ZoneTrust: Fast Zone-Based Node Compromise Detection and Revocation in Wireless Sensor Networks Using Sequential Hypothesis Testing

Jun-Won Ho
Department of Information Security
Seoul Women's University
621 Hwarangro, Nowon-Gu, Seoul, South Korea
jwho@swu.ac.kr
Matthew Wright, and Sajal K. Das
Department of Computer Science and Engineering
The University of Texas at Arlington
Arlington, TX, USA
{mwright, das}@cse.uta.edu

***Abstract*—Due to the unattended nature of wireless sensor networks, an adversary can physically capture and compromise sensor nodes and then mount a variety of attacks with the compromised nodes. To minimize the damage incurred by the compromised nodes, the system should detect and revoke them as soon as possible. To meet this need, researchers have recently proposed a variety of node compromise detection schemes in wireless ad hoc and sensor networks. For example, reputation-based trust management schemes identify malicious nodes but do not revoke them due to the risk of false positives. Similarly, software-attestation schemes detect the subverted software modules of compromised nodes. However, they require each sensor node to be attested periodically, thus incurring substantial overhead. To mitigate the limitations of the existing schemes, we propose a zone-based node compromise detection and revocation scheme in wireless sensor networks. The main idea behind our scheme is to use sequential hypothesis testing to detect suspect regions in which compromised nodes are likely placed. In these suspect regions, the network operator performs software attestation against sensor nodes, leading to the detection and revocation of the compromised nodes. Through quantitative analysis and simulation experiments, we show that the proposed scheme detects the compromised nodes with a small number of samples while reducing false positive and negative rates, even if a substantial fraction of the nodes in the zone are compromised. Additionally, we model the detection problem using a game theoretic analysis, derive the optimal strategies for the attacker and the defender, and show that the attacker's gain from node compromise is greatly limited by the defender when both the attacker and the defender follow their optimal strategies.**

I. Introduction

Wireless sensor networks are being deployed in a wide variety of scenarios including hostile environments where adversaries may be present. Because of the applications of these networks, the sensors are often deployed in an unattended manner and controlled remotely by the network operator. The unattended nature of wireless sensor networks make them more vulnerable to attack. Specifically, an attacker may capture and compromise sensor nodes and launch a variety of attacks by leveraging the compromised nodes. For example, the attacker could inject false data values into the network to corrupt its monitoring operation. He could also disrupt common network operations such as cluster formation, routing, and data aggregation. Therefore, it is extremely important to detect and revoke any compromised nodes as quickly as possible.

To mitigate node compromise attacks, researchers have proposed a variety of detection schemes for wireless ad hoc and sensor networks [1], [6], [13], [15], [17], [19], [23]. For example, reputation-based trust management schemes have been proposed to manage an individual node's trust in accordance with its activities [6], [13], [19]. Although malicious nodes can be identified by these schemes, they are not easily revoked due to the risk of false positives. In fact, all reputation-based detection schemes have small but non-zero false positive rates. When the reputation schemes are repeatedly applied to the nodes in the network, a substantial fraction of honest nodes may be caught. Directly revoking these nodes will be very costly with respect to the number of nodes removed from the network.

Another approach to deal with the compromised nodes in sensor networks is software attestation [1], [15], [17], [23]. Such schemes take advantage of the fact that the sensor nodes typically run homogeneous software stored in a flash memory image and should not change without explicit instructions from the base station. These schemes validate that the software running on a node matches the expected software image; subverted image codes can then be detected. This approach features virtually zero false positives, implying that honest nodes will not be unnecessarily removed from the network. However, to achieve a high node compromise detection capability, these schemes require every sensor node to be attested periodically. Hence this *unconditional attestation* incurs a large overhead from the benign sensor nodes.

The motivation behind our work is to address the limitations of the above existing schemes. More specifically, we propose a

reputation-based trust management scheme that is designed to facilitate node compromise detection and revocation through software attestation.[1] The key idea behind our scheme is to detect untrustworthy *zones* and perform software attestation against nodes in these zones to detect and revoke the compromised nodes. To be precise, we first divide the network into a set of zones, establish *trust levels* for each zone, and detect untrustworthy zones with the help of the Sequential Probability Ratio Test (SPRT) [22]. The SPRT determines a zone as untrustworthy if its trust level is low over a period of time or if it oscillates between a high level and a low level. Once a zone is determined to be untrustworthy, the network operator performs software attestation against all nodes in the untrustworthy zone, detects compromised nodes with subverted software modules, and consequently revokes them.

The key observation driving the zone-based approach is that an attacker will not gain much from having a few compromised nodes that are isolated in the network. The attacker can inject a greater concentration of false data or do more to disrupt the control protocols with at least a few compromised nodes in a small region than with a single node. Given this observation, we can gain great advantages by using a zone-based detection approach. Specifically, we can achieve fast node compromise detection and revocation while performing software attestation against only untrustworthy zones and thus saving the large amount of time and effort that would otherwise be incurred from unconditional software attestation. By detecting an entire zone at once, the system can identify the approximate source of bad behavior and react quickly, rather than waiting for a specific node to be identified. Also, when multiple nodes are compromised in one zone, they can all be detected and revoked at one time.

This approach has substantial advantages over prior misbehavior detection schemes in sensor networks. While existing reputation-based systems to revoke misbehaving nodes can suffer from bad-mouthing attacks, our zone-based approach has each zone report on itself. If nodes in the zone attempt to bad-mouth their own zone, it will only accelerate their own detection. Also, the costs of a false positive due to errors are limited to a single round of software attestation within the zone.

Our analysis shows that the SPRT will fail to detect untrustworthy zone in which more than half of nodes are compromised. To enhance the security resilience of the SPRT, we implement a biased sampling strategy in the SPRT in such a way as to detect and revoke untrustworthy zones even if more than half of the nodes in each zone are compromised. The biased strategy necessarily increases false positives and the time to decide that a zone is trustworthy. In our design, however, we can make these costs largely negligible and still enable very robust detection capability.

Through quantitative analysis, we show that our scheme detects untrustworthy zones with few samples while achieving low false positive and negative rates, even if a large number of nodes are compromised in zone. Additionally, we present a detailed game-theoretic analysis that shows the limits of any attack strategy over any number of time slots. Specifically, we formulate a two-player repeated game to model the interaction between the attacker and the defender, find the optimal attack and defense strategies, and show that the attacker's gain is significantly restrained when the attacker and the defender employ their respective optimal strategies. We also demonstrate via simulation experiments that our scheme quickly identifies untrustworthy zones with high detection and low error rates. For instance, our biased sampling strategy helps to detect an untrustworthy zone in just 4.4 samples on an average, with at least 99.0% detection rate and a false positive rate of at most 0.9%, even if 74% of the nodes in the zone are compromised.

The rest of paper is organized as follows. Section II describes the network assumptions and attacker models that we use, as well as our system requirements. Section III describes the proposed scheme of compromised node detection and revocation using the SPRT and analyzes its security and performance. Section IV proposes a biased sampling strategy to further improve the security resilience of our scheme and analyzes its security and performance when biased sampling is used. Section V presents the simulation results. Section VI summarizes the related work from the literature. Finally, Section VII concludes the paper.

II. Model and System Requirements

In this section, we describe our model of wireless sensor networks and the attacker model under which we investigate our schemes. We also describe the system requirements for effective defense.

A. Network Assumptions

We assume a *static* sensor network in which the sensor nodes do not change their locations after deployment. We also assume that all direct communication links between the sensor nodes are bidirectional. We also assume that the base station is a trusted entity. This a standard assumption; if the base station is compromised, the entire mission of the sensor network can be easily undermined. We assume that every sensor node is able to obtain its location information and identify its placement zone by using an existing secure localization scheme such as [2], [14]. Finally, we assume time synchronization such that the clocks of all nodes in a zone are loosely synchronized. This can be achieved by using one of the existing secure time synchronization methods [9], [18]. The incurred overhead is low since the time synchronization protocol only needs to be used within the zone.

B. Attacker Model

We assume that the attacker attempts to maximize the impact of node compromise attacks by compromising a *subset of the nodes* in each target region. In particular, we assume that the attacker could gain more by compromising at least a few nodes in one region, rather than compromising nodes that are isolated from each other. This is because a subset of compromised nodes in a region can launch much more

[1] A preliminary version of this work appeared in [8].

effective attacks through collaboration than an isolated and compromised node can. For instance, false data injection attacks can make the reported observations in a region appear quite different from reality. Attacks on control protocols can also more substantially disrupt system operations, such as routing, clustering, and aggregation.

At the same time, maximizing his attack means that he will not compromise a majority of the nodes in any one region. The attacker could easily undermine the sensing and operations of a region with less than a majority of the nodes. Compromising nodes takes time and effort to locate each node, compromise it, and upload the malicious code. Rather than compromising most of the nodes in a single region, the same time and effort could instead be used to spread out compromised nodes over a wider area and cause greater disruption to the network. Moreover, it is worth noting that the adversary can easily defeat most node compromise detection schemes if he could compromise a large fraction of nodes in any one region. This is because a large number of compromised nodes in a region can prevent the relatively small number of benign nodes from performing node compromise detection and reporting the results. Nevertheless, we show that our scheme is resilient to a substantial majority fraction of compromised nodes in a region.

C. System Requirements

Given the attacker model, we now focus on the requirements for an effective defense against node compromise attacks in a wireless sensor network. A defense mechanism should have several key attributes: (i) fast and accurate detection capability; (ii) efficiency; and (iii) low cost for false positives.

Fast detection of malicious nodes, without missing the attackers due to false negatives, is obviously a critical attribute for a defense against node compromise. We have designed our scheme for high detection rates with just a few observations.

Efficiency is always critical in wireless sensor networks due to the fact that nodes are battery powered, meaning that they have limited energy lifetimes. A defense against node compromise should use minimal computational and communication overhead. Software attestation requires substantial memory traversals and other computations, as well as communication between the attesting and attested nodes [1], [15], [17], [23]. If software attestation is the only means of stopping node compromise, then the system must choose between fast detection through frequent attestation of all nodes or saving energy by attesting nodes infrequently. In our scheme, we aim to save the costs of frequent attestation by using a low-cost reputation scheme to decide when to apply attestation.

Finally, it is critical that false positives do not make the detection system itself a liability. If the detection system has a non-zero false positive rate, then we cannot simply revoke every detected node without revoking many honest nodes and thereby reducing the defense capability of the network. Additionally, if the attacker can perform bad-mouthing attacks or otherwise trigger false alarms against honest nodes, then he can leverage the detection system for denial of service attacks. Our scheme is designed to prevent these problems by having nodes in a region report on their own region only and limiting the cost of false positives to the overhead of software attestation.

III. SPRT-Based Node Compromise Detection and Revocation

This section presents the details of our scheme to detect node compromises and misbehavior on the basis of zones.

In our scheme, we divide the network into a set of zones, establish trust values for each zone, and detect untrustworthy zones in accordance with the zone trust values. Once a zone is determined to be untrustworthy, the network operator attests the software modules of all sensors in the untrustworthy zone, and detects and revokes compromised nodes in that zone. Since benign nodes are attested only in untrustworthy zones, our scheme reduces the overhead incurred from attesting all benign nodes, as in the existing schemes based on software attestation. However, unlike other reputation-based schemes, our scheme benefits from the lack of false positives and ability to fully revoke compromised nodes that software attestation provides.

A straightforward approach for untrustworthy zone detection is to decide that a zone is untrustworthy by observing a single trust value that is less than a trust threshold. However, this approach does not consider the chance of error in the zone trust measurement. Due to the errors in zone trust measurement, a trustworthy (resp. untrustworthy) zone could be detected as untrustworthy (resp. trustworthy). To minimize the impact of such errors, we need to make a decision with multiple pieces of evidence rather than a single one. For this purpose, we use the Sequential Probability Ratio Test (SPRT) [22], which is a statistical decision process that makes a decision with multiple pieces of evidence. The SPRT is considered a one-dimensional random walk [10] with lower and upper limits associated with null and alternate hypotheses, respectively. A random walk moves toward the lower or upper limits according to the type of observation. Once it hits or crosses over the lower (resp. upper) limits, the null (resp. alternate) hypothesis is accepted. The advantage of using the SPRT is that it reaches a correct decision with few pieces of evidence while achieving low false positive and false negative rates [22].

