Using Learned Data Patterns to Detect Malicious Nodes in Sensor Networks

Partha Mukherjee and Sandip Sen

Department of Mathematical and Computer Science
University of Tulsa
{partha-mukherjee,sandip}@utulsa.edu

Abstract. As sensor network applications often involve remote, distributed monitoring of inaccessible and hostile locations, they are vulnerable to both physical and electronic security breaches. The sensor nodes, once compromised, can send erroneous data to the base station, thereby possibly compromising network effectiveness. We consider sensor nