Comparing Trust Mechanisms for Monitoring Aggregator Nodes in Sensor Networks

Oly Mistry
The University of Tulsa
Tulsa, OK
oly-mistry@utulsa.edu

Anıl Gürsel
The University of Tulsa
Tulsa, OK
anil-gursel@utulsa.edu

Sandip Sen
The University of Tulsa
Tulsa, OK
sandip@utulsa.edu

ABSTRACT

Sensor nodes are often used to collect data from locations inaccessible or hazardous for humans. As they are not under normal supervision, these nodes are particularly susceptible to physical damage or remote tampering. Generally, a hierarchical data collection scheme is used by the sensors to report data to the base station. It is difficult to precisely identify and eliminate a tampered node in such a data collecting hierarchy. Most security schemes for sensor networks focuses on developing mechanism for nodes located higher in the hierarchy to monitor those located at lower levels. We propose a complementary mechanism with which the nodes at lower levels can monitor their parents in the hierarchy to detect malicious behavior. Every node maintains a reputation value of its parent and updates this at the end of every data reporting cycle. We propose a novel combination of statistical testing techniques and existing reputation management and reinforcement learning schemes to manage the reputation of a parent node. The probability that the parent node is malicious is calculated using various combination of the Q-learning algorithm and the β-Reputation scheme. The input to the β-Reputation scheme is a history of boolean events consisting of correct or erroneous data reporting events by the parent node. The boolean events are generated at each data reporting period using statistical tests. Our approach can be viewed as a mechanism composed of different modules for the detection of a malicious event, interpretation of the malicious event and updating node reputation value based on the interpretation. We have created different versions of our system by varying these components. We compared the effectiveness of these alternative designs in detecting different types of malicious behavior in sensor networks.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*

General Terms

Algorithms, Performance, Security

Cite as: Comparing Trust Mechanisms for Monitoring Aggregator Nodes in Sensor Networks, Oly Mistry, Anıl Gürsel & Sandip Sen, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. XXX-XXX.

Keywords

sensor networks, reputation, data integrity, learning

1. INTRODUCTION

With increasingly smaller sizes and reduced costs, the range of possible application domains for sensor networks has been expanding continually. Currently, sensor networks are used for diverse monitoring opportunities, such as fire detection, securing sensitive locations, detecting nuclear leakage, etc. An average sensor network consists of hundreds of node agents. Each node sends its individual sensing data to a *Base Station (BS)*. The BS processes the data and gives the final decision to alert the network operator or consider possible autonomous responses, e.g., shutting down the gas supply to a building in case of a fire.

Due to their inexpensive and small designs, the capability and resources of sensor nodes, including communication speed, memory space, battery, etc., are very limited. If every node sends its individual sensing value to the BS, through multiple hops, there will be a huge amount of network traffic resulting in network congestion and unnecessary consumption of valuable, limited power resources. In addition, since the nodes are sensing a common phenomenon, there will be a high redundancy of raw data. Hence, aggregation of raw data is often used to improve efficiency and lifetime of deployed sensor networks. The importance of effective data aggregation is well-recognized in the sensor network literature [6, 9]. As such, many data aggregation protocols have been proposed in the literature [4, 7, 19, 12, 3].

However, all these aggregation schemes assume that the sensor nodes are trustworthy and reporting data truthfully. In practice though, sensors are usually deployed in open and unattended environments, and hence are susceptible to physical tampering. When a node is compromised, the adversary can inject bogus or misleading data into the network. Traditional cryptographic techniques are not sufficient to address such security breaches in sensor network environments. In addition, the use of nodes with aggregation roles poses other integrity problems. In each aggregation, the individual sensing values are lost. If an aggregator node is compromised, its effect can be even more damaging than when an individual sensing node is compromised.

Research in sensor network security tries to identify malicious or faulty node agents using standard statistical techniques like outlier detection. When a node is identified to be malicious, it is either eliminated immediately from the network or at least its effect on the final result is reduced.

In current literature on sensor network security, the mon-

itoring for sensor node is almost exclusively performed in a top down process. As data flows from sensor nodes to the base station via intermediary nodes, a logical hierarchical structure is imposed on the network. A higher level node in such a hierarchy monitors data reported to it by its children. For example, a parent node can collect data reported to it by its children within a time interval and suspect a child node to be possibly compromised if it is sending values inconsistent with that reported by its siblings. If the same node is repeatedly suspected to be compromised, further data reported by it may be ignored. Unfortunately, this also means eliminating both that node and all of its descendants from the network unless a time and resource intensive restructuring of the hierarchy is performed. In particular, it is not easy for the parent node to identify if its immediate children or any of their descendants is responsible for the erroneous data. Hence, it is difficult to precisely pinpoint the actual node that is compromised in a top-down approach.

We propose a complementary, *bottom-up*, agent-based solution to this problem. We assign agents to each node that monitor the security status of the network. In our proposed solution, these child level node agents keep monitoring the aggregator level node agents for possible inconsistency in the data they forward. When a aggregator level node is independently identified as malicious by its children, it is marked as malicious and the information is sent to the base station. The marked node is then excluded from the hierarchy without eliminating any of its children. This is possible due to the bottom-up monitoring and precise identification of the compromised node.

We presume that each child level node in the hierarchy is capable of listening to the data forwarded by its parent to higher level after aggregation. We justify this assumption by referring to the fact that the aggregator will be located within the communication range of the child node. We further assumed that, it can store the history of such data reporting incidents for certain intervals, subject to their hardwire limitations. We have combined techniques from popular statistical methods and machine learning algorithms to develop a novel and robust malicious node detection mechanism. We have used statistical hypothesis tests to find out the boolean characterization of a data reporting event as *correct* or *erroneous*. Based on these history of boolean events, we calculate and update the current reputation values for the parent node agents using a reinforcement learning scheme. We have assumed that the data sensed by the children of an aggregator node are drawn from the distributions with a common mean. Our assumption is reasonable, as, being located in physical proximity, they can be assumed to be sensing similar data values from the environment and their sensed data items should be comparable. The reinforcement learning scheme used makes our mechanism robust and prevents false identification of non-malicious aggregators.

In the next section we describe related work on security management in sensor network and different types of reinforcement learning scheme in machine learning literature. In section 3 and 4, we describe our framework and different statistical techniques. In Section 5, we describe the machine learning approaches along with the insight on using them. In section 6, we present the different schemes we used and along with the experimental results and in section 7, we explain the crux of our algorithm and outline further research prospects.

2. RELATED WORK

A lot of work has been done in securing sensor network applications like key establishment, secrecy, authentication, robustness to denial-of-service attacks and secure routing. Most of this research have been based on the conventional cryptography techniques to protect confidentiality, integrity and availability. SPINS [15] implements symmetric key cryptography with delayed key disclosure to achieve asymmetric key cryptography. SPINS incorporate both Secure Network Encryption Protocol (SNEP) which provides data confidentiality, authentication, integrity freshness and μTESLA which provides authentication to broadcast. In addition, [2] and [5] proposed probabilistic key sharing mechanisms to detect the key distribution problem. However, sensor networks are commonly deployed in open and unattended areas, and as a consequence, vulnerable to physical tampering. Whenever a sensor node is compromised, the adversary can obtain its confidential keys and inject bogus information into the network without being detected. Thus, cryptography alone is not sufficient to protect sensor network environments.

Most work on data aggregation [4, 7, 19, 12, 3] assume trustworthy environments. SDAP [23] has proposed a secure aggregation approach in presence of dishonest sensors, which can detect and drop false reports. Nonetheless, in SDAP, only BS is responsible for the detection of deceitful nodes and hence it is not a distributed approach and is susceptible to single point of failure. Besides, in case of an attestation failure all data flowing from the marked group is discarded and the sensing value of many nodes are unnecessarily eliminated. Sampling techniques have been exercised by SIA [16] to calculate summary report even when a fraction of the node agents are malicious. SIA does not deal with per-hop data aggregation and BS is the only aggregator. It assumes all node agents send their raw reading values to BS. Hu and Evans [8] introduce a hop-by-hop secure data aggregation system, but their system works only if one node is compromised. On the other hand, only the lead node agents can have their own sensing values.

Interest in applying artificial intelligence techniques to securing sensor network environments is rising. Mukherjee and Sen [14] uses an offline neural network based learning technique to model spatial patterns in sensed data. The net is used to predict the sensed data while working on-line and according to differences between prediction and real sensor reading, the parent updates the reputation of each of its children. $Q - Learning$ and $\beta - reputation$ based approaches are used to detect faulty nodes. Their mechanism works in a top-down manner, so in case of a fault detection the node and all its descendants are eliminated. Additionally, all the patterns need to be learned before detection mechanism starts, which might not be applicable for all sensor network environments, such as nuclear leakage detection and fire detection.

Servin and Kudenko [18] applies reinforcement learning techniques for intrusion detection. Wu etal. [22], Ruairi and Keane [17] base their detection systems on multi-agent systems. The critical issue of aggregation, however, is not addressed by any of these researches.

In this research, our aim is to propose a mechanism that utilizes statistics and Artificial Intelligence techniques to detect malicious node agents in a sensor network environment without unnecessarily eliminating honest node agents,

e.g., the descendants of a malicious node. Our system supplements existing sensor network security approaches, e.g., cryptographic techniques and top-down monitoring.

3. PROBLEM DESCRIPTION

In sensor network environments, a malicious node agent might have two objectives: (a) introducing arbitrary noise in the final data reported to the base station, or (b) biasing the results to systematically report under or over-estimations of the actual environmental parameters being monitored. In the former case, the node agent introduces random errors to the system and in the latter case malicious nodes consistently add errors of the same sign. The range of such errors and hence the possible damage to the system can be limited by using outlier detection systems. The effect of individual malicious node agent, therefore, will be small on the final data reported to the BS. Multiple colluding node agents can have an effect only if they synchronize their reported errors. For example, if a number of colluding node agents are trying to prevent detection of fire, to outperform outlier detection schemes, all of them must coordinate to under-report the temperature. The effect of a malicious node on the data reported to the BS increases if the malicious node is closer to the base station in the hierarchy. We believe that a distributed, rather than a centralized, security envelop is indispensable for truly distributed systems like sensor networks to counter security threats. While almost all sensor network security work focuses on top down monitoring of child node agents by parents in the network, we investigate a complementary bottom-up approach where children monitor their parents for erroneous data reporting.

In our framework, node agents are arranged in a hierarchy. An aggregator node agent calculates the aggregation value from its own reading (if it is also sensing) and the values received from its child node agents. In this paper, we use the averaging function as the aggregation function. We assume local continuity in the environment. Hence, the data values sensed by a node agent, its siblings, and their parent are drawn from the same data distribution. We further assume that the children of a node agent can read the aggregated data forward by the parent node agent to node agents further upstream towards the base station. The research problem is then for a child node agent to detect consistent deviations form the true mean of the average value reported by the parent node. Note that this is difficult to do in general because a child node does not have access to the data reported by its siblings and hence the true mean.

We now specify the bottom-up malicious node agent detection problem more formally. Let us consider a node agent in the hierarchy with n children. The data sensed by the ith child at time t is represented by a random variable, X_i^t. The true aggregate of the reported values is represented by $\overline{X^t} = \sum_{i=1}^{n} X_i^t$. Let X^t be the data actually reported by the parent node as the aggregate value. The ith child node agent then has access to only the values of X_i^t and X^t. Its goal is to identify consistent deviations between X^t and $\overline{X^t}$. Though this cannot be done directly, we restate the problem of detecting consistent deviations between X_i^t and X^t to be an indirect indicator of the actual problem.

In case of directional broadcast communication, a malicious aggregator might try to send different values to its parent and its children. As public key encryption technology is commonly used in wireless sensor network environments [11, 21], to prevent such a situation, we present a verification protocol, that utilizes public key authentication scheme for the child nodes to confirm the value reported by their parent further downstream. To illustrate this we use a part of the network presented in Figure 1:

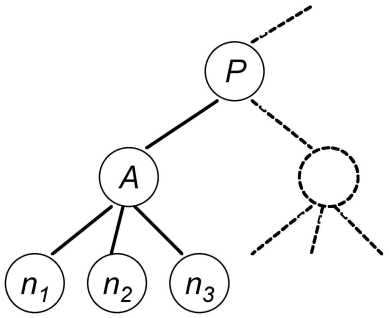

Figure 1: A sample topology.

1. An aggregator node agent A sends its aggregation value to its parent P as well as broadcasts it to its children $C_1, C_2, \ldots, C_n$.
2. When the parent node agent P receives the aggregation value, it signs it with its private key and sends both the raw and signed aggregation results to the aggregator node A.
3. The aggregator A forwards the two values raw and that signed by P, to its children $C_1, C_2, \ldots, C_n$.
4. Child node agents verify the aggregation value forwarded by A, by comparing it to the signed value received from P. Moreover, since P sends a signature as well, A cannot tamper the raw aggregation value sent by P, cause the children can unsign it by using the public key of P.

To summarize, the message transactions are as follows:

$A \rightarrow P$:	agg_val
$A \rightarrow n_1, n_2, n_3$:	agg_val
$P \rightarrow A$:	agg_val & Signature(agg_val)
$A \rightarrow n_1, n_2, n_3$:	agg_val & Signature(agg_val)

where agg_val is the aggregation value.

Note that we are not dealing with the secrecy of the aggregation values. Instead, we are concerned about the truthfulness of the aggregated value. Although we have not included any cryptographic technique for the privacy of messages, an actual system might use them.

4. STATISTICAL APPROACH

We consider the fact that the sensor node agents are monitoring real environment, so there will be variation in the data pattern sensed by the sensors even though they are in close proximity. Moreover, the sensors are not necessarily identically calibrated. So, same data can be sensed as different by two different sensors. Yet, when we find out the differences of sensed and reported data pattern, we should get unbiased difference values, *i.e.*, there will be as many positive difference as negative differences. Based on this assumption we use statistical techniques to determine malicious biases.