We believe that the SPRT is well-suited for tackling untrustworthy zone detection problem in the sense that we can construct a random walk with two limits in such a way that each walk is determined by the trust value of a zone; the lower and upper limits are properly configured to be associated with the excess and shortfall of a trust threshold, respectively. Specifically, we apply the SPRT to the node compromise detection and revocation problem as follows. Every sensor node in a zone acts as a *trust aggregator* in a round robin manner. In each time slot, the trust aggregator computes a trust level for its zone and reports the zone's trust level to the base station. The base station performs the SPRT with the zone trust information. Each time a zone's trust is below (resp. above or equal to) a trust threshold, it will expedite the test process to accept the alternate (resp. null) hypothesis that a

zone is untrustworthy (resp. trustworthy). Once the base station decides that a zone is untrustworthy, the network operator performs software attestations against all sensor nodes to detect and revoke the compromised nodes in that zone.

Let us first describe the scheme and then present its security and performance analysis.

A. Protocol Description

Before deployment, the network operator assigns a unique ID to every sensor node and divides the network into z non-overlapping zones. Although we do not place any limits on the shape and size of a zone, zone size will affect the communication cost of the scheme. Specifically, an increase in zone size results in a rise in intra-zone communication cost as local trust reports may require multiple hops to reach the trust aggregator. On the other hand, if the zone size is too small, it will be difficult to place enough number of nodes for node compromise detection in a zone. Hence, to minimize the intra-zone communication cost while deploying enough number of nodes in a zone, the optimal zone size would be the maximum size of zone in which every node can directly communicate with each other. For instance, if the shape of zone is square with perimeter L, we will have the optimal zone size when the length of a diagonal of square $\sqrt{2}L$ is equal to a node's communication range r. Thus, the optimal zone size is $\frac{r^2}{2}$.

The network operator also pre-loads secret keying materials onto each sensor node for pairwise key establishment; we can use any key pre-distribution technique for sensor networks, such as [4], [5], [26]. The network operator also pre-loads onto each node a shared secret key with the base station. Our protocol proceeds in three phases.

1) Phase I: Zone Discovery and Trust Aggregator Selection: After deployment, every sensor node u finds out its location and determines the zone to which it belongs. We call this zone the *home zone*. From u's point of view, we call other zones the *foreign zones*. Node u discovers the IDs of all other nodes in its home zone and establishes pairwise secret keys with them. After the zone discovery process, the Trust Aggregator (TA) is selected in a round robin manner as follows. The time domain of a zone is divided into a series of time slots. Each node pseudorandomly decides its duty time slots, during which it acts as a trust aggregator, before each round starts, where a round consists of S time slots such that S is the number of nodes residing in the zone. The reason why we adopt the pseudorandom settings of duty time slots is because the pseudorandom order of the duty time slots is beneficial for the secrecy of our scheme. We argue this in detail in Section III-B3.

We now describe the selection of time duty slots in more detail. All nodes in the network share a pseudorandom number generator (PRNG) in which the starting time of each round is used as the seed value. According to the clock synchronization assumption in Section II-A, the starting time of each round (i.e. seed value) will be the same to all nodes in zone. Each node thus will generate the same sequence of random values uniformly distributed between 0 and 1. We then perform a simple shuffle (the Fisher-Yates shuffle [12]) of the values from 1 to S using the random numbers; since the PRNG of every node is the same, the permutation generated will also be the same.

Each node sets a duty time slot to a random number that corresponds to its order when the nodes in the zone are sorted in ascending order. It repeats this duty time slot selection mechanism for each round before a round starts. Through this pseudorandom duty time slot mechanism, it is guaranteed that duty time slots are pseudo randomly determined per round and only one node is assigned to TA per time slot.

2) Phase II: Trust Formation and Forwarding: For each time slot T_i, each node u in zone Z computes *neighborhood-trust* that is defined in accordance with the difference between the probability distributions of the information generated by u and the information sent to u by u's neighboring nodes in zone Z. Neighborhood-trust acts as an indicator of how much information is shared by two neighboring nodes. The more information u shares with its neighbors, the more belief u has in its neighbors. To compute the difference between two probability distributions, we use the information-theoretic metric Kullback-Leibler (KL)-divergence that is known to be suited for this computation [3]. Specifically, let us assume that u's generated information follows a probability distribution with mean μ and standard deviation σ and falls within the range $[\mu - c\sigma, \mu + c\sigma]$ with probability p, where c is a constant value. Moreover, we assume that the information sent and processed to u by u's neighbors falls within the range $[\mu - c\sigma, \mu + c\sigma]$ with probability q. Node u computes KL-divergence D as follows:

$$D = p \times \ln \frac{p}{q} + (1-p) \times \ln \frac{1-p}{1-q}$$

such that $p > \frac{1}{2}$ and $p \geq q$.

It then computes neighborhood-trust $n_u = \min(1, \frac{1}{1+D})$. This definition of neighborhood-trust is reasonable in the sense that neighborhood-trust is inversely proportional to D and thus it will be one if two probability distributions exactly match with each other. The main rationale behind the restriction of $p > \frac{1}{2}$ is to make the majority of the information generated by u fall within the expected range, leading to better accuracy in neighborhood-trust measurement than the case of $p \leq \frac{1}{2}$. Also, n_u is a neighborhood-trust calculated from the perspective of u and accordingly it has its maximum value of one when $p = q$. Thus, it is reasonable to set $p \geq q$. Since $p \geq q$ should hold, in case that $q > p$, node u calculates D after resetting $q = p$.

The values of p and q are based on the distribution of the sensed data and can be pre-computed in advance by the network operator. For example, if the sensed data is expected to follow a normal distribution, then we can use the empirical rule for c = 1, 2, and 3 (p = 0.68, 0.95, and 0.997, respectively). Finding the corresponding range is simply a matter of calculating the average and standard deviation of the sensed or transmitted data. Ranges for distributions like the exponential and Pareto can be easily calculated for fixed values of p and q by using a CDF based on the mean.

We note that the KL-divergence is computed over the data from a single time slot, and therefore only the information collected for one time slot needs to be stored. Moreover, the

mean and standard deviation of the information processed per time slot can be incrementally calculated in such a way that the mean and standard deviation are updated each time a piece of information is processed.

Let node v be the TA in time slot T_i. Node u sends v a neighborhood-trust report that is defined as $\{u||n_u||MAC_{K_{uv}}(u||n_u)\}$, where MAC stands for Message Authentication Code and K_{uv} is the shared secret key between the nodes u and v. Upon receiving a neighborhood-trust report from u, node v verifies the authenticity of u's neighborhood-trust report with K_{uv} and discards the report if it is not authentic. v collects the neighborhood-trust reports that were measured during T_i from all nodes in zone Z and aggregates the received neighborhood-trusts by using the *mean* function. We call the aggregated version of neighborhood-trusts the *zone-trust*. v sends the base station Z's zone-trust report, defined as $\{v||s_v||Z||t||MAC_{K_v}(v||s_v||Z||t)\}$, where s_v is a timestamp indicating the generation time of report, t is Z's zone-trust and K_v is the shared secret key between v and the base station.

3) Phase III: Detection and Revocation: Upon receiving a zone-trust report from a TA in zone Z, the base station verifies the authenticity of the TA's report with the secret shared key between TA and itself and the freshness of the timestamp, and the base station discards the report if it is not authentic or contains a stale timestamp. The base station also maintains a record per TA associating each TA's ID with its home zone and timestamp. This prevents the compromised TAs from claiming multiple home zones and from launching replay attacks with benign zone-trust reports. We denote the authentic reports from the TAs in zone Z by $R_1, R_2, \ldots$. The base station extracts the zone trust information t_i from report R_i. Let τ be a trust threshold and B_i denote a Bernoulli random variable defined as:

$$B_i = \begin{cases} 1 & \text{if } t_i < \tau \\ 0 & \text{if } t_i \geq \tau \end{cases}$$

The success probability ρ of Bernoulli distribution is defined as

$$\rho = \Pr(B_i = 1) = 1 - \Pr(B_i = 0) \tag{1}$$

If ρ is smaller than or equal to a trust threshold ρ', it is likely that the zone Z is trustworthy. On the contrary, if $\rho > \rho'$, it is likely that the zone Z is untrustworthy. The problem of deciding whether Z is trustworthy or not can be formulated as a hypothesis testing problem with null and alternate hypotheses of $\rho \leq \rho_0$ and $\rho \geq \rho_1$, respectively, such that $\rho_0 < \rho_1$. In this problem, the acceptance of the alternate hypothesis is considered to be a false positive error when $\rho \leq \rho_0$, and the acceptance of the null hypothesis is considered to be a false negative error when $\rho \geq \rho_1$. We define user-configured false positive rate α' and false negative rate β' in order to provide upper bounds on the false positives and false negatives in the hypothesis testing problem. These upper bounds will be presented in the security analysis in the next subsection.

We now describe how the SPRT is used to make a decision about zone Z from the n observed samples, where trust information t_i is treated as a sample. Let us define H_0 as the null hypothesis that zone Z is trustworthy and H_1 as the alternate hypothesis that zone Z is untrustworthy. We then define L_n as the log-probability ratio on n samples, given as:

$$L_n = \ln \frac{\Pr(B_1, \ldots, B_n | H_1)}{\Pr(B_1, \ldots, B_n | H_0)}$$

We assume that each genuine trust measurement for a given zone is independent of the other genuine trust measurements. This is a reasonable assumption in the sense that each benign node independently sends and receives data and control messages to and from its neighboring benign nodes for each time slot, and thus the measured zone trust values are independent of each other. However, this assumption is not applied to false trust measurement. This is because false trust values are artificially generated by compromised nodes and thus they are not likely be independent and identically distributed, but highly likely be correlated with each other. Accordingly, we assume that B_i is independent and identically distributed. Then L_n can be rewritten as:

$$L_n = \ln \frac{\prod_{i=1}^{n} \Pr(B_i|H_1)}{\prod_{i=1}^{n} \Pr(B_i|H_0)} = \sum_{i=1}^{n} \ln \frac{\Pr(B_i|H_1)}{\Pr(B_i|H_0)}$$

Let ω_n denote the number of times that $B_i = 1$ in the n samples. Then we have:

$$L_n = \omega_n \ln \frac{\rho_1}{\rho_0} + (n - \omega_n) \ln \frac{1-\rho_1}{1-\rho_0}$$

where $\rho_0 = \Pr(B_i = 1|H_0)$, $\rho_1 = \Pr(B_i = 1|H_1)$. The rationale behind the configuration of ρ_0 and ρ_1 is as follows: ρ_0 should be configured in accordance with the likelihood of the occurrence that the trustworthy zone is determined to have low trust value due to neighborhood-trust measurement error; ρ_1 should be configured to take into consideration the likelihood of the occurrence that an untrustworthy zone is determined to have low trust value.

On the basis of the log-probability ratio L_n, the SPRT for H_0 against H_1 is given as follows:

- $\omega_n \leq \lambda_0(n)$: accept H_0 and terminate the test.
- $\omega_n \geq \lambda_1(n)$: accept H_1 and terminate the test.
- $\lambda_0(n) < \omega_n < \lambda_1(n)$: continue the test process with another observation.

Where:

$$\lambda_0(n) = \frac{\ln \frac{\beta'}{1-\alpha'} + n \ln \frac{1-\rho_0}{1-\rho_1}}{\ln \frac{\rho_1}{\rho_0} - \ln \frac{1-\rho_1}{1-\rho_0}}, \quad \lambda_1(n) = \frac{\ln \frac{1-\beta'}{\alpha'} + n \ln \frac{1-\rho_0}{1-\rho_1}}{\ln \frac{\rho_1}{\rho_0} - \ln \frac{1-\rho_1}{1-\rho_0}}$$

If a zone Z is judged as trustworthy, the base station restarts the SPRT with newly arrived zone-trust reports. If, however, Z is determined to be untrustworthy, the base station terminates the SPRT on Z, and the network operator detects and revokes the compromised nodes by having nodes in other zones perform software attestation against sensor nodes in zone Z. We can use any software attestation techniques proposed in [1], [15], [17]. The main idea of these techniques is to detect the subverted parts in the flash image codes by checking whether the tested image codes match with the original image codes.

We use *mean* function for neighborhood-trust aggregation. As Wagner points out, the mean function is insecure for

data aggregation because the compromised nodes can force aggregator to generate false mean value by sending false data values to aggregator [21]. However, the mean function is reasonably secure for neighborhood-trust aggregation and is actually a *better* choice than the median. Certainly, if the number of attackers is less than half of the nodes in the zone, they will raise the average neighborhood-trust value more than they raise the median. However, in this case, we show through security analysis (Section III-B) and simulation (Section V) that the scheme still detects the attacker with very low error rates. In the case that the number of attackers is half or more of the nodes in the zone, then the median will now be at least as high as the lowest neighborhood-trust score submitted by an attacker node. In that case, the attacker is sure to go without being detected as he can set the neighborhood-trust value to the maximum of 1.0. In our scheme with Biased SPRT (see Section IV-A), using the mean continues to detect the attacker even when the fraction of attackers in the zone is more than half. Essentially, as long as the trust threshold can be configured high enough without incurring too many false positives, the mean suffices as the aggregation function.

Our scheme should be robust to unreliable communication in zone-trust report process. If zone-trust reports are frequently lost due to unreliable communication between trust aggregators and the base station, it will badly affect the decision process. For example, it could lead to a delay in decision process. Also, it could make the base station misidentify untrustworthy (resp. trustworthy) zone as trustworthy (resp. untrustworthy). To some extent, this issue is orthogonal to our problem; secure routing and robust packet delivery schemes [11], [24] are likely to be necessary in a sensor network with serious security concerns. However, we note that the attacker can aim to block honest reports from reaching the base station. To address this, we have the base station track the reporting from each TA in the zone; this is also necessary for ensuring equal reporting frequency. If the multiple nodes in a zone are having their reports blocked, the base station will treat the zone as untrustworthy and initiate software attestation on the zone. If, during software attestation, the non-reporting TAs are found to be malfunctioning or out of power, the base station can remove them from the list of TAs to prevent future false positives.

B. Security Analysis

In this section, we will first describe the detection accuracy of our proposed scheme and then discuss possible attack approaches and the defense strategies against them. Finally, we will present a game-theoretic analysis to show that the attacker's gain is greatly restrained by the defense strategy over any number of time slots.

1) Detection Accuracy: In the SPRT, α and β are the false positive and false negative probabilities. Moreover, α' and β' are user-configured values for these probabilities. According to Wald's theory [22], the upper bounds are given by $\alpha \leq \frac{\alpha'}{1-\beta'}, \beta \leq \frac{\beta'}{1-\alpha'}$ and the inequality $\alpha + \beta \leq \alpha' + \beta'$ also holds. Since β is the false negative probability, $(1-\beta)$ is the probability of correctly detecting an untrustworthy zone. Thus, the lower bound on the untrustworthy zone detection probability will be $(1-\beta) \geq \frac{1-\alpha'-\beta'}{1-\alpha'}$. Therefore, lower values of user-configured false positive and negative rates will lead to a lower false negative rate for the sequential test process, thus leading to higher detection rates. For instance, if the user configures both α' and β' to 0.01, then the detection of untrustworthy zone is guaranteed with probability of 0.99.

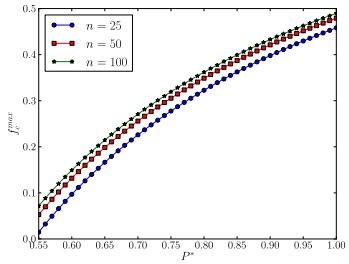

Fig. 1: f_c^{max} vs. P^* when $\alpha' = 0.01$ and $\beta' = 0.01$, $\rho_0 = 0.1$ and $\rho_1 = 0.9$.

2) Limitations of Node Compromise Attacks: We first consider the *false zone-trust report attack* in which the compromised TAs report false zone-trust values to the base station. This attack can take two forms. First, the compromised TAs could send reports of low zone-trust values to the base station when the zone-trust is actually high. Since this attack leads to quicker detection of the compromised TAs, the attacker will not benefit from this approach. Second, the compromised TAs could send reports of high zone-trust values to the base station when the zone-trust is actually low. In this type of attack, the attacker will have all compromised TAs report a zone-trust of 1.0 to the base station in order to prevent the untrustworthy zone from being detected. We investigate the impact of this attack on the detection capability of our scheme. For this investigation, we look into the impact of the fraction of compromised nodes in a zone on the detection capability of our scheme through the following Lemma. Recall that B_i is a Bernoulli random variable indicating whether the zone-trust report is below ($B_i = 1$) or (equal to or above) ($B_i = 0$) the trust threshold τ.

Lemma 3.1: Let f_c denote the fraction of compromised nodes in an untrustworthy zone. Let P^* denote the minimum value of $Pr(B_i = 1)$ in an untrustworthy zone, where $i = 1, 2, \ldots, n$. Given n samples such that

$$n > \frac{\ln \frac{1-\beta'}{\alpha'}}{P^*(\ln \frac{\rho_1}{\rho_0} - \ln \frac{1-\rho_1}{1-\rho_0}) - \ln \frac{1-\rho_0}{1-\rho_1}}$$

and

$$P^* > \frac{\ln \frac{1-\rho_0}{1-\rho_1}}{\ln \frac{\rho_1}{\rho_0} - \ln \frac{1-\rho_1}{1-\rho_0}},$$

an untrustworthy zone is detected as long as

$$0 < f_c \leq f_c^{max} = 1 - \frac{\ln \frac{1-\beta'}{\alpha'} + n \ln \frac{1-\rho_0}{1-\rho_1}}{nP^*(\ln \frac{\rho_1}{\rho_0} - \ln \frac{1-\rho_1}{1-\rho_0})}.$$

Proof: Since all compromised TAs in an untrustworthy zone generate a zone-trust of 1.0, only non-compromised TAs

in the untrustworthy zone report correct zone-trusts to the base station. Hence, $\omega_n = (1 - f_c)\sum_{i=1}^{n} Pr(B_i = 1) \geq n(1 - f_c)P^*$. If the conditions given in the lemma for n, $P*$, and f_c hold, then it is clear that $\omega_n \geq \lambda_1(n)$. Finally, if $\omega_n \geq \lambda_1(n)$, then the SPRT accepts an alternate hypothesis H_1 and accordingly the untrustworthy zone is detected. ■

Now we study analytically how P^* and n affect f_c^{max} as shown in Figure 1. For this study, we set $\alpha' = 0.01$, $\beta' = 0.01$ and $\rho_0 = 0.1$, $\rho_1 = 0.9$ while varying n from 25 to 100. In this configuration, $f_c^{max} = 1 - \frac{\ln 99 + n \ln 9}{2nP^* \ln 9}$ holds under the conditions that $P^* > \frac{1}{2}$ and $n > \frac{\ln 99}{2P^* \ln 9 - \ln 9}$. When $P^* = 1$, f_c^{max} approaches 0.5 as n increases. This means that our scheme will fail to detect untrustworthy zones using a false zone-trust report attack if more than half of the nodes in a zone are compromised. In other words, our scheme is resilient against the false zone-trust report attack as long as the fraction of compromised nodes is at most 50%. As shown in Figure 1, we see that f_c^{max} tends to rise as P^* increases in all cases of n. This means that a rise in P^* makes the scheme be resilient against an increase in the fraction of the compromised nodes. For a given value of P^*, we observe that a small value of n^* is achieved for low values of f_c^{max}. This indicates that the SPRT accepts H_1 in fewer samples as the fraction of the compromised nodes in a zone decreases.

Another important attack to address is the *reporting rule violation attack* in which the compromised TAs in a zone violate the rule of reporting zone-trust values in only their respective duty time slots by sending fake zone-trust reports to the base station more frequently. In particular, if the attacker has his compromised TAs report fake zone-trusts to the base station at very fast rate, it will cause the base station to make a wrong decision according to the fake zone-trust values even though the base station receives genuine zone-trust reports from benign TAs in their correct duty time slots. We can detect this attack by simply keeping track of the number of reports received in a window of time containing W time slots. We make two assumptions: (i) the network delivery time is much less than the length of one time slot; and (ii) the clock skew during the W time slots is less than one time slot. Given this, the number of reports should always be in the range $[W - 1, W + 1]$. If the number of reports is ever more than expected, the zone should be considered untrustworthy.

Finally, the attacker can launch a *neighboring zone attack* in which the compromised nodes do not perform malicious activities in their home zone Z but in a neighboring zone Z' that falls within their communication range. In this attack, the compromised nodes do not participate in the zone-trust report process so as to conceal their locations from the base station. Accordingly, even if the base station decides that the neighboring zone Z' is untrustworthy and the network operator performs software attestation against the neighboring zone Z', it will fail to detect these compromised nodes since they are not in Z'. To defend against this attack, the base station makes a list of the nodes residing in zone Z' by recording the IDs of the TAs when it receives their zone-trust reports. After each ID has been seen once, which can be determined when IDs start to appear a second time in their respective TA duty slots, it then sends this list to every node in zone Z'. Upon receiving this list from the base station, each node u in zone Z' performs software attestation against any neighbor node v that is not in this list. Note that, when there is no attacker using the neighboring zone attack, the list is very likely to be empty or very short for most nodes. The main purpose of this attestation is to verify that neighbor nodes, which are suspected of being malicious, are really compromised. Hence, we would say that the attestation operation for the verification purpose is imperative and does not cause large overhead. If v is found to be malicious, u halts all communication with v. Moreover, u informs the nodes in zone Z that v is malicious. Thus, v will also be attested and detected by nodes in Z and will be isolated from the network. Accordingly, the compromised node v will fail to affect the foreign zone Z' as well as its home zone Z.

3) Game-Theoretic Analysis: The above analysis demonstrated that our scheme achieves robust detection capability, unless a substantial number of nodes in a zone are compromised. Also, we described some attack strategies to disrupt the scheme and defense strategies against them. However, the above analysis does not reflect the interactions between the attacker and the defender. A study on these interactions is necessary to see how much the defender restrains the attacker's benefits through untrustworthy zone detection over a long period of time. To meet this requirement, we now present a game theoretic model of zone-trust information and the SPRT-based decision process. Since the defender decides the trustworthiness of zone by running the SPRT with samples of zone-trust information, this model fully captures the interactions between the attacker and defender.

In particular, we formulate this game-theoretic problem as a two-player repeated game with complete information, where the the first player is the *defender* and the second player is an *attacker*. We assume that two players are rational in the sense that they choose their strategies in such a way as to maximize their benefits. To fully understand the interactions between two players, it is reasonable to investigate the interactions for a long period of time. Therefore, we believe that the repeated game with complete information is suitable model for our analysis. In Table I, we summarize the notations that are frequently used in our game theoretic analysis.

Let N_z denote the number of sensor nodes residing in the zone and accordingly denote the sensor nodes in the zone by $S_1, S_2, \ldots, S_{N_z}$. Let f_c denote the fraction of compromised nodes in a zone. In Lemma 3.1, we prove that our scheme is robust as long as f_c is at most f_c^{max}. Since game-theoretic analysis only makes sense when our scheme is robust against false zone-trust report attack, it is reasonable to set the maximum fraction of comprised nodes to f_c^{max} given by Lemma 3.1. Let c_t denote the costs that all nodes in an untrustworthy zone pay for being attested by the network operator. Let c_r denote the costs that the network operator pays for attesting all nodes in an untrustworthy zone and revoking the compromised nodes.

Assume that each node S_i $(1 \leq i \leq N_z)$ in the kth $(k \geq 1)$ time slot generates the information falling within the range $[\mu_{i,k} - c\sigma_{i,k}, \mu_{i,k} + c\sigma_{i,k}]$ with probability $p_{i,k}$, where $\mu_{i,k}$ and $\sigma_{i,k}$ are the mean and standard deviation of the information generated by S_i in the kth time slot, and c is constant value.

TABLE I: Notations in our game-theoretic analysis.

N_z	number of nodes in a zone
S_i	a node in a zone ($1 \leq i \leq N_z$)
f_c	fraction of compromised nodes in a zone
f_c^{max}	maximum fraction of compromised nodes in a zone
$p_{i,k}$	probability of sensor S_i generating information in the expected range in the kth time slot
$q_{i,k}$	probability of sensor S_i receiving information in the expected range from a neighboring node in the kth time slot
$A_{i,k}$	amount of information that a node S_i receives and processes from its neighbors in the kth time slot
A_{max}	maximum amount of information that a node can receive and process from its neighbors in a time slot
G_k	the attacker's gain during the kth time slot, in terms of the average amount of false information injected into a malicious zone
c_t	costs that all nodes in an untrustworthy zone pay for being attested by the network operator
c_r	costs that the network operator pay for attesting all nodes in an untrustworthy zone and revoking the compromised nodes
I_k	indicator of untrustworthy zone attestation and revocation in the kth time slot
ψ_k	$(q_{1,k}, A_{1,k}), \ldots, (q_{N_z,k}, A_{N_z,k})$.
τ_k	trust threshold in the kth time slot ($0 \leq \tau_k \leq 1$)

TABLE II: Strategies and limit-of-means payoffs of the players.

	Strategies	Limit-of-means payoffs
Player 1	$\tau_1, \tau_2, \ldots$	$\lim_{T\to\infty} \frac{1}{T} \sum_{k=1}^{T} -G_k - I_k c_r$
Player 2	$f_c, \psi'_1, \psi'_2, \ldots$	$\lim_{T\to\infty} \frac{1}{T} \sum_{k=1}^{T} G_k - I_k c_t$

Also assume that each node S_i in the kth time slot receives the information generated by its neighbors and that the information falls within the range $[\mu_{i,k} - c\sigma_{i,k}, \mu_{i,k} + c\sigma_{i,k}]$ with probability $q_{i,k}$. Thus, $1 - q_{i,k}$ is the probability of a benign node receiving information outside of the expected information range. Let $A_{i,k}$ denote the total amount of information that a node S_i can receive and process from its neighbors in the kth time slot. Let A_{max} denote the maximum value of $A_{i,k}$. We denote $\psi_k = (q_{1,k}, A_{1,k}), \ldots, (q_{N_z,k}, A_{N_z,k})$.

We now describe each player's strategies. The attacker's possible strategies can be expressed as the setting of f_c and $\psi'_k = (q'_{1,k}, A'_{1,k}), \ldots, (q'_{N_z,k}, A'_{N_z,k})$ in the kth time slot. The attacker has his compromised nodes inject false information into a zone. We call a zone with compromised nodes *malicious* zone. His goal is to maximize the average amount of false information to be injected into malicious zone while not being detected by the scheme. He achieves this goal by controlling f_c, $q'_{i,k}$, and $A'_{i,k}$ such that $f_c > 0$ and $A'_{i,k} \geq A_{i,k}$. The defender's strategy can be expressed as the trust threshold τ_k in the kth time slot. His goal is to quickly detect untrustworthy zones and thus minimize the average amount of false information injected. He achieves this goal by controlling τ_k such that $0 \leq \tau_k \leq 1$.

To model the payoffs of the attacker and defender for an extended period of time, we use the limit-of-means payoff as in [20]. In this model, each player's long term payoff is expressed in terms of the expected payoff per time slot. Thus, it can converge to a certain value when the number of time slots goes to infinity. To define the limit-of-means payoff of each player, we denote the following notations. Let G_k denote the attacker's gain in the kth time slot, measured by the average amount of false information that the compromised nodes inject by employing strategies f_c and ψ'_k. We consider false information in accordance with the expected range. False information falling outside of (resp. in) the expected range is regarded as the gain (resp. loss) to the attacker. Thus, the average amount of false information to be injected is represented as the average amount of false information falling outside of the expected range minus the one falling in the expected range.

If a malicious zone is determined to be untrustworthy at the end of the $(k-1)$th time slot such that $\omega_{k-1} \geq \lambda_1(k-1)$, attacker's gain will be zero from the kth time slot to the rest of time slots since the compromised nodes in the malicious zone are detected and revoked. Assuming no deliberately injected false information, we expect node S_i to receive on an average $\sum_{i=1}^{N_z}(1 - q_{i,k})A_{i,k}$ pieces of information that fall outside of $[\mu_{i,k} - c\sigma_{i,k}, \mu_{i,k} + c\sigma_{i,k}]$ in the kth time slot. Hence, if $\omega_{k-1} \geq \lambda_1(k-1)$,

$$G_k = G_{k+1} = \cdots = 0.0.$$

Otherwise,

$$\begin{aligned} G_k &= \sum_{i=1}^{N_z}((1 - q'_{i,k})A'_{i,k} - (1 - q_{i,k})A_{i,k}) \\ &- (q'_{i,k}A'_{i,k} - q_{i,k}A_{i,k}) \\ &= \sum_{i=1}^{N_z}(1 - 2q'_{i,k})A'_{i,k} - (1 - 2q_{i,k})A_{i,k} \end{aligned}$$

We define I_k to associate one-time attestation and revocation costs with untrustworthy zone detection. I_1 is initialized to 0.0. If a zone is determined to be untrustworthy at the end of the $(k-1)$th time slot ($\omega_{k-1} \geq \lambda_1(k-1)$), $I_k = 1.0, I_{k+1} = I_{k+2} = \cdots = 0.0$. Otherwise, $I_k = 0.0$.

By using G_k and I_k, we express the limit-of-means payoffs of the attacker and defender as follows. The defender will lose as much as G_k if he fails to detect untrustworthy zone in the kth time slot. Additionally, he needs to pay one-time attestation and revocation costs if he detects untrustworthy zone. Therefore, his payoff in the kth time slot is given by

$-G_k - I_k c_r$. On the other hand, the attacker will get as much gain as G_k, but pay one-time attestation costs if it is detected as an untrustworthy zone. As a consequence, the attacker's payoff is given by $G_k - I_k c_t$. The strategies and the limit-of-means payoffs of the players are summarized in Table II in which player 1 and 2 indicate the defender and attacker, respectively.

To derive the optimal strategies for the attacker and defender, we first define the following notations. Let us denote $\tau^* = \tau_1^*, \tau_2^*, \ldots$ such that $\tau_1^* = \tau_2^* = \cdots = 1$ and denote $\psi^* = \psi_1^*, \psi_2^*, \ldots, \psi_k^*, \ldots$. We also denote $f_c^* = f_c^{max}$.

Definition 3.1: We define $\psi_k^* = (q_{1,k}^*, A_{1,k}^*)$, …, $(q_{i,k}^*, A_{i,k}^*)$, …, $(q_{N_z,k}^*, A_{N_z,k}^*)$ $(1 \le i \le N_z)$ in accordance with whether the trust aggregator (TA) is malicious. When the TA is compromised in the kth time slot such that $\omega_{k-1} < \lambda_1(k-1)$, $q_{i,k}^* = \frac{q_{i,k} A_{i,k}}{A_{max}}$ and $A_{i,k}^* = A_{max}$. Recall that B_i indicates whether the zone-trust report is below the trust threshold τ and that the SPRT does not decide that the zone is untrustworthy until the condition for H_1 acceptance, $\omega_k \ge \lambda_1(k)$ holds. We assume, as a worst case, that the attacker can accurately simulate the base station's SPRT based on his injected false information. Then, when the TA is not compromised in the kth time slot such that $\omega_{k-1} < \lambda_1(k-1)$, we define $q_{i,k}^*$ and $A_{i,k}^*$ as follows:

$$q_{i,k}^* = \begin{cases} \frac{q_{i,k} A_{i,k}}{A_{max}} & \text{if } \omega_k < \lambda_1(k) \text{ holds when simulating} \\ & B_k = 1 \text{ in the } (k-1)\text{th time slot.} \\ p_{i,k} & \text{if } \omega_k \ge \lambda_1(k) \text{ holds when simulating} \\ & B_k = 1 \text{ in the } (k-1)\text{th time slot.} \end{cases}$$

$$A_{i,k}^* = \begin{cases} A_{max} & \text{if } \omega_k < \lambda_1(k) \text{ holds when simulating} \\ & B_k = 1 \text{ in the } (k-1)\text{th time slot.} \\ A_{i,k}(\frac{1-q_{i,k}}{1-p_{i,k}}) & \text{if } \omega_k \ge \lambda_1(k) \text{ holds when simulating} \\ & B_k = 1 \text{ in the } (k-1)\text{th time slot.} \end{cases}$$

Theorem 3.1: The strategy τ^* is optimal for the defender. The strategies f_c^* and ψ^* are optimal for the attacker.

Proof: Since $0 \le f_c \le f_c^{max}$, the attacker maximizes his payoff when $f_c = f_c^{max}$. Thus, $f_c^* = f_c^{max}$ is the optimal strategy for the attacker.

In the limit-of-means payoffs for both the attacker and the defender, the second terms of the equations are $I_k c_t$ and $I_k c_r$, respectively. Since I_k is included in the second terms of both players, and c_t and c_r are regarded as constant costs, these second terms do not affect our derivation of the optimal strategies for either player. Hence, the game can be converted to a minimax game as follows:

$$\max_{f_c, \psi_1', \psi_2', \ldots} \min_{\tau_1, \tau_2, \ldots} U = \lim_{T \to \infty} \frac{1}{T} \sum_{k=1}^{T} G_k \tag{2}$$

In this minimax game, the defender's goal is to minimize the value of U. A higher value of τ_k contributes to a higher chance of untrustworthy zone detection. Also, as the likelihood that a malicious zone is detected as untrustworthy increases, U will decrease. Thus, U reaches its minimum value when $\tau_1 = \tau_2 = \cdots = 1$. Recall that $\tau^* = \tau_1^*, \tau_2^*, \ldots$ such that $\tau_1^* = \tau_2^* = \cdots = 1$. Thus, τ^* is the optimal strategy for the defender. Let us define U_{min} as the minimum value of U when τ^* is employed.

The attacker's goal is to maximize the value of U_{min}. According to the definition of G_k, if $\omega_{k-1} \ge \lambda_1(k-1)$, $G_k = G_{k+1} = \cdots = 0.0$ hold and thus U_{min} will go to zero as T goes to infinity. Accordingly, U_{min} reaches its maximum value when the value of $G_k > 0$ is maximized for every time slot k such that $\omega_k < \lambda_1(k)$. To achieve the maximum value of U_{min}, the attacker should set $q'_{i,k}$ and $A'_{i,k}$ in such a way as to obtain the maximum value of $G_k > 0$ while satisfying the condition that $\omega_k < \lambda_1(k)$ for every time slot k. We consider three sub-cases for $G_k > 0$ as follows:

- *Sub-case 1: TA is compromised in the kth time slot such that $\omega_{k-1} < \lambda_1(k-1)$.*
  In this sub-case, the compromised TA sends a zone-trust of 1.0 to the base station, leading to $\omega_k < \lambda_1(k)$ under the defender's optimal strategy of $\tau_k = \tau_k^* = 1$. Accordingly, in order to obtain the maximum value of G_k, the attacker injects the maximum possible amount of false information falling outside of the expected range for each node S_i in the kth time slot, $A_{max} - A_{i,k}$, into the malicious zone and thus has $A'_{i,k} = A_{max} - A_{i,k} + A_{i,k} = A_{max}$, while not injecting any false information falling in the expected range and accordingly having $A'_{i,k} q'_{i,k} = A_{i,k} q_{i,k}$. Hence, the attacker's optimal strategies are $q'_{i,k} = \frac{q_{i,k} A_{i,k}}{A_{max}}$ and $A'_{i,k} = A_{max}$.
- *Sub-case 2: TA is benign in the kth time slot such that $\omega_{k-1} < \lambda_1(k-1)$ and $\omega_k < \lambda_1(k)$ holds when simulating the configuration of $B_k = 1$ in the $(k-1)$th time slot.*
  In this sub-case, $\omega_k < \lambda_1(k)$ holds under the defender's optimal strategy of $\tau_k = \tau_k^* = 1$. Therefore, as in sub-case 1, G_k reaches its maximum value when $q'_{i,k} = \frac{q_{i,k} A_{i,k}}{A_{max}}$ and $A'_{i,k} = A_{max}$, which are the optimal strategies for the attacker.
- *Sub-case 3: TA is benign in the kth time slot such that $\omega_{k-1} < \lambda_1(k-1)$ and $\omega_k \ge \lambda_1(k)$ holds when simulating the configuration of $B_k = 1$ in the $(k-1)$th time slot.*
  In this sub-case, if $q'_{i,k} = p_{i,k}$, the zone-trust value would be set to one in the kth time slot and thus $B_k = 0$ would hold, leading to $\omega_k < \lambda_1(k)$ under the defender's optimal strategy of $\tau_k = \tau_k^* = 1$. Hence, $q'_{i,k} = p_{i,k}$ is the optimal strategy for the attacker. Recall that $p_{i,k}$ and $q_{i,k}$ are defined such that $p_{i,k} \ge q_{i,k}$ in Section III-A. To achieve the optimal strategy $q'_{i,k} = p_{i,k}$ under the condition that $p_{i,k} \ge q_{i,k}$, the attacker should inject δ amount of false information in the expected range for each node S_i in the kth time slot while not injecting any false information out of the expected range, leading to $A'_{i,k} = A_{i,k} + \delta$. Since $A'_{i,k}$ is associated with $q'_{i,k}$, $A'_{i,k}$ is the optimal strategy for the attacker. According to the definition of $q'_{i,k}$, we have $q'_{i,k} = \frac{A_{i,k} q_{i,k} + \delta}{A'_{i,k}} = \frac{A_{i,k} q_{i,k} + \delta}{A_{i,k} + \delta}$ and accordingly $\delta = \frac{A_{i,k}(q'_{i,k} - q_{i,k})}{1 - q'_{i,k}}$. By plugging δ into $A'_{i,k} = A_{i,k} + \delta$, we obtain $A'_{i,k} = A_{i,k}(\frac{1-q_{i,k}}{1-q'_{i,k}}) = A_{i,k}(\frac{1-q_{i,k}}{1-p_{i,k}})$.

