on average when $m = 20$.

We observe that the width of the window w determines a tradeoff between the communication overhead and the latency. If we divide the synopses into wider windows, the number of MACs sent and hence the communication overhead will increase while the latency of the aggregation process will decrease, and vice versa.

6.4 Discussion

An alternative approach to the sliding window-based approach described above is one in which the base station computes the aggregate of interest in the first epoch using the original Synopsis Diffusion algorithm. It then broadcasts a message requesting only the nodes that contribute to the bit window that contains R to send the MACs authenticating their local synopses. If the BS successfully verifies all the MACs it receives, then the protocol terminates at the end of the second epoch. However, if it does not receive the requested MACs or if one or more MACs are invalid, the BS executes the sliding-window protocol described above to compute the correct value of R. If the probability of compromised nodes being present in the network is low, then this alternative approach is preferable to the extended approach since it will have much lower latency on average.

7. SIMULATION RESULTS

In this section, we report on a detailed simulation study that examined the performance of our attack-resilient aggregation algorithms discussed in Sections 5 and 6. Our simulations were written using the TAG simulator developed by Madden et al. [11]. We added the attack-resilient functionality to the source code provided by Considine et al. [3] which simulates their multipath aggregation algorithms in the TAG simulator environment.

7.1 Simulation Environment

For our basic experimental network topology, we used a regular 30×30 grid with 900 sensor nodes, where one sensor is placed at each grid point and the base station is at the center of the grid, as in [3]. The communication radius of each node is $\sqrt{2}$ unit allowing the nearest eight grid neighbors to be reached.

The goal of our simulation experiments is to examine the communication overhead and accuracy of our scheme in the presence of packet losses, which are relatively frequent in sensor networks. We use a simple packet loss model in which packets are dropped with a fixed probability; this packet loss rate is assumed to include packets that are lost due to being dropped by compromised nodes.

We do not model any additional attacks by compromised nodes, specifically the falsified subaggregate and the falsified local value attacks, in our simulation. This is because we have already shown that these attacks cannot affect the estimate of the aggregate computed at the sink. Consequently, these attacks simply have the effect of increasing the communication and computation overhead; in effect, they become a form of DOS or resource consumption attacks.

We assign a unique id to each sensor, and we assume that the sensor reading is a random integer uniformly distributed in the range of 0 to 250 units. We compute 20 synopses in parallel using the PCSA algorithm as in the experiments reported in [3, 14]. We use the method of independent replications as our simulation methodology. Each simulation experiment was repeated 200 times with a different seed. The plots below show the 95% confidence intervals of the reported metric.

7.2 Results and Discussion

Due to space constraints, we will only present the results of our extended approach for computing the Sum aggregate.

Accuracy of our estimate In the first set of experiments, we validate our claim that our attack-resilient approach has the same accuracy in computing the true value of the aggregate as the original synopsis diffusion approach. Figure 4a plots the estimates of our approach and the synopsis diffusion approach as a function of the packet loss rate. We observe that the two estimates are indeed very close in all loss rate conditions. We observe that the average value of the sensor reading is approximately 125, i.e., the accurate Sum is $900 \times 125 = 11250$.

Communication overhead We now compare the communication overhead of our approach to that of the original synopsis diffusion approach. Figure 4(b) plots the total number of bytes transmitted for computing the Sum aggregate. As discussed in Section 5.3, for preventing a node from using a false reading to generate its own local synopsis, we can adopt two approaches. In the first approach, we ignore the impact of the falsified local value attack; in the figure, this approach is labeled as ARSD (attack-resilient synopsis diffusion). The second approach requires the contributing node to include a XMAC, which corresponds to an endorsement from its neighbors, in the message; in the figure, this approach is labeled ARSD+XMAC.

For ARSD+XMAC, each contributing node sends an authentication message which has two parts: the first part contains the ID (2

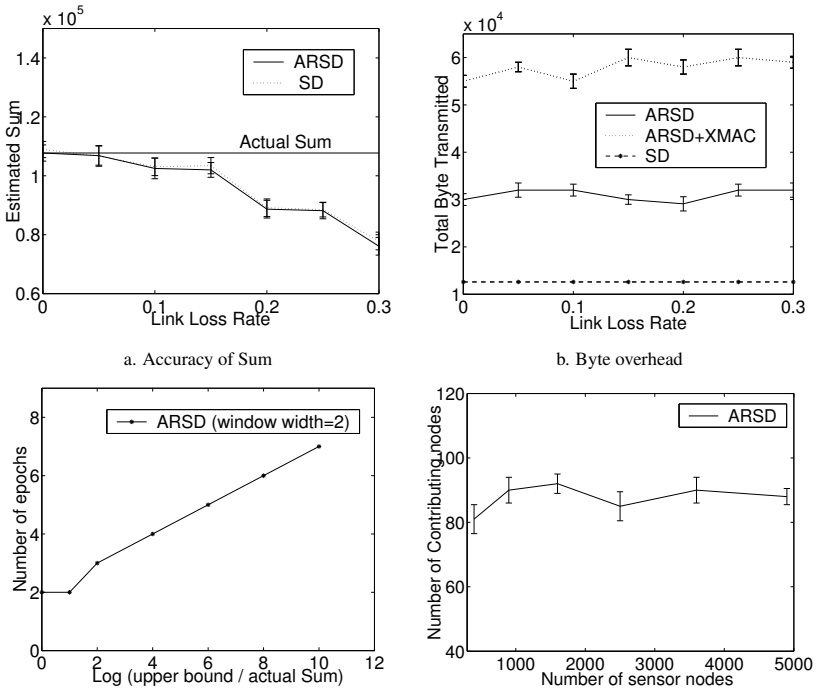

Figure 4: Experimental Results

bytes) of the contributing node and its sensed value (3 bytes), and the second part includes the IDs of the k neighbors and a XMAC (4 bytes). If the value of k is not more than 4 then a node needs 8 bytes to specify the identity of the neighbors whose MACs are used to generate the XMAC. Thus, the size of one authentication message is 17 bytes. For ARSD, the contributing node just needs to send its own MAC; no neighbor endorsement is needed, which reduces the authentication message size to 9 bytes.

Figure 4(b) shows that the byte overhead of the ARSD+XMAC scheme is roughly 5 times larger than the original approach, whereas ARSD is 2.5 times larger than the original approach. One might expect that if loss rate is high our extended approach may take more time to stop because some MACs could be lost en-route, and, as a result, the communication overhead could increase. But Figure 4(b) demonstrates that the overhead of the extended approach does not increase with the loss rate.

Latency As discussed in Section 6, the latency of the extended approach depends on the looseness of the base station's estimate of the upper bound of Sum. Figure 4(c) plots the number of epochs taken by our extended approach as a function of the ratio of the upper bound to the actual value of the aggregate. The figure shows that the number of epochs increases at logarithmic scale with the ratio of the upper bound to the actual Sum. We note, however, that the byte overhead of our scheme is independent of this ratio.

Effect of network size In this experiment, we study the impact of the network size on the communication overhead of the extended approach. The communication overhead depends upon the number of contributing nodes that send a MAC to the base station, authenticating their synopsis. Recall from Section 6 that the expected number of contributing nodes is independent of the network size. Figure 4(d) confirms our analysis; we observe that the number of contributing nodes is more or less constant as the network size increases[4]. This figure thus illustrates the scalability of our approach for attack-resilient aggregation.

8. RELATED WORK

Several data aggregation protocols [11, 19, 23] have been proposed in the literature which efficiently fuse the sensed information en-route to the base station to reduce the communication overhead. Since packet losses and node failures are relatively common in sensor networks, several studies have investigated the design of robust aggregation algorithms. Considine et al. [3] and Nath et al. [14, 12] have presented robust aggregation approaches that combine the use of multi-path routing with clever algorithms that avoid double-counting of sensor readings. Jelasity et al. [9] proposed a robust gossip-based protocol for computing aggregates over network components in a fully decentralized fashion. They assume that nodes form an overlay network where any pair of nodes are considered to be neighbors, which makes this protocol impractical for sensor networks. We note that none of the above algorithms were designed with security in mind.