Depending on the outcome of the statistical tests, we develop a robust reputation mechanism to detect the malicious

aggregators. The outcomes of the statistical tests are interpreted in different ways in various configurations.

Real Interpretation: We have used the raw probability value obtained from the statistical tests as the current instant reputation value.

Boolean Interpretation: We have also used the *significance level* for the hypothesis tests as a threshold to produce a boolean value of *corret* or *erroneous* for each data reporting event. Based on these boolean events we calculate the current reputation value using the β-*reputation* scheme. This current reputation value is then used to update the actual reputation using the *Q- learning* scheme.

We have simulated different types of errors to find out the limitations of statistical test and then to plan for possible solutions. For example, a child node is capable of detecting consistently smaller (or greater) X_i^t than X^t by using *Sign Test*. But it will not be sufficient to detect smarter one-sided bias, which will occur when the aggregator will intelligently introduce negative and positive errors with equal frequency but significantly different magnitude. In that case, *Wilcoxon Signed Rank Test* will be useful.

We explain the operation technique of sign test and signed rank test below:

Sign Test:.

The sign test makes use of the positive and negative differences between observations [13]. Let, (X_{1j}, X_{2j}), $j = 1, 2, ...n$ be a collection of paired observations from two continuous population and the paired differences are

$$D_j = X_{1j} - X_{2j}, j = 1, 2, ...n.$$

Our aim is to determine whether the two data sets have the same median $\widetilde{\mu_1} = \widetilde{\mu_2}$ (or mean). It can also be accomplished by testing of the median of difference, $\widetilde{\mu_D}$ is 0. We apply sign test to our sensor network environment as follows:

X_{1j}:	The node's readings
X_{2j}:	Aggregated value from parent.
n_+:	Number of positive differences.
n_-:	Number of negative differences.
$n = n_+ + n_-$	Total number of readings
$k = min(n_+, n_-)$	

The probability that the two data sets are produced from the same distribution is calculated by the equation

$$2\sum_{i=0}^{k} \frac{n!}{i!(n-i)!}\frac{1}{2^n}.$$

Wilcoxon Signed-Rank Test:.

Even if the sign test is applied, an adversary can still undermine the system by providing equal number of positive and negative errors which differ in magnitude. Sign test does not take into account the amplitude of the differences [13]. For instance, in the scenario of Table 1, the number of positive and negative differences are equal, but the magnitudes of these difference values are quite different. Even though the results are faulty, sign test cannot identify this aggregator as malicious. Frank Wilcoxon devised a test procedure that uses both direction (sign) and the magnitude of errors [13]. The algorithm works as follows:

Table 1: An example situation where sign test does not work.

i	X_1	X_2	Difference
1	30	36	-6
2	31	30	+1
3	28	26	+2
4	33	38	-5

Table 2: Wilcoxon signed-ranked test.

i	X_1	X_2	Difference	Rank
2	31	30	+1	+1
3	28	26	+2	+2
4	33	38	-5	-3
1	30	36	-6	-4

i. Sort the differences according to their absolute values.

ii. Assign their ranks correspondingly with their position in the list.

iii. Add the sign of corresponding difference value to each rank.

Table 2 displays how to assign the ranks. In the table, the absolute sum of positive ranks is $w_+ = 1 + 2 = 3$ and the absolute sum of negative ranks is $w_- = |-3| + |-4| = 7$. Using the values of w_- and w_+, we calculate

$$Z = \frac{W - 0.5 - N(N+1)/4}{\sqrt{N(N+1)(2N+1)/24}}.$$

where, $W = max(w_-, w_+)$ and $N =$*number of pairs*. From Z we obtain *p-values* using *Standard Normal Approximation*. Finally, based on the chosen *significance level*, we mark the data reporting event as *accurate* if the calculated *p-value* $>$ *significance level* or *erroneous* if the calculated *p-value* $<$ *significance level*.

Following statistical theories of hypothesis testing, if the test *p-value* $<$ *significance level*, we can refute the *Null Hypotheses* (H_0) with the error in this decision influenced by the *significance level* chosen. The *Null Hypotheses* (H_0) posits that the data distributions under consideration have the same mean.

5. REPUTATION UPDATE SCHEME

As already mentioned, we have used two different types of interpretations from the probability value obtained from the statistical tests: (a) **Real Interpretation** or (b) **Boolean Interpretation**. We use the results of these statisical tests as a measure of the correctness of data reported at a particular time period. Because of the underlying noisy environments or variabilities in the sensor node agents, it is not advisable to use these point probabilities at any one time period to identify a node agent to be malicious or safe. Sequences of such tests for each node must be carefully integrated to arrive at a conclusion about its trustworthiness. To develop and maintain the reputation of a node agent over time, these test results are fed to a reputation management scheme. We have used either a $\beta - reputation$ [10] scheme

or Q-learning [20], a reinforcement learning scheme, or a combination of the two.