By Definition 3.1, the values of $q'_{i,k}$ and $A'_{i,k}$ that make G_k have maximum value while making $\omega_k < \lambda_1(k)$ hold in the three sub-cases are $q_{i,k}^*$ and $A_{i,k}^*$, respectively. Recall that $\psi^* = \psi_1^*, \psi_2^*, \ldots, \psi_k^*, \ldots$. Therefore, ψ^* is the optimal

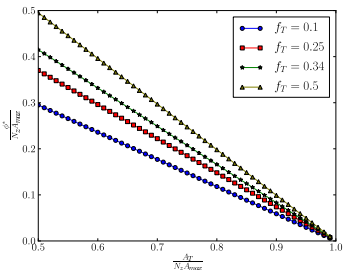

Fig. 2: Attacker gain $\frac{\phi^*}{N_z A_{max}}$ vs. network activity $\frac{A_T}{N_z A_{max}}$ when $f_c^{max} = 0.49$, $\alpha' = 0.01$ and $\beta' = 0.01$, $\rho_0 = 0.1$ and $\rho_1 = 0.9$.

strategy for the attacker.

■

When the attacker and defender follow their respective optimal strategies, $I_k = 0$ holds for all $k \geq 1$ because malicious zone will never be determined as untrustworthy. Thus, the attacker's payoff under the Nash Equilibrium is, per time slot:

$$\begin{aligned}\phi &= \lim_{T\to\infty} \frac{1}{T} \sum_{k=1}^{T} G_k \\ &= \lim_{T\to\infty} \frac{1}{T} \sum_{k=1}^{T} \sum_{i=1}^{N_z} (1-2q_{i,k}^*)A_{i,k}^* - (1-2q_{i,k})A_{i,k}\end{aligned}$$

Since the attacker's gain ϕ is the defender's loss when $I_k = 0$ for all $k \geq 1$ according to Table II, we only focus on ϕ in this analysis. Let f_T denote the fraction of time slots in which sub-case 2 holds in the T time slots. Also, f_c^{max} indicates the fraction of time slots in which sub-case 1 holds in the T time slots since the optimal fraction of compromised nodes is $f_c^* = f_c^{max}$. Since $(1-2q_{i,k}^*)A_{i,k}^* - (1-2q_{i,k})A_{i,k} = A_{max} - A_{ik}$ holds in sub-cases 1 and 2, and $(1-2q_{i,k}^*)A_{i,k}^* - (1-2q_{i,k})A_{i,k} = (1-2p_{i,k})(\frac{1-q_{i,k}}{1-p_{i,k}})A_{i,k} - (1-2q_{i,k})A_{i,k}$ holds in sub-case 3, we have:

$$\begin{aligned}\phi &= \lim_{T\to\infty} \frac{1}{T} \sum_{k=1}^{T} \sum_{i=1}^{N_z} (f_c^{max} + f_T)(A_{max} - A_{i,k}) \\ &+ (1 - f_c^{max} - f_T)((1-2p_{i,k})(\frac{1-q_{i,k}}{1-p_{i,k}})A_{i,k} \\ &- (1-2q_{i,k})A_{i,k})\end{aligned}$$

Since $p_{i,k}$ and $q_{i,k}$ are defined such that $p_{i,k} > \frac{1}{2}$ and $p_{i,k} \geq q_{i,k}$ in Section III-A, $((1-2p_{i,k})(\frac{1-q_{i,k}}{1-p_{i,k}}) - (1-2q_{i,k}))A_{i,k} \leq 0$ holds in sub-case 3. Thus, sub-case 3 actually represents the loss for the attacker. As a consequence, the attacker obtains $\phi^* = (f_c^{max} + f_T) \lim_{T\to\infty} \frac{1}{T} \sum_{k=1}^{T} \sum_{i=1}^{N_z} (A_{max} - A_{i,k})$ gain in the best case where there is zero loss in sub-case 3. Since the best case in the attacker's gain is the worst case to the defender and the worst case analysis is efficient and effective to evaluate the defender's resilience under the Nash Equilibrium, we perform analysis with ϕ^* instead of ϕ. In the Nash Equilibrium, $B_k = 1$ holds in sub-case 2. Thus $f_T \times T$ indicates the number of time slots in which $B_k = 1$. Since the zone should not be detected as untrustworthy in the Nash Equilibrium, $\omega_T = f_T T < \lambda_1(T)$ should hold. Accordingly, $0 < f_T < \frac{\lambda_1(T)}{T}$ should hold. In the Nash Equilibrium, we investigate how much the attacker gains under different values of f_c^{max}, f_T, and $A_T = \lim_{T\to\infty} \frac{1}{T} \sum_{k=1}^{T} \sum_{i=1}^{N_z} A_{i,k}$. A_T indicates the total amount of information that all nodes in the zone would receive from their neighbors and process per time slot if compromised nodes do not inject any false information. For this investigation, we set $\alpha' = 0.01$, $\beta' = 0.01$ and $\rho_0 = 0.1$, $\rho_1 = 0.9$. Moreover, we set $T = 100$ for the configuration of f_T such that $0 < f_T < 0.51$. In these configurations, the maximum value of f_c^{max} is calculated as 0.49. We set $f_c^{max} = 0.49$ to study the attacker's gain under the worst case of the maximum value of f_c^{max}.

We also use a normalized gain $\frac{\phi^*}{N_z A_{max}} = \frac{(f_c^{max}+f_T)(N_z A_{max} - A_T)}{N_z A_{max}}$ ranging from 0 to 1. As shown in Figure 2, we observe that $\frac{\phi^*}{N_z A_{max}}$ decreases as $\frac{A_T}{N_z A_{max}}$ increases. This means that the smaller the gap between A_T and A_{max}, the less gain the attacker gets. In particular, the attacker's normalized gain will be kept under half of the maximum gain as long as A_T is more than half of $N_z \times A_{max}$, irrespective of f_T. We also see that a decrease in f_T leads to a decline in $\frac{\phi^*}{N_z A_{max}}$. This means that the attacker will gain less as the likelihood of sub-case 2 decreases. The attacker can achieve high $f_T = 0.5$ if he can set the duty time slots of the compromised TAs in such a way that the duty time slot of a compromised TA is immediately followed by the one of the benign TAs, leading to an increase in the likelihood of the occurrence of $\omega_k < \lambda_1(k)$. However, each TA's duty time slots are determined in a pseudorandom manner; it is thus difficult for the attacker to control the duty time slots of the compromised nodes in such a way as to fulfill high $f_T = 0.5$. Therefore, the attacker's normalized gain under the Nash Equilibrium will be substantially limited due to relatively low values of f_T.

C. Performance Analysis

We now analyze the performance of our scheme in terms of the average number of samples needed to detect untrustworthy zones, as well as the overheads for communication, computation, storage, and attestation.

1) Average Number of Samples for Decision: We begin by studying how many samples are required on an average for the base station to decide whether a zone is trustworthy or not.

Let n denote the number of samples needed to terminate the SPRT. Since n varies with the types of samples, it is treated as a random variable with expected value $E[n]$. According to [22], $E[n]$ is obtained as follows:

$$E[n] = \frac{E[L_n]}{E\left[\ln \frac{\Pr(B_i|H_1)}{\Pr(B_i|H_0)}\right]} \tag{3}$$

From Equation 3, we compute the expected values of n

conditioned on the hypotheses H_0 and H_1 as follows:

$$E[n|H_0] = \frac{(1-\alpha')\ln\frac{\beta'}{1-\alpha'} + \alpha'\ln\frac{1-\beta'}{\alpha'}}{\rho_0\ln\frac{\rho_1}{\rho_0} + (1-\rho_0)\ln\frac{1-\rho_1}{1-\rho_0}}$$

$$E[n|H_1] = \frac{\beta'\ln\frac{\beta'}{1-\alpha'} + (1-\beta')\ln\frac{1-\beta'}{\alpha'}}{\rho_1\ln\frac{\rho_1}{\rho_0} + (1-\rho_1)\ln\frac{1-\rho_1}{1-\rho_0}} \quad (4)$$

From the equations on $E[n|H_0]$ and $E[n|H_1]$, we see that the number of samples tends to increase in proportion to ρ_0, given fixed values of $\rho_1 = 0.7, 0.9$. We also see that small values of ρ_0 help detect untrustworthy and trustworthy zones with a small number of reports. For a given value of ρ_0, $E[n|H_0]$ and $E[n|H_1]$ are larger when $\rho_1 = 0.7$ than when $\rho_1 = 0.9$. This means that large values of ρ_1 reduce the number of reports required for untrustworthy and trustworthy zone detection.

2) Communication Overhead: We define the communication overhead as the average number of zone-trust reports and neighborhood-trusts that are sent or forwarded by the sensor nodes in the network. Assume that the network is partitioned into z zones. The base station receives at most z zone-trust reports per time slot. Since the average hop distance between two randomly chosen nodes is given by $O(\sqrt{N})$ [16], where N is the total number of sensor nodes, the average number of zone-trust reports will be at most $O(z \times \sqrt{N})$ per time slot. Assume that there are b nodes on an average within a zone. Under the optimal zone size that is described in Section III-A, every node can directly communicate with each other. Thus, $b-1$ neighborhood-trusts in a zone on an average are sent to a TA in each time slot. Since there are z zones, the average number of neighborhood-trusts sent by nodes will be $z(b-1)$ per time slot. Hence, the communication overhead will be at most $O(z \times \sqrt{N}) + z(b-1)$ per time slot.

3) Computation and Storage Overhead: We define the computation and storage overhead as the average number of Message Authentication Codes (MACs) that are generated and verified by a node and the average number of zone-trust reports that need to be stored by a node, respectively. Assume that there are b nodes on an average within a zone. In a zone, every node acts as the TA in its designated time slot while acting as a zone member in the other time slots. For each time slot, each zone member generates a MAC of its neighborhood-trust report, which is sent to a TA. Each TA in turn performs $b-1$ MAC verifications on the received neighborhood-trust reports and generates a MAC of its zone-trust report. Thus, b nodes perform $2b-1$ MAC generations and verifications every time slot. Accordingly, the computation overhead per node will be $O(1)$ per time slot on an average. The base station will perform z MAC verifications every time slot because z TAs report their zone-trusts each time slot.

There is no storage overhead for the sensor nodes, as they do not need to keep any reports. However, the base station needs to extract the ID, home zone, and timestamp information from zone-trust reports and store them per node in order to prevent replay attacks, leading to $O(N)$ storage overhead.

4) Attestation Overhead: We define attestation overhead as the average number of software attestations that are performed per time slot. We assume that there are z zones in the network and that $\frac{N}{z}$ nodes are placed per zone on average. Overheads are more important in the common case, in which nodes are not compromised.[2] In this case, attestations are only performed after a false positive, when a zone is mis-identified as being untrustworthy. The false positive probability α is given by $\alpha \leq \frac{\alpha'}{1-\beta'}$. Recall that α can be set to be a low value by setting α' and β' appropriately; $\alpha = 0.01$ is a reasonable value.

Recall that the SPRT requires $E[n|H_1]$ time slots on average to decide that a zone is untrustworthy; $E[n|H_1]$ is given by Equation 4. Since all nodes in a zone are attested only when zone is decided to be untrustworthy, attestation overhead is calculated as $\alpha \times z \times \frac{N}{z} \times (1/E[n|H_1]) = \alpha \times N \times (1/E[n|H_1])$. Since at least one time slot is required for untrustworthy zone detection, the maximum value of $(1/E[n|H_1])$ is 1. Thus, the attestation overhead in the worst case is rewritten as $\alpha \times N$.

Now we compare the attestation overhead of our scheme to the ones of the related works. For this comparison, we classify the related works into two categories: *centralized*, *decentralized*. In centralized approaches [1], [15], [17], each sensor node is attested by the base station or the server. Thus, one software attestation per time slot is required for each node and attestation overhead is exactly N attestations per time slot. Centralized software attestation also requires communication overhead of $O(\sqrt{N})$ per attestation, for a total of $O(N\sqrt{N})$. Our overheads are the same, but multiplied by the constant α. For $\alpha = 0.01$, this is a reduction of two orders of magnitude.

In the decentralized approach [23], each sensor node is attested by its neighboring nodes. Thus, b software attestations per time slot are required for each node, where b is the average number of neighbors, and attestation overhead is computed as $b \times N$ per time slot, which is $\frac{b}{\alpha}$ times our system's attestation overhead. For $\alpha = 0.01, 10 \leq b < 100$, our system reduces the attestation overhead of the decentralized approach by three orders of magnitude. The communication overhead is also $O(b \times N)$. In many systems, b and $\sqrt{N}$ are comparable and thus $b > \alpha\sqrt{N}$ holds, in which case our system's communication overhead is much less than the decentralized technique.

The main reason why our scheme outperforms the related works in terms of attestation overhead is because our scheme performs attestations only against nodes in zones that appear to be compromised, while the related works perform unconditional attestation against every node.

IV. Biased-SPRT

In Section III, we propose a SPRT-based node compromise detection and revocation scheme and analyze its security and performance. Although this scheme achieves fast and accurate node compromise detection and revocation, it will not work if more than 50% of the nodes in each zone are compromised under reasonable configurations of the SPRT. To enhance the resilience of the SPRT-based scheme against the false zone-trust report attack with a large number of compromised nodes,

[2] If the common case is that nodes are frequently compromised, many nodes will need to be revoked or manually restored regardless of the detection and revocation method. In such a scenario, the network would be prohibitively expensive to operate securely.

we modify the sampling strategy in the SPRT in such a way that the SPRT takes the samples leading to acceptance of H_0 (*high-trust samples*) with less weight than the ones leading to acceptance of H_1 (*low-trust samples*), while ensuring that the false positive rate remains below the desired rate. We call this modification *biased sampling* and the corresponding scheme as the *Biased-SPRT*. Since a high-trust sample is less likely to be accepted than a low-trust sample, H_1 will be more likely to be accepted when the zone is untrustworthy. Biased sampling results in greater delay in accepting the null hypothesis and greater false positive rates, but these are not major costs in the system as designed. Note that benign zones are continually tested for trust values, so there is no benefit to quickly detecting that the zone is trustworthy. Also, a false positive only costs the additional overhead of a single software attestation against the nodes in the zone. The benefit of biased sampling is that even a relatively small number of honest nodes can send zone-trust reports with low trust values, leading to detection. We will show that biased sampling improves the resilience of the proposed scheme against the false zone-trust report attack, even when the fraction of compromised nodes is more than 50%.

Let us first describe the biased sampling technique and then present its security and performance analyses.

A. Biased Sampling

In the biased sampling, the samples with type of H_0 are taken into the sequential test process with less weight than H_1 in such a way as to replace ρ_0 with $(\rho_0)^\epsilon$, where $\epsilon > 1$ is a biased sampling factor. Thus, the log-probability on n samples L_n is changed to:

$$L_n = \omega_n \ln \frac{\rho_1}{(\rho_0)^\epsilon} + (n - \omega_n) \ln \frac{1-\rho_1}{1-(\rho_0)^\epsilon}$$

Accordingly, the SPRT for H_0 against H_1 is changed to:

- $\omega_n \leq \lambda_0(n)$: accept H_0 and terminate the test.
- $\omega_n \geq \lambda_1(n)$: accept H_1 and terminate the test.
- $\lambda_0(n) < \omega_n < \lambda_1(n)$: continue the test process with another observation.

Where:

$$\lambda_0(n) = \frac{\ln \frac{\beta'}{1-\alpha'} + n \ln \frac{1-(\rho_0)^\epsilon}{1-\rho_1}}{\ln \frac{\rho_1}{(\rho_0)^\epsilon} - \ln \frac{1-\rho_1}{1-(\rho_0)^\epsilon}}$$

$$\lambda_1(n) = \frac{\ln \frac{1-\beta'}{\alpha'} + n \ln \frac{1-(\rho_0)^\epsilon}{1-\rho_1}}{\ln \frac{\rho_1}{(\rho_0)^\epsilon} - \ln \frac{1-\rho_1}{1-(\rho_0)^\epsilon}}$$

An increase in ϵ leads to faster acceptance of H_1 but slower acceptance of H_0. Intuitively, we can imagine the SPRT's one-dimensional random walk taking larger steps towards H_1 and smaller steps towards H_0. This modification has several consequences. First, untrustworthy zones will be more likely and quickly to be detected; a few low-trust reports will lead to acceptance of H_1. Second, even a zone with a majority of compromised nodes that send false zone-trust reports can be detected as untrustworthy. Despite a number of false high-trust reports that cause the SPRT to take small steps towards H_0, the regular presence of true low-trust reports will take bigger steps towards H_1. Finally, trustworthy zones will have a higher chance to be incorrectly detected as untrustworthy. Because of this last point, it is important to configure ϵ in such a way as to enhance the resilience against the false zone-trust report attack while maintaining the desired false positive rate. We discuss how ϵ impacts the false positive rate in Section IV-B.

In general, we believe that Biased-SPRT can be useful whenever it is more important to quickly and reliably reach one decision (in our case, untrustworthy zone detection) than another decision (trustworthy zone detection), under the condition that a balance can be achieved in conformity with the error rate for incorrect decisions (false positives).

B. Security Analysis

In this section, we first investigate how ϵ affects the false positive rate of the Biased-SPRT and then show how much the system's resilience against the false zone-trust report attack can be enhanced by the Biased-SPRT. Finally, we will demonstrate that the Biased-SPRT causes the attacker to have substantially lower gains than the ones in the SPRT in terms of our game-theoretic analysis.

1) False Positive Rate: To examine how the biased sampling factor ϵ affects the false positive rate α, we use the estimated value of α instead of the upper bound derived in Section III-B, because the estimated value will contribute to more a accurate investigation than the upper bound. According to [22], α in the SPRT is approximately estimated as follows:

$$\alpha \sim \frac{1 - (\frac{\beta'}{1-\alpha'})^h}{(\frac{1-\beta'}{\alpha'})^h - (\frac{\beta'}{1-\alpha'})^h} \tag{5}$$

where $h = h(\rho)$ and ρ is given by

$$\rho = \frac{1 - (\frac{1-\rho_1}{1-\rho_0})^h}{(\frac{\rho_1}{\rho_0})^h - (\frac{1-\rho_1}{1-\rho_0})^h} \tag{6}$$

By replacing ρ_0 with $(\rho_0)^\epsilon$ in Equations 5 and 6, we can obtain the estimated value of the α in the Biased-SPRT. Since the success probability ρ in the Bernoulli distribution is a parameter required to estimate α, we take into account ρ for the study of ϵ's impact on α. We consider two cases: $\alpha' = \beta' = 0.01$ and $\alpha' = \beta' = 0.05$. We also examine these two cases: $\rho_0 = 0.1$ and $\rho_1 = 0.9$, $\rho = 0.3$ and $\rho_1 = 0.7$. Moreover, we set ϵ to 1, 2, or 3 such that $\epsilon = 1$ and $\epsilon = 2, 3$ indicate the SPRT and Biased-SPRT, respectively. As shown in Figure 3, α decreases as h increases and as α' and β' decrease. As shown in Figures 4 and 5, ρ diminishes as h and ρ_1 increase and as ρ_0 decreases. We also notice that a rise in ϵ contributes to a decrease in ρ when h is fixed. Given that $h = 1.0$, $\alpha' = \beta' = 0.01$, and $\rho_0 = 0.1$ and $\rho_1 = 0.9$ (Case I), we have $\alpha \sim 0.01$ and $\rho = 0.1, 0.01, 0.001$ when $\epsilon = 1, 2, 3$, respectively. This means that the false positive rates are estimated as 1% as long as a zone-trust value is measured to be less than trust threshold with probability of 0.1 (resp. 0.01, 0.001) when the SPRT (resp. Biased-SPRT) is employed. Thus, the Biased-SPRT requires ρ to be lower than the one in the SPRT in order to achieve high detection accuracy. Given

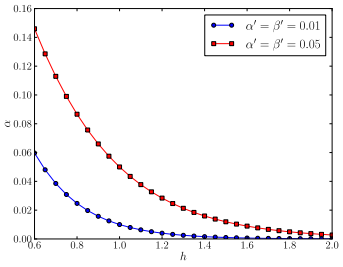

Fig. 3: α vs. h.

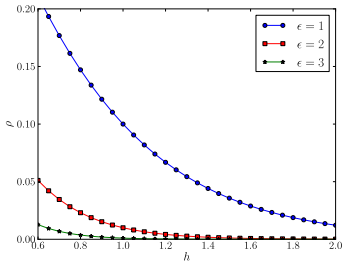

Fig. 4: ρ vs. h when $\rho_0 = 0.1$ and $\rho_1 = 0.9$.

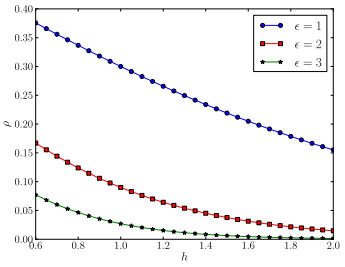

Fig. 5: ρ vs. h when $\rho_0 = 0.3$ and $\rho_1 = 0.7$.

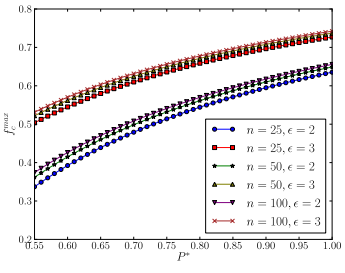

Fig. 6: f_c^{max} vs. P^* when $\alpha' = 0.01$ and $\beta' = 0.01$, $\rho_0 = 0.1$ and $\rho_1 = 0.9$.

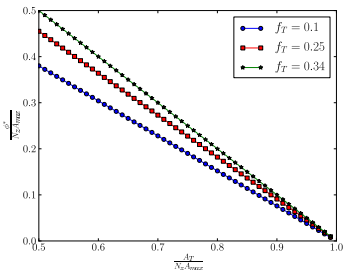

Fig. 7: $\frac{\phi^*}{N_z A_{max}}$ vs. $\frac{A_T}{N_z A_{max}}$ when $\epsilon = 2$, $f_c^{max} = 0.66$, $\alpha' = 0.01$ and $\beta' = 0.01$, $\rho_0 = 0.1$ and $\rho_1 = 0.9$.

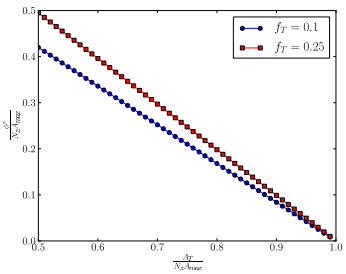

Fig. 8: $\frac{\phi^*}{N_z A_{max}}$ vs. $\frac{A_T}{N_z A_{max}}$ when $\epsilon = 3$, $f_c^{max} = 0.74$, $\alpha' = 0.01$ and $\beta' = 0.01$, $\rho_0 = 0.1$ and $\rho_1 = 0.9$.

that $h = 0.6$, $\alpha' = \beta' = 0.01$, and $\rho_0 = 0.1$ and $\rho_1 = 0.9$ (Case II), we have $\alpha \sim 0.06$ and $\rho = 0.05, 0.013$ when $\epsilon = 2, 3$, respectively. This indicates that the Biased-SPRT with lower value of h can achieved less but still reasonable false positive rates at higher value of ρ. When ρ_0 and ρ_1 in Cases I and II are configured to a larger value (0.3) and a smaller value (0.7) respectively, we see that the Biased-SPRT fulfills the same false positive rates as Cases I and II while requiring higher value of ρ than Cases I and II.

2) Limitations of Node Compromise Attacks: The Biased-SPRT does not affect the countermeasures against the reporting rule violation and neighboring zone attacks discussed in Section III-B and thus they are used without change when the Biased-SPRT is employed. However, we need to provide a new analysis on the false zone-trust report attack since a biased sampling factor ϵ impacts on the resilience of the proposed scheme against that attack. To meet this need, we replace ρ_0 with $(\rho_0)^\epsilon$ in Lemma 3.1 and show that our scheme is robust as long as f_c is at most $f_c^{max} = 1 - \frac{\ln \frac{1-\beta'}{\alpha'} + n \ln \frac{1-(\rho_0)^\epsilon}{1-\rho_1}}{nP^*(\ln \frac{\rho_1}{(\rho_0)^\epsilon} - \ln \frac{1-\rho_1}{1-(\rho_0)^\epsilon})}$ under conditions that $n > \frac{\ln \frac{1-\beta'}{\alpha'}}{P^*(\ln \frac{\rho_1}{(\rho_0)^\epsilon} - \ln \frac{1-\rho_1}{1-(\rho_0)^\epsilon}) - \ln \frac{1-(\rho_0)^\epsilon}{1-\rho_1}}$ and $P^* > \frac{\ln \frac{1-(\rho_0)^\epsilon}{1-\rho_1}}{\ln \frac{\rho_1}{(\rho_0)^\epsilon} - \ln \frac{1-\rho_1}{1-(\rho_0)^\epsilon}}$.

Now we analyze how ϵ, P^*, and n affect f_c^{max} as shown in Figure 6. For this study, we set $\alpha' = 0.01$, $\beta' = 0.01$ and $\rho_0 = 0.1$, $\rho_1 = 0.9$ while increasing ϵ from 2 to 3 and n from 25 to 100. In this configuration, when $\epsilon = 2$, $f_c^{max} = 1 - \frac{\ln 99 + n \ln 9.9}{nP^*(\ln 90 - \ln \frac{10}{99})}$ holds under the conditions that $P^* > 0.3375$ and $n > \frac{\ln 99}{P^*(\ln 90 - \ln \frac{10}{99}) - \ln 9.9}$. When $\epsilon = 3$, $f_c^{max} = 1 - \frac{\ln 99 + n \ln 9.99}{nP^*(\ln 900 - \ln \frac{100}{999})}$ holds under the conditions that $P^* > 0.2528$ and $n > \frac{\ln 99}{P^*(\ln 900 - \ln \frac{100}{999}) - \ln 9.99}$. We observe that increases in ϵ, P^*, and n result in the increase in f_c^{max}. In case of $P^* = 1$, f_c^{max} becomes 0.663, 0.747 as n increases when $\epsilon = 2, 3$, respectively. This indicates that the Biased-SPRT is resilient against false zone-trust report attack as long as the fraction of compromised nodes is at most 66.3% and 74.7% when $\epsilon = 2, 3$, respectively. Note that the maximum value of f_c^{max} is 0.5 when $P^* = 1$ in the SPRT. Thus, the Biased-SPRT substantially improves the resilience of the SPRT by 32.6% and 49.4% when $\epsilon = 2, 3$, respectively.

3) Game-Theoretic Analysis: The Nash Equilibrium derived in Section III-B also holds in the Biased-SPRT. In the Nash Equilibrium, however, the attacker's gain will be different from the one in Section III-B since a new parameter ϵ is added to the gain. Thus, we only focus our investigation on how f_c^{max}, f_T, and ϵ affect the attacker's gain from malicious zone in the Nash Equilibrium. For this study, we consider the attacker's gain (ϕ^*) in the best case as in Section III-B. For this study, we set $\alpha' = 0.01$, $\beta' = 0.01$ and $\rho_0 = 0.1$, $\rho_1 = 0.9$, $T = 100$. We also configure $\epsilon = 2, 3$. When $\epsilon = 2, 3$, we have $0 < f_0 \leq 0.344$ and $0 < f_c^{max} \leq 0.656$, $0 < f_0 \leq 0.258$ and $0 < f_c^{max} \leq 0.742$, respectively. We set $f_c^{max} = 0.656, 0.742$ to study the attacker's normalized gain ($\frac{\phi^*}{N_z A_{max}}$) under the worst case of the maximum value of f_c^{max}. Figures 7 and 8 show how the attacker's normalized gain is affected by $\frac{A_T}{N_z A_{max}}$. The attacker's normalized gain tends to decrease as $\frac{A_T}{N_z A_{max}}$ increases. This means that the attacker achieves less normalized gain as more information is generated in the zone. We observe that the increase in ϵ and

f_c^{max} lead to a slight growth in the attacker's normalized gain. Overall, the attacker gets slightly more normalized gains when compared to the SPRT. However, the attacker's normalized gains are still restricted below 0.5 as long as $\frac{A_T}{N_z A_{max}} \geq 0.5$. Therefore, the Biased-SPRT greatly limits the attacker's gain as in the SPRT even if a major fraction (0.656 $\sim$ 0.742) of nodes are compromised in the zone.

C. Performance Analysis

The Biased-SPRT requires the same communication, computation, and storage overheads as the ones of the SPRT, because it only changes the sampling strategy of the SPRT and thus does not affect these overheads. However, the change in the sampling strategy affects the average number of samples and attestation overhead of the SPRT. Hence, we investigate the average number of samples and attestation overhead in the Biased-SPRT.

1) Average Number of Samples for Decision: By replacing ρ_0 with $(\rho_0)^\epsilon$ in $E[n|H_0]$ and $E[n|H_1]$ of the SPRT, we compute $E[n|H_0]$ and $E[n|H_1]$ as follows:

$$E[n|H_0] = \frac{(1-\alpha')\ln\frac{\beta'}{1-\alpha'} + \alpha'\ln\frac{1-\beta'}{\alpha'}}{(\rho_0)^\epsilon \ln\frac{\rho_1}{(\rho_0)^\epsilon} + (1-(\rho_0)^\epsilon)\ln\frac{1-\rho_1}{1-(\rho_0)^\epsilon}}$$

$$E[n|H_1] = \frac{\beta'\ln\frac{\beta'}{1-\alpha'} + (1-\beta')\ln\frac{1-\beta'}{\alpha'}}{\rho_1 \ln\frac{\rho_1}{(\rho_0)^\epsilon} + (1-\rho_1)\ln\frac{1-\rho_1}{1-(\rho_0)^\epsilon}} \quad (7)$$

From the equations on $E[n|H_0]$ and $E[n|H_1]$, we observe that small values of ρ_0, and large values of ϵ and ρ_1 contribute to the detection of untrustworthy and trustworthy zones with a few number of reports. Given the values of ρ_0 and ρ_1, we perceive that the expected number of samples in the Biased-SPRT are at most 6 while the ones in the SPRT are at most 13. Moreover, we see that the growth rates of $E[n|H_0]$ and $E[n|H_1]$ in the Biased-SPRT are much slower than the ones in the SPRT. From these observations, we see that $\epsilon \geq 2$ plays a major role in restricting the growth rate of the average number of samples required for untrustworthy and trustworthy zone detection.

2) Attestation Overhead: By substituting $(\rho_0)^\epsilon$ for ρ_0 in the attestation overhead of the SPRT, we obtain the attestation overhead in the Biased-SPRT. Since $(\rho_0)^\epsilon$ does not affect the worst case overhead of the SPRT, the Biased-SPRT has the same worst case attestation overhead as the SPRT.

V. Simulation Study

In this section, we will first describe our simulation experimental environment and then discuss the simulation results.

A. Simulation Environment

We developed a simple simulation program to evaluate the SPRT and the Biased-SPRT. We simulate data generation and exchange by having nodes randomly generate data and exchange them with other nodes in the same zone. Every benign sensor node uses the normal distribution for generating data. Specifically, we first set the global data mean $\mu = 100$, global data deviation variable $\chi = 5$, and local standard deviation $\sigma = 5$. Each benign sensor node v then selects local data mean μ_v uniformly at random from the range $[\mu-\chi, \mu+\chi]$ and generates data in accordance with the normal distribution $N(\mu_v, \sigma)$. Compromised nodes generate data from a range that excludes all points in $[\mu-\chi, \mu+\chi]$. We place 100 nodes in a zone. For each time slot, each benign node and each compromised node determine the number of data values to generate uniformly at random from the range $[0, 1000]$ and $[0, 5000]$, respectively. We set $c = 1$ and $p = 0.68$ to compute the KL-divergence described in Section III-A.

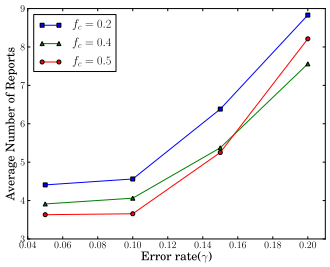

Fig. 9: Average number of reports vs. error rate (γ) in case of the SPRT without false zone-trust report attack.

TABLE III: Simulation Parameter Values.

	SPRT				Biased-SPRT			
γ	0.05	0.1	0.15	0.2	0.05	0.1	0.15	0.2
τ	0.9	0.9	0.85	0.8	0.9	0.88	0.83	0.78

We also set both the user-configured false positive threshold $\alpha = 0.01$ and the false negative threshold $\beta = 0.01$, and we set $\rho_0 = 0.1$ and $\rho_1 = 0.9$, respectively. The rationale behind these configurations is discussed in Section III-C. To emulate the zone-trust measurement errors caused by the inaccuracy of neighborhood-trust measurement, we modify the measured zone-trusts with error rate γ. Specifically, we take zone-trust t measured through perfect neighborhood-trust measurement and generate zone-trust t' selected uniformly at random from the range $[t - t\gamma, t + t\gamma]$. We set $\gamma = 0.05, 0.1, 0.15, 0.2$. As shown in Table III, when there is no false zone-trust report attack, we configure a trust threshold τ in accordance with γ. When there is false zone-trust report attack, we use the same values as in Table III while excluding the cases of $\gamma = 0.2$. Note that τ is configured in such a way that it decreases as γ increases. This is reasonable because the rise of error rate likely results in the decrease of zone-trust values. Moreover, the cases of $\gamma = 0.2$ is excluded when there is false zone-trust report attack. This is because the false zone-trust values can be regarded as if they were generated due to high measurement errors. Thus the case of high error rate can be thought of as already being reflected in false zone-trust report attack and it is reasonable to exclude the case of high error rate $\gamma = 0.2$. We also set $f_c = 0.0, 0.2, 0.4, 0.5$ in the SPRT, $f_c = 0.0, 0.5, 0.6, 0.65$ in the Biased-SPRT with $\epsilon = 2$, and $f_c = 0.0, 0.6, 0.7, 0.74$ in the Biased-SPRT with $\epsilon = 3$. A zone is regarded as trustworthy when $f_c = 0.0$, and untrustworthy when $f_c > 0$.

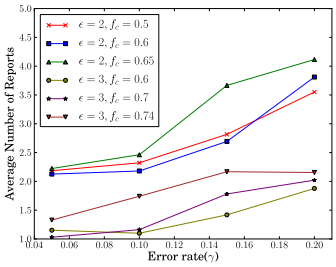

Fig. 10: Average number of reports vs. error rate (γ) in case of the Biased-SPRT without false zone-trust report attack.

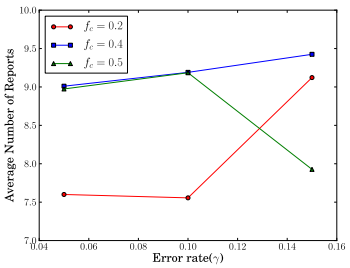

Fig. 11: Average number of reports vs. error rate (γ) in case of the SPRT with false zone-trust report attack.

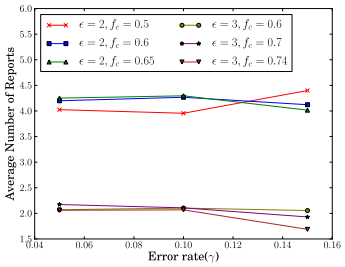

Fig. 12: Average number of reports vs. error rate (γ) in case of the Biased-SPRT with false zone-trust report attack.

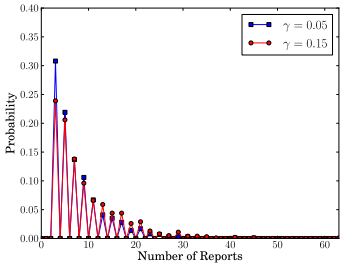

Fig. 13: Probability distribution of the number of reports in case of the SPRT with false zone-trust report attack ($f_c = 0.2$).

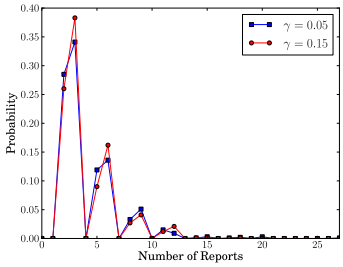

Fig. 14: Probability distribution of the number of reports in case of the Biased-SPRT with false zone-trust report attack ($\epsilon = 2, f_c = 0.6$).

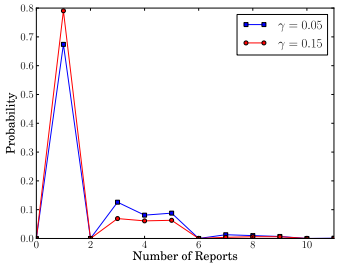

Fig. 15: Probability distribution of the number of reports in case of the Biased-SPRT with false zone-trust report attack ($\epsilon = 3, f_c = 0.74$).

B. Simulation Results

We use the following metrics to evaluate the performance of our scheme:

- *Number of Reports* is the number of zone-trust reports required for the base station to decide whether a zone is trustworthy or not.
- *False Positive* is the error probability that a trustworthy zone is misidentified as an untrustworthy zone.
- *False Negative* is the error probability that an untrustworthy zone is misidentified as a trustworthy zone.

We present the average results for 1000 runs of the simulation in each configuration. Each run is executed for 100 time slots. For each run, we obtain each metric as the average of the results of the SPRTs that are performed. Note that the SPRT will terminate if it determines that the zone is untrustworthy. We investigate the false positive and false negative rates and the number of reports.

First, we found that false positive rates in the SPRT and the Biased-SPRT were at most 1.2% and 0.9% for all values of γ. Specifically, the highest false positives in the SPRT and the Biased-SPRT were measured as 1.2% and 0.9% when $\gamma = 0.1$ and $\epsilon = 1$, $\gamma = 0.1$ and $\epsilon = 3$, respectively. In both the SPRT and the Biased-SPRT without false zone-trust report attack, there were no observed false negatives for any value of γ. In the SPRT with false zone-trust report attack, the false negative rates were at most 0.3% for all values of f_c and γ except the case of $f_c = 0.5$ and $\gamma = 0.15$ in which the false negative rate was 3.6%. In the Biased-SPRT ($\epsilon = 2$) with false zone-trust report attack, there were no observed false negatives for any values of f_c and γ except the case of $f_c = 0.65$ and $\gamma = 0.15$ in which the false negative rate was 1.0%. In the Biased-SPRT ($\epsilon = 3$) with false zone-trust report attack, there were no observed false negatives for any values of f_c and γ except the cases of $f_c = 0.74$ and $\gamma = 0.15$ in which the false negative rate was 0.2%. Thus, the trustworthy zones were misidentified as untrustworthy with at most probability of 0.009 (resp. 0.012) in the Biased-SPRT (resp. SPRT). Untrustworthy zones were misidentified as trustworthy with at most probability of 0.01 (resp. 0.036) in the Biased-SPRT (resp. SPRT). Hence, both the Biased-SPRT and the SPRT achieves high detection accuracy while the former enhances the detection accuracy of the latter.

Second, we present the results of the average number of reports for two cases. One case is that there are no compromised nodes ($f_c = 0.0$) in a zone and this zone is determined to be trustworthy. We call this case *trueNegative*. The other case is that there are compromised nodes ($f_c > 0.0$) in a zone, and this zone is determined to be untrustworthy. We call this case *truePositive*. In the trueNegative case, the average number of reports were between 3.0 and 3.475 in the SPRT. the average one was 3.0, 2.0 in the Biased-SPRT with $\epsilon = 2, 3$, respectively. This indicates that both the SPRT and the Biased-SPRT require a few number of reports on an average to decide on the untrustworthy zones. Figures 9 and 10 show the results of the truePositive cases in the SPRT and the Biased-SPRT when there is no false zone-trust report attack. Figures 11

and 12 show the same results when there is false zone-trust report attack. In the truePositive case, the average number of reports was at most 8.830 (resp. 4.115) in case of the SPRT (resp. Biased-SPRT) without false zone-trust report attack. It was at most 9.425 (resp. 4.399) in case of the SPRT (resp. Biased-SPRT) with false zone-trust report attack.

From these observations, we see that the base station detects untrustworthy zones with a small number of reports on an average in both the SPRT and the Biased-SPRT regardless of the fraction of compromised nodes. Given an error rate γ, the maximum value of the average number of reports in the Biased-SPRT is approximately half of that in the SPRT. This means that the Biased-SPRT reaches a decision with substantially less number of reports than the SPRT on an average. We also see that an increase in ϵ contributes to a reduction in the average number of reports in the Biased-SPRT. We perceive that the average number of reports tends to increase with the rise of γ when there is no false zone-trust report attack. This means that an increase in the neighborhood-trust measurement error rate leads to a rise in the average number of reports when there is no false zone-trust report attack. However, γ does not noticeably affect the average number of reports when there is false zone-trust report attack except the case of the SPRT with $f_c = 0.2$. We infer from this that false zone-trust reports are thought of as if they were generated due to high measurement error rates and thus the effects of high error rates are already reflected in false zone-trust report attack, as long as at least 40% of the nodes in the zone are compromised.

Finally, Figure 13 shows the probability distribution of the number of reports in the case of truePositive when $f_c = 0.2$ in the SPRT with false zone-trust report attack. Figures 14 and 15 show the one when $f_c = 0.6, \epsilon = 2$ and $f_c = 0.74, \epsilon = 3$ respectively in the Biased-SPRT with false zone-trust report attack. For this distribution, we examine two scenarios of $\gamma = 0.05$ and $\gamma = 0.15$. When $f_c = 0.2$ in the SPRT, a total of 77% and 67.9% of the cases fall in the range from 3 to 9 reports in the case of $\gamma = 0.05$ and $\gamma = 0.15$, respectively. When $f_c = 0.6, \epsilon = 2$ in the Biased-SPRT, a total of 88.1% and 89.5% of the cases fall in the range from 2 to 6 reports in the case of $\gamma = 0.05$ and $\gamma = 0.15$, respectively. When $f_c = 0.74, \epsilon = 3$ in the Biased-SPRT, a total of 88.1% and 92.08% of the cases fall in the range from 1 to 4 reports in the case of $\gamma = 0.05$ and $\gamma = 0.15$, respectively From these observations, we see that in most cases, the number of reports is less than or close to the average and thus the SPRT and Biased-SPRT detect untrustworthy zones with at most nine and six reports in most cases when there is false zone-trust report attack, respectively. Furthermore, we notice that the probability distributions near the average are denser in the Biased-SPRT than the SPRT. We infer from this observation that the Biased-SPRT requires fewer number of reports in most cases than the SPRT when there is false zone-trust report attack.

These results provide further evidence that the compromise of a majority of the nodes in a zone will be challenging for the attacker. The attacker must be fast to compromise so many nodes in one zone without detection; when the Biased-SPRT is used, compromise of fewer than 74% of the nodes in a zone will lead to the detection in just a few reports in most cases. The network operator could use the expected minimum time to compromise a node, along with the number of nodes per zone, to help set a length for the period of time between the reports. If the estimates are accurate, this would prevent the attacker from compromising the majority of nodes in a zone without being detected.

VI. Related Work

There exist a number of works on the node compromise detection in wireless sensor networks. Software-attestation based schemes have been proposed to detect the subverted software modules of sensor nodes [1], [15], [17], [23]. Specifically, the base station checks whether the flash image codes have been maliciously altered by performing attestation in randomly chosen portions of image codes or the entire codes [1], [15], [17]. In [23], a sensor node's image codes are attested by its neighbors. However, all these schemes require every sensor to be attested and thus the benign sensor nodes will waste their resources for participating in attestations even though they do not need to be attested.

Reputation-based trust management schemes have been proposed to manage individual node's trust based on its actions [6], [13], [19]. Specifically, Ganeriwal et al. [6] proposed a reputation-based trust management scheme in which a Bayesian formulation is used to compute an individual node's trust. Sun et al. [19] proposed information theoretic frameworks for trust evaluation. Specifically, entropy-based and probability-based schemes have been proposed to compute an individual node's trust. Li et al. [13] leveraged node mobility to reduce an uncertainty in trust computation and speed up trust convergence. However, the malicious nodes are not easily revoked due to the risk of false positives under the presence of these trust management schemes.

The ID traceback schemes have been proposed to locate the malicious source of false data [25], [27]. However, they only trace a source of the data sent to the base station and thus do not locate the malicious sources that send false data or control messages to other benign nodes in the network.

Replica node detection schemes can also be considered as related works [7], [16]. In the replica node attacks, the attacker generates many replicas of a compromised node to reduce the time and effort needed to compromise the equivalent number of benign nodes. Parno et al. [16] proposed static replica node detection schemes by leveraging the intuition that replica nodes are placed in more than one location in the static sensor networks. Ho et al. [7] proposed a mobile replica node detection scheme by leveraging the intuition that replica nodes appear to move faster than benign nodes and thus highly likely exceed the predefined maximum speed in the mobile sensor networks. However, these schemes cannot be used to detect and revoke the compromised nodes, because the main intuitions for replica detection are not applied to node compromise detection.

VII. Conclusions

In this paper, we have proposed a zone-based node compromise detection and revocation scheme for sensor networks

using the Sequential Probability Ratio Test (SPRT). We also enhanced the robustness of the SPRT with biased sampling. We have shown that our scheme achieves robust untrustworthy zone detection capability even if a majority of nodes in each zone are compromised. Furthermore, we have proposed countermeasures against the attacks that might be launched to disrupt the proposed scheme. We also modeled the interaction between the defender and the adversary as a repeated game with complete information and found a Nash Equilibrium. We show that the defender greatly limits the gains of the adversary under the Nash Equilibrium. We evaluated the proposed scheme through simulation experiments under various scenarios. Our experimental results show that our scheme quickly detects untrustworthy zones with a small number of zone-trust reports.

VIII. Acknowledgements

This work was supported by a special research grant from Seoul Women's University (2012).

This work was supported in part by the National Science Foundation under award numbers IIS-0326505, CNS-0721951, and CNS-0916221, and the Air Force Office of Scientific Research under award number A9550-08-1-0260. The work of S. K. Das is also supported by (while serving at) the National Science Foundation. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] T. Abuhmed, N.Nyamaa, and D. Nyang. Software-Based Remote Code Attestation in Wireless Sensor Network. In *IEEE GLOBECOM*, December 2009.

[2] S. Čapkun and J.P. Hubaux. Secure positioning in wireless networks. *IEEE Journal on Selected Areas in Communications*, 24(2):221–232, February 2006.

[3] T.M. Cover and J.A. Thomas. Elements of information theory. *Wiley-Interscience*, 2006.

[4] F. Delgosha and F. Fekri. Threshold key-establishment in distributed sensor networks using a multivariate scheme. In *IEEE INFOCOM*, April 2006.

[5] W. Du, J. Deng, Y.S. Han, S. Chen, and P. Varshney. A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge. In *IEEE INFOCOM*, pages:586-597, March 2004.

[6] S. Ganeriwal and M. Srivastava. Reputation-based framework for high integrity sensor networks. In *ACM SASN*, October 2004.

[7] J. Ho, M. Wright, and S.K. Das. Fast Detection of Replica Node Attacks in Sensor Networks Using Sequential Analysis. In *IEEE INFOCOM*, April 2009.

[8] J. Ho, M. Wright, and S.K. Das. ZoneTrust: Fast Zone-Based Node Compromise Detection and Revocation in Sensor Networks Using Sequential Analysis. In *IEEE Symposium on Reliable Distributed Systems (SRDS)*, September 2009.

[9] X. Hu, T. Park, and K. G. Shin. Attack-tolerant time-synchronization in wireless sensor networks. In *IEEE INFOCOM*, April 2008.

[10] J. Jung, V. Paxon, A.W. Berger, and H. Balakrishnan. Fast port scan detection using sequential hypothesis testing. In *IEEE S&P*, May 2004.

[11] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: attacks and countermeasures. In *IEEE Workshop on Sensor Network Protocols and Applications*, May 2003.

[12] D. Knuth. The Art of Computer Programming vol. 2 (3rd ed.), pp. 145-146. *Addison-Wesley*, 1998.

[13] F. Li and J. Wu. Mobility reduces uncertainty in {MANET}. In *IEEE INFOCOM*, May 2007.

[14] Z. Li, W. Trappe, Y. Zhang, and B. Nath. Robust statistical methods for securing wireless localization in sensor networks. In *IEEE IPSN*, April 2005.

[15] T. Park and K. G. Shin. Soft tamper-proofing via program integrity verification in wireless sensor networks. *IEEE Trans. Mob. Comput.*, 4(3):297-309, 2005.

[16] B. Parno, A. Perrig, and V.D. Gligor. Distributed detection of node replication attacks in sensor networks. In *IEEE S&P*, May 2005.

[17] A. Seshadri, A. Perrig, L. van Doorn, and P. Khosla. SWATT: SoftWare-based ATTestation for embedded devices. In *IEEE S&P*, May 2004.

[18] K. Sun, P. Ning, C. Wang, A. Liu, and Y. Zhou. TinySeRSync: Secure and resilient time synchronization in wireless sensor networks. In *ACM CCS*, October 2006.

[19] Y. Sun, Z. Han, W. Yu, and K. Liu. A trust evaluation framework in distributed networks: vulnerability analysis and defense against attacks. In *IEEE INFOCOM*, April 2006.

[20] G. Theodorakopoulos and J. S. Baras. Game theoretic modeling of malicious users in collaborative networks. In *IEEE Journal on Selected Areas in Communications*, 26(7):1317 - 1326, 2008.

[21] D. Wagner. Resilient Aggregation in Sensor Networks. In *ACM SASN*, 2004.

[22] A. Wald. Sequential analysis. *Dover Publications*, 2004.

[23] Y. Yang, X. Wang, S. Zhu, and G. Cao. Distributed software-based attestation for node compromise detection in sensor networks. In *IEEE SRDS*, October 2007.

[24] F. Ye, G. Zhong, S. Lu, and L. Zhang. A robust data delivery protocol for large scale sensor networks. In *IEEE IPSN*, April 2003.

[25] F. Ye, H. Yang, and Z. Liu. Catching "moles" in sensor networks. In *IEEE ICDCS*, June 2007.

[26] W. Zhang, M. Tran, S. Zhu, and G. Cao. A random perturbation-based scheme for pairwise key establishment in sensor networks. In *ACM Mobihoc*, September 2007.

[27] Y. Zhang, J. Yang, L. Jin, and W. Li Locating compromised sensor nodes through incremental hashing authentication. In *DCOSS*, June 2006.