Recently several researchers have examined security issues in aggregation. Wagner [17] examined the problem of resilient data aggregation in presence of malicious nodes, and provided guidelines for selecting aggregation functions in a sensor network. Buttyan et al. [2] proposed a model of resilient aggregation and analyzed the maximum deviation from the true value of the aggregate that an adversary could introduce while remaining undetected. The models used by by both Buttyan et al and Wagner assume that there is no in-network aggregation, that is, the aggregation is performed at the sink. Przydatek et al [16] present protocols that can be used by a trusted remote user to query a sensor network in which the base

[4]The link loss rate is held at 20% in this set of experiments.

station may be compromised and the base station is the only aggregator. One of the protocols described by Przydatek et al is a robust approach for counting distinct elements in a data stream that can be used for estimating the size of the network, i.e., the Count aggregate. Their approach for counting distinct elements is similar to our scheme for Count in the sense that in both cases only a subset of elements need to be verified.

The first secure in-network data aggregation protocol was designed by Hu and Evans [8]. Their protocol is effective only if no more than one node is compromised. Recently, Yang et al. [18] proposed SDAP, a secure hop-by-hop data aggregation protocol which can tolerate more than one compromised node. SDAP is a tree-based aggregation protocol with communication cost comparable with that of the ordinary aggregation protocols while it provides certain level of assurance on the trustworthiness of the aggregation result. As SDAP is a tree-based protocol, it is vulnerable to link loss and node failures which are relatively common in sensor networks, whereas our protocol is robust to this communication loss and, at the same time, secure against compromised nodes.

We note that our work is related to the general problem of preventing false data injection. Du et al. [4] proposed a mechanism that allows the base station to check the aggregated values submitted by several designated aggregators, based on the endorsements provided by a certain number of witness nodes around the aggregators. Their scheme does not provide per-hop aggregation. Several other works [20, 21, 25] have also proposed solutions to prevent false data injection attacks in sensor networks, but they do not involve data aggregation.

9. CONCLUSION

In this paper, we investigated the security issues of synopsis diffusion framework in presence of compromised nodes. We showed that a compromised node can launch several simple attacks on the existing aggregation algorithms, which could significantly deviate the estimate of the aggregate. We also proposed modifications to the aggregation algorithms that guard against these attacks. Our analytical results and simulation results show that our approach is effective and it incurs minimal computation and communication overhead.

In this paper, we assume that a sensor node has a security association only with the base station, and, as a result, the authentication messages cannot be processed in-network in our approach. To further reduce the communication overhead, we plan to exploit other security settings, e.g., local pairwise keys among nodes, as a part of our future work.

10. REFERENCES

[1] M. Bellare, R. Guerin, and P. Rogaway. XOR MACs: New methods for message authentication using finite pseudorandom functions. In *Proc. of the 15th Annual International Cryptology Conference on Advances in Cryptology - CRYPTO'95*, pages 15–28, 1995.

[2] L. Buttyan, P. Schaffer, and I. Vajda. Resilient aggregation with attack detection in sensor networks. In *Proc. of 2nd IEEE Workshop on Sensor Networks and Systems for Pervasive Computing*, 2006.

[3] J. Considine, F. Li, G. Kollios, and J. Byers. Approximate aggregation techniques for sensor databases. In *Proc. of IEEE Int'l Conf. on Data Engineering (ICDE)*, 2004.

[4] W. Du, J. Deng, Y. S. Han, and P. Varshney. A pairwise key pre-distribution scheme for wireless sensor networks. In *Proc. of the 10th ACM Conference on Computer and Communications Security (CCS '03).*, 2003.

[5] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences*, 31(2):182–209, 1985.

[6] S. Ganeriwal and M. B. Sribastava. Reputation-based framework for highly integrity sensor networks. In *Proc. of ACM Workshop on Security of Sensor and Adhoc Networks (SASN)*, Washington, DC, 2004.

[7] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient energy-efficient multipath routing in wireless sensor networks. *Mobile Comuting and Communication Review*, 4(5):11–25, 2001.

[8] L. Hu and D. Evans. Secure aggregation for wireless networks. In *Proc. of Workshop on Security and Assurance in Ad hoc Networks.*, 2003.

[9] M. Jelasity, A. Montresor, and O. Babaoglu. Gossip-based aggregation in large dynamic networks. *ACM Transactions on Computer Systems*, 23(3):219–252, 2005.

[10] F. Koushanfar, M. Potkonjak, and A. Sangiovanni-Vincentelli. Fault tolerance techniques in wireless ad-hoc sensor networks. In *Sensors 2002. Proceedings of IEEE*, pages 1491 – 1496.

[11] S. Madden, M. J. Franklin, J.M. Hellerstein, and W. Hong. TAG: A tiny aggregation service for ad hoc sensor networks. In *Proc. of 5th USENIX Symposium on Operating Systems Design and Implementation*, 2002.

[12] A. Manjhi, S. Nath, and P. Gibbons. Tributeries and deltas : Efficient and robust aggregation in sensor network streams. In *Proc. of ACM International Conference on Management of Data (SIGMOD)*, 2005.

[13] Mica Motes. http://www.xbow.com.

[14] S. Nath, P. B. Gibbons, S. Seshan, and Z. Anderson. Synopsis diffusion for robust aggregation in sensor networks. In *Proc. of the 2nd international conference on Embedded networked sensor systems (SenSys)*, 2004.

[15] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. In *Seventh Annual International Conference on Mobile Computing and Networks (MobiCOM)*, 2001.

[16] B. Przydatek, D. Song, and A. Perrig. SIA: Secure information aggregation in sensor networks. In *Proc. of the 1st international conference on Embedded networked sensor systems (SenSys)*, 2003.

[17] D. Wagner. Resilient aggregation in sensor networks. In *Proc. of ACM Workshop on Security of Sensor and Adhoc Networks (SASN)*, 2004.

[18] Y. Yang, X. Wang, S. Zhu, and G. Cao. SDAP: A secure hop-by-hop data aggregation protocol for sensor networks. In *Proc. of ACM MOBIHOC*, 2006.

[19] Y. Yao and J. E. Gehrke. The cougar approach to in-network query processing in sensor networks. *ACM SIGMOD Record*, 31(2):9–18, September 2002.

[20] Fan Ye, Haiyun Luo, Songwu Lu, and Lixia Zhang. Statistical en-route filtering of injected false data in sensor networks. In *Proc. of IEEE Infocom*, 2004.

[21] W. Zhang and G. Cao. Group rekeying for filtering false data in sensor networks: A predistribution and local collaboration-based approach. *Proc. of IEEE Infocom*, 2005.

[22] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proc. of*

the 1st international conference on Embedded networked sensor systems (SenSys), 2003.

[23] J. Zhao, R. Govindan, and D. Estrin. Computing aggregates for monitoring sensor networks. In *Proc. of the 2nd IEEE International Workshop on Sensor Network Protocols and Applications*, 2003.

[24] S. Zhu, S. Setia, and S. Jajodia. LEAP: Efficient security mechanisms for large-scale distributed sensor networks. In *Proc. of the 10th ACM Conference on Computer and Communications Security (CCS '03).*, 2003.

[25] S. Zhu, S. Setia, S. Jajodia, and P. Ning. An interleaved hop-by-hop authentication scheme for filtering injected false data in sensor networks. In *Proc. of IEEE Symposium on Security and Privacy*, 2004.

Appendix

A.

Below we describe the algorithm (SecureCount) executed by each node in response to a Count query. X represents the node Id.

Algorithm 2 SecureCount($X, Seed, a, b$)

1: $\mathcal{M} = \{\}$; // $\mathcal{M}$ is initialized as an empty set
2: $i = SG(X, Seed)$; // X contributes to bit i
3: **if** $(a \leq i \leq b)$ **then**
4: $m = [X|i|Seed]$;
5: $M = MAC(K_X, m)$;
6: $\mathcal{M} = \mathcal{M} \cup M$;
7: **end if**
8: $S_l = SF()$; // S_l is the fused synopsis at X
9: $\mathcal{M} = \mathcal{M} \cup \mathcal{C}$; // $\mathcal{C}$ represents the set of MACs X received from // its child nodes
10: $X \rightarrow * : S_l|\mathcal{M}$;

B. Proofs of Theorems

We provide the proofs for the theorems present in the paper.

Theorem 1. Let there be n nodes in the sensor network among which ρ nodes are compromised. Let u_v and a_v denote the upper bound and the average value of the sensor reading respectively. Let S be the final synposis computed at the sink and let R be the length of the prefix of all ones in S. Let s denote the value of the Sum aggregate. If each compromised node claims that its sensed value is equal to the upper bound u_v, and if $(\rho \cdot u_v) < s$, then the probability $\Pr[S[R+1] = 1]$ is proportional to the product of the fraction of compromised nodes in the network, ρ/n, and the ratio u_v/a_v.

PROOF. By property 2 in Section 2, the expected value of the estimator R, for the Sum synopsis S, is $\log_2(\phi s)$, where s denotes the Sum. As a node X with sensed value v invokes the function CT() v times (in the synopsis generation phase), the probability that X does not contribute to bit i in S is $(1-\frac{1}{2^i})^{u_v}$. So, the probability (p) that a node with sensed value u_v will contribute to the $(R+1)$th bit is $(1-(1-\frac{1}{2^{R+1}})^{u_v})$. After simplifying, we get

$$p = 1-(1-\frac{1}{2\phi s})^{u_v} \approx 1-(1-\frac{u_v}{2\phi s}) = \frac{u_v}{2\phi s}$$

The above approximation is valid as u_v is smaller than s. If there are ρ compromised nodes, then $\Pr(S[R+1] = 1)$ is

$$q = 1-(1-p)^{\rho} \approx \rho \cdot p = \frac{\rho u_v}{2\phi s} = (\frac{1}{2\phi}) \cdot (\frac{\rho}{n}) \cdot (\frac{u_v}{a_v})$$

□

To prove Theorem 2, we first prove the following results.

Lemma 1. Let E_i, $1 \leq i \leq k-2$ denote the event that the string "011" appears in a synopsis S from bit i to bit $(i+2)$ (i.e., $S[i] = 0$, $S[i+1] = 1$, and $S[i+2] = 1$), where k is the length of S. The maximum value of the probability (p_i) of the event E_i is 0.037 for any value of i and for any value of Count (or Sum) shared by S.

PROOF. If function $CT()$ is invoked once (ref. Section 2), then $\Pr[S[j] = 1] = q_j = \frac{1}{2^j}, 1 \leq j \leq k$. This probability increases if it is given that bit j', $1 \leq j' \leq k$ will remain 0. Specifically,

$$\Pr[S[j] = 1 | S[j'] = 0] = \frac{q_j}{1-q_{j'}} = \frac{1}{(2^j) \cdot (1-\frac{1}{2^{j'}})}.$$

If ψ is the total Count (or Sum) shared by synopsis S, then $\Pr[S[i] = 0] = (1-q_i)^{\psi}$, and

$$\begin{aligned} &\Pr[S[i+1] = 1, S[i+2] = 1] \\ &= 1 - \Pr[S[i+1] = 0] - \Pr[S[i+2] = 0] \\ &\quad + \Pr[S[i+1] = 0, S[i+2] = 0] \qquad (1) \\ &= 1-(1-q_{i+1})^{\psi} - (1-q_{i+2})^{\psi} \\ &\quad + (1-q_{i+1}-q_{i+2})^{\psi} \\ &= 1-(1-\tfrac{1}{2^{i+1}})^{\psi} - (1-\tfrac{1}{2^{i+2}})^{\psi} \\ &\quad + (1-\tfrac{1}{2^{i+1}}-\tfrac{1}{2^{i+2}})^{\psi} \end{aligned}$$

So, the probability of the event E_i is

$$\begin{aligned} p_i &= \Pr[S[i] = 0] \times \\ &\quad \Pr[S[i+1] = 1, S[i+2] = 1 \mid S[i] = 0] \\ &= (1-\tfrac{1}{2^i})^{\psi} \times \\ &\quad [1-(1-\tfrac{1}{(2^{i+1}) \cdot (1-\frac{1}{2^i})})^{\psi} - (1-\tfrac{1}{(2^{i+2}) \cdot (1-\frac{1}{2^i})})^{\psi} \\ &\quad + (1-\tfrac{1}{(2^{i+1}) \cdot (1-\frac{1}{2^i})} - \tfrac{1}{(2^{i+2}) \cdot (1-\frac{1}{2^i})})^{\psi}] \qquad (2) \end{aligned}$$

Note that if $i << log_2(\psi)$, the 1st factor is close to 0 and second factor is close to 1, making p_i close to 0. On the other hand, if $i >> log_2(\psi)$, the 1st factor is close to 1, but the 2nd factor are close to 0, again making p_i close to 0. p_i attains the highest value when i is close to $log_2(\psi)$. We have numerically found that the maximum value of p_i is 0.037, for any value of i or ψ.