The $\beta-reputation$ scheme calculates the reputation value of a node agent based on the number and order of occurrence of *correct* and *erroneous* data reported by this node in the past. Let x and $\bar{x}$ be the *correct* and *erroneous* data reporting events respectively obtained from the statistical test used, and r_t and s_t be the number of occurrence of x and $\bar{x}$ respectively. Then the reputation at any instant t, P_t, is calculated by the expected value of the underlying β distribution as:

$$P_t = \frac{r_t - s_t}{r_t + s_t + 2}.$$

For *Recency* β scheme, which gives more importance to more recent experiences, we use the same equation for calculating P_t. However, in that case r_t and s_t are calculated as follows:

$$r_t = \sum_{i=1}^{t} x_i \cdot \gamma^{(t-i)}$$

$$s_t = \sum_{i=1}^{t} \bar{x}_i \cdot \gamma^{(t-i)}$$

where, $0 \leq \gamma \leq 1$ is the *discount factor*. We experimentally determined a suitable value for γ to be 0.7.

A reinforcement learning scheme, like Q-learning, can be used to update the estimation of any time-varying performance metric. We use the calculated reputation value R_t from a statistical test at an instant t to calculate actual reputation value using the *Q-learning* mechanism. The reputation scheme keeps an exponential weighted moving average, r_i^t, of the probability of the parent of node agent i being trustworthy after the t^{th} data reporting epoch. This value is incrementally updated by the following reinforcement learning scheme [20]:

$$r_t^i = \alpha \cdot r_{t-1}^i + (1 - \alpha) \cdot R_t^i$$

where for time period t, node agent i calculates the likelihood of data reported by its parent to be accurate as R_t^i using one of the statistical tests. $\alpha \in [0, 1)$ is the learning rate. We have used $\alpha = 0.1$.

A parent node is marked as malicious if its reputation value falls below a threshold in the estimation of a predetermined number of its children.

6. EXPERIMENTAL RESULTS

To evaluate the relative effectiveness of Q-learning and β-reputation in efficiently identifying malicious parent node agents, we performed experiments with different combination of Q-learning and β-reputation modules. For all the experiments, we used a publicly available configuration of sensors deployed in a lab environment in Intel Berkeley Research Lab[1]. In this environment, there are 54 nodes located strategically in a room to sense various environmental factors. We have used only the configuration of the node agents not the data collected by the environment. In that network we have chosen 8 clusters of sensor nodes and 1 aggregator node per cluster. For all of these experiments, we have chosen 5 out of 8 aggregators to be malicious. To obtain stable values with the β-reputation scheme, we calculate the reputation values based on the last 100 data reporting iterations, *i.e.*, $n = 100$.

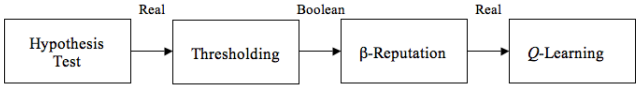

Figure 2: β & Q

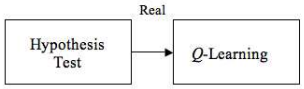

Figure 3: Only Q

We have drawn samples from a Normal Distribution of the form $N(0, 0.01)$ to form the data set for the sensors. We assume that only the leaf level node agents sense data from the environment. All the node agents located further above in the hierarchy only forwards the data reported to it after aggregation. We assume that at every data reporting event each leaf node agent gathers data and forwards that to its parents. Malicious aggregator node agents start injecting errors in the aggregated values after the first 100 data reporting events. We create different Error models to show the strengths and limitations of the tests used. We present our results in the following sequence of scenarios:

1. In the first scenario, the sign test is expected to identify malicious node agents,
2. In the next scenario, the sign test should fail but the sign rank test is expected to be successful in detecting malicious behavior.
3. Finally we present a scenario where the node agents using only sign rank test as their detection mechanism can be fooled but can successfully detect such behavior if they use the Sign Test.

Along with these three types of variation of *Error Model*, we also observe the effect of using different schemes consisting of different reputation maintenance modules. The performance metric we used for these tests is the iteration at which all the malicious node agents are successfully detected.

6.1 Error Detection Schemes:

The *Error Detection Scheme* we employ uses some combination of *Thresholding* and *β-Reputation* or *Q-learning* modules after obtaining the probability value from the statistical tests. The role of these modules are described below:

Thresholding: Upon receiving the *p-value* from the *Statistical Tests*, we use the *Significance Level* to determine the probability with which the statistical test is refuting the *Null Hypothesis* (H_0). For our experiments, refutation of *Null Hypothesis* (H_0) indicates the fact that the data sets fed to the statistical test are drawn form distributions having different mean value. We refer to this process as *Thresholding* by *Significance Level*.

β-Reputation: This module is used to calculate the reputation given either a sequence of probabilities or boolean

values. We have used both *Simple* β and *Recency weighted-*β reputation mechanisms to devlop different security schemes.

Q-learning: We use *Q-learning* to maintain the reputation values of parents by each of its children. It takes into account the history of significance test results from previous data reporting intervals. We can increase or decrease the effect of history on the computed reputation by adjusting the value of the learning rate, α.

We form different reputation management schemes utilizing various combinations of the above mentioned modules to devlop the following schemes:

6.1.1 Scheme 1 (β & Q):

In this scheme, at first we perform sign test or sign rank test to obtain the *p-value*. Then we use *Thresholding* to get a boolean value. This boolean value is used as input to the β-Reputation module. This module calculates the reputation value based on the last 100 statistical test results. The probability values calculated by this module is fed to the *Q-learning* mechanism to determine the final reputation value. Thus considers previous history of the parent nodes. *Q-learning* together with β-Reputation adds some inertia to the belief of leaf level nodes about their parents. Due to the case of multiple modules this scheme is computationally more resource intensive than the other schemes elaborated below.

6.1.2 Scheme 2 (Only Q):

In this scheme we directly use the *p-value* obtained from the statistical tests in the *Q-learning* equation. This scheme is computationally cheap, but depending on the value of α, may be either too sensitive or insensitive to past history. Being over-sensitive to recent history can lead to detection of non-malicious nodes as malicious. On the other extreme it might take a long time to detect malicious behaviors. Hence, it is important to judiciously choose the learning rate. This problem, however, is a characteristic of any recency-weighted scheme.

6.1.3 Scheme 3 (Recency β):

In this scheme we perform *Thresholding* on the *p-value* obtained from the statistical tests and then input that value to the *Recency weighted-*β reputation scheme (this is similar to the basic β-reputation scheme but puts exponentially higher weights on more recent events). This module gives more inertia to the detection mechanism than *Scheme 2* and is computationally more efficient than *Scheme 1*.

6.1.4 Scheme 4 (Q with Thresholding):

In this scheme we use *Q-learning* mechanism coupled with the *Thresholding* technique. The boolean value obtained from the thresholding scheme is fed to the *Q-learning* equation as the current reputation (R_t^i).

We have used the four *schemes* mentioned above in different scenarios and compared their results.

6.2 Detecting more frequent one-sided errors (Error model 1):

In this experiment, we incorporate errors sampled from another Normal Distribution with mean being k times the Standard Deviation (σ) of the Sensing data set. We varied

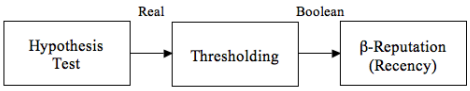

Figure 4: Recency β

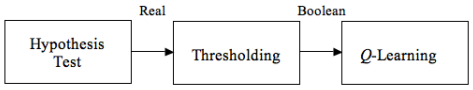

Figure 5: *Q* with *Thresholding*

k from 0.5 to 1.5 in steps of 0.25. Error generated from this distribution is directly added to the aggregated value of the malicious aggregators. As expected, both Sign Test and Sign Rank Test are almost equally capable of detecting such errors as shown in Figures 6 and 7 respectively.

It is clear from the results that different schemes have different efficiencies for detecting malicious node agents. This can be explained by the internal mechanism that is involved in the computation of reputation for each scheme. For Scheme 1, detection is considerably delayed compared to other schemes. This occurs because the basic β-reputation scheme gives significantly more importance to the history of the parent node than any other scheme. The Q-learning Scheme (Scheme 2) appears to be particularly fast in identifying malicious parents. We, however, observed that for larger error rates (error rates greater than 0.01) it also produces false positives, misidentifying normal parents as erroneous ones. Schemes 3 and 4, though not as fast as Scheme 2 in detecting malicious parents, are more robust than Scheme 2 and does not raise any false alarms. Hence, our recommendation is to use Scheme 2 for lower error rates and either Scheme 3 or 4 for larger error rates.

6.3 Detecting larger errors on one side (Error model 2:)

In this error model, we incorporate positive and negative errors in the system with equal frequency with error magnitudes on one side being larger than that on the other side. We again sample errors from a similar Normal Distribution as in the previous error model. But this time we square the errors on one side to generate different magnitudes for positive and negative errors though both types of errors occur with equal frequency. In this case we expected the malicious node agents to remain undetected if the child node uses Sign Test, whereas they should be correctly identified by using Sign Rank Test. Our results support this hypothese. The relative success of the various reputation management schemes using the Sign Rank test is shown in Figure 8. The trends are consistent with those discussed in Section 6.2.

6.4 Detecting balanced errors of different frequency (Error Model 3):

We design this final error model to emphasize the importance of using both of the statistical tests. We sample errors from an uniform distribution. If the generated value is greater than $0 < \delta < 1$ we add a positive error but after

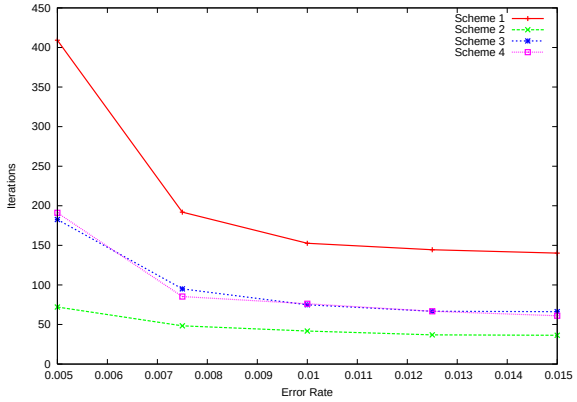

Figure 6: Sign Test results with Error model 1.

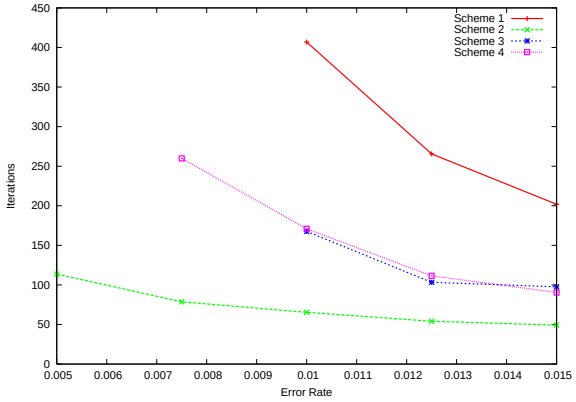

Figure 8: Sign Rank Test results for Error Model 2.

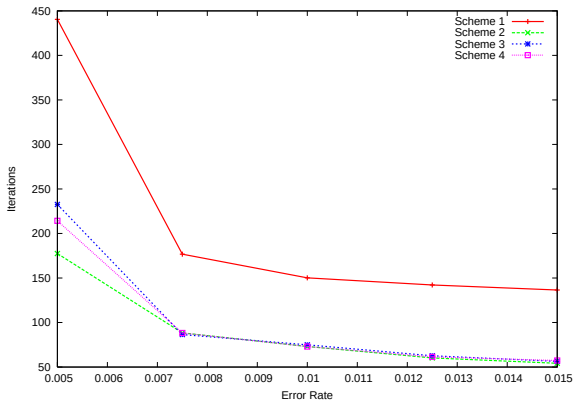

Figure 7: Sign Rank Test results with Error model 1.

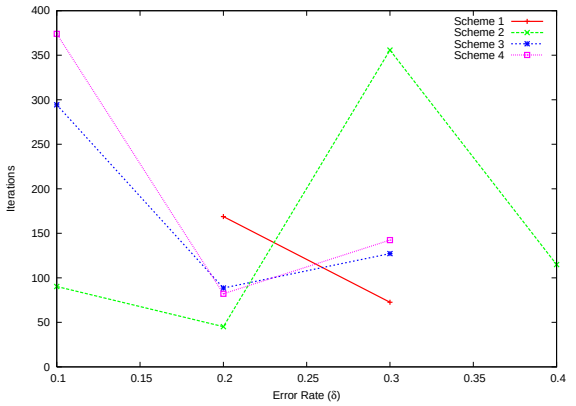

Figure 9: Sign Test results with Error Model 3.

scaling it down appropriately. Otherwise, the negative error of that value is added without any scaling. The scaling down is performed such that the total value of the positive errors is the same as the total value of the negative errors. This way we make sure that the malicious aggregators are introducing errors with different frequencies but the cumulative error of one type is almost equal to that of the other type.

In this case, mechanisms using the Sign Rank test cannot identify the malicious node agents. The mechanisms could detect the malicious node agents using the Sign Test for some erorr rates but their performance appears to be more erratic than for the other error models. For example, only Scheme 2 can detect all malicious nodes for $\delta = 0.4$. This case is difficult to detect with Sign Test as positive and negative errors are almost equally frequent.

7. CONCLUSION

In existing security techniques the aggregator node agents monitor their children. In this paper, we propose a novel mechanism where node agents reporting to an aggregator node can detect whether the latter is malicious. Our mechanism combines techniques from statistics and artificial intelligence for robust error detection when there is a bias in the reported errors. We experiment with various types of malicious behavior to demonstrate the robustness of our combined approach. An important observation is that very high learning rates can increase detection but also introduce unacceptable false positives.

In the future, we plan to examine additional combinations of statistical and AI approaches for use in malicious node detection in sensor networks. One drawback of the schemes presented here is that they cannot detect unbiased errors introduced by the aggregator node. We will implement reputation calculation schemes for more sophisticated malicious node agents by incorporating techniques like the Chernoff bounds and Chebyshev inequality. We also plan to test the system with more complex attacks. For instance, we want to visualize the reaction of the system to dynamic behavior changes of the compromised node agents.

Acknowledgment: This work has been supported in part by a DOD-ARO Grant #W911NF-05-1-0285.

8. REFERENCES

[1] *Intel Research Lab, Berkeley.*

[2] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy*, page 197, Washington, DC, USA, 2003. IEEE Computer Society.

[3] J.-Y. Chen, G. Pandurangan, and D. Xu. Robust computation of aggregates in wireless sensor networks: distributed randomized algorithms and analysis. In

IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks, page 46, Piscataway, NJ, USA, 2005. IEEE Press.

[4] A. Deshpande, S. Nath, P. B. Gibbons, and S. Seshan. Cache-and-query for wide area sensor databases. In *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 503–514, New York, NY, USA, 2003. ACM Press.

[5] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*, pages 41–47, New York, NY, USA, 2002. ACM Press.

[6] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar. Next century challenges: scalable coordination in sensor networks. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 263–270, New York, NY, USA, 1999. ACM Press.

[7] T. He, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. Aida: Adaptive application-independent data aggregation in wireless sensor networks. *Trans. on Embedded Computing Sys.*, 3(2):426–457, 2004.

[8] L. Hu and D. Evans. Secure aggregation for wireless networks. In *SAINT-W '03: Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT'03 Workshops)*, page 384, Washington, DC, USA, 2003. IEEE Computer Society.

[9] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. Impact of network density on data aggregation in wireless sensor networks. In *ICDCS '02: ProceDeDeshpande03:CacheQueryshpande03:CacheQueryedings of the 22 nd International Conference on Distributed Computing Systems (ICDCS'02)*, page 457, Washington, DC, USA, 2002. IEEE Computer Society.

[10] A. Josang and R. Ismail. The beta reputation system. In *15th Bled Electronic Commerce Conference*, 2002.

[11] J. Lopez. Unleashing public-key cryptography in wireless sensor networks. *J. Comput. Secur.*, 14(5):469–482, 2006.

[12] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tag: a tiny aggregation service for ad-hoc sensor networks. *SIGOPS Oper. Syst. Rev.*, 36(SI):131–146, 2002.

[13] D. C. Montgomery and G. C. Runger. *Applied Statistics and Probability for Engineers*. WILEY, 2006.

[14] P. Mukherjee and S. Sen. Detecting malicious sensor nodes from learned data patterns. In *ATSN '07: Proceedings of the Workshop on Agent Technology for Sensor Networks, AAMAS*, pages 11–17, 2007.

[15] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler. Spins: security protocols for sensor networks. *Wirel. Netw.*, 8(5):521–534, 2002.

[16] B. Przydatek, D. Song, and A. Perrig. Sia: secure information aggregation in sensor networks. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 255–265, New York, NY, USA, 2003. ACM Press.

[17] R. M. Ruairi and M. T. Keane. An energy-efficient, multi-agent sensor network for detecting diffuse events. In *IJCAI*, pages 1390–1395, 2007.

[18] A. L. Servin and D. Kudenko. Multi-agent reinforcement learning for intrusion detection. In *Adaptive Learning Agents and Multi Agent Systems 2007*, pages 158–170, 2007.

[19] M. Sharifzadeh and C. Shahabi. Supporting spatial aggregation in sensor network databases. In *GIS '04: Proceedings of the 12th annual ACM international workshop on Geographic information systems*, pages 166–175, New York, NY, USA, 2004. ACM Press.

[20] C. J. C. H. Watkins and P. D. Dayan. Q-learning. *Machine Learning*, 3:279 – 292, 1992.

[21] R. Watro, D. Kong, S. fen Cuti, C. Gardiner, C. Lynn, and P. Kruus. Tinypk: securing sensor networks with public key technology. In *SASN '04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, pages 59–64, New York, NY, USA, 2004. ACM.

[22] J. Wu, C. jun Wang, J. Wang, and S. fu Chen. Dynamic hierarchical distributed intrusion detection system based on multi-agent system. In *WI-IATW '06: Proceedings of the 2006 IEEE/WIC/ACM international conference on Web Intelligence and Intelligent Agent Technology*, pages 89–93, Washington, DC, USA, 2006. IEEE Computer Society.

[23] Y. Yang, X. Wang, S. Zhu, and G. Cao. Sdap: a secure hop-by-hop data aggregation protocol for sensor networks. In *MobiHoc '06: Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, pages 356–367, New York, NY, USA, 2006. ACM Press.