□

Lemma 2. Let E denote the event that the string "011" appears in a synopsis S at any position. The probability of the event E is less than 0.099.

PROOF. E_i denotes the event that "011" appears in a synopsis S where 0 is at the ith bit. We observe that the events E_i, E_{i+1}, E_{i+2} are mutually exclusive, for any value of i. Following the same direction of Lemma 1, we can show that the probability that " $011s^l011$" appears in synopsis S is close to zero, where s^l represents any string of length l, $l \geq 0$. So, the probability that two events E_i and E_j where $j \geq (i+3)$ can occur together is negligible, for any value of i. As a result, we can approximate that events E_is are mutually exclusive and hence the probability of event E is

$$p = \sum_{i=1}^{k} p_i,$$

where p_i is given by expression (2) and k is the length of S. We have numerically found that maximum value of p is 0.099.

□

Lemma 3. Let F_i denote the event that a string "$0s^i11$" appears in a synopsis S, and let F denote the general event that a string

"$0s^l11$", $l \geq 0$ appears in S, where s^l represents any string of length l. $\Pr[F] = \Pr[F_0]$

PROOF. As the string "011" is a special case of string "$0s^l11$" where $l = 0$, $\Pr[F] \geq \Pr[F_0]$. On the other hand, if string $s' =$ "$0s^l11$", $l \geq 0$ appears in S, string "011" must also appear as a substring of s'. As an example, if $s' =$ "01011" where $s^l =$ "10", we can see "011" as a substring of s'. Hence, $\Pr[F] \leq \Pr[F_0]$. So, we get that $\Pr[F] = \Pr[F_0]$.

□

Theorem 2. Let F denote the event that the string "$0s^l11$" where s^l represents any string of length l, $l \geq 0$ appears in a synopsis S. The probability of the event F is less than 10%.

PROOF. As the event F_0 in Lemma 3 is same as the event E in Lemma 2, we get that the probability of event F is less than 10%. □

Theorem 3. Let G_i denote the event that both bit i and $(i+1)$ in a synopsis S are 1. Let λ denote the expected value of the estimator $\bar{R}$. Then, $\Pr[G_\lambda] = 0.3454$, $\Pr[G_{\lambda+1}] = 0.1315$, and $\Pr[G_{\lambda+2}] = 0.0412$.

PROOF. The expected value of $\bar{R}$ is $\log_2(\phi \cdot \frac{\gamma}{m})$, which we denote by λ, where γ is the total Count (or Sum) shared by m synopses following the algorithm PCSA. If function $CT()$ is invoked once (ref. Section 2),

$$\Pr[S[i] = 1] = q_i = \frac{1}{m} \cdot \frac{1}{2^i}, \quad 1 \leq i \leq k$$

because synopsis S is selected with probability $\frac{1}{m}$ among m synopses. As γ is the total Count (or Sum) shared by all synopses, we get by using similar expression as (1) in Lemma 1 that

$$\begin{aligned}\Pr[G_i] = 1 - (1 - \tfrac{1}{m \cdot 2^i})^\gamma - (1 - \tfrac{1}{m \cdot 2^{i+1}})^\gamma \\ + (1 - \tfrac{1}{m \cdot 2^i} - \tfrac{1}{m \cdot 2^{i+1}})^\gamma\end{aligned}$$

So,

$$\begin{aligned}\Pr[G_\lambda] &= 1 - (1 - \tfrac{1}{m \cdot \phi \cdot \frac{\gamma}{m}})^\gamma - (1 - \tfrac{1}{m \cdot \phi \cdot 2 \cdot \frac{\gamma}{m}})^\gamma \\ &\quad + (1 - \tfrac{1}{m \cdot \phi \cdot \frac{\gamma}{m}} - \tfrac{1}{m \cdot \phi \cdot 2 \cdot \frac{\gamma}{m}})^\gamma \\ &\approx 1 - e^{-\frac{1}{\phi}} - e^{-\frac{1}{2 \cdot \phi}} + e^{-\frac{3}{2 \cdot \phi}} = 0.3454\end{aligned}$$

Similarly, we find $\Pr[G_{\lambda+1}] = 0.1315$, and $\Pr[G_{\lambda+2}] = 0.0412$.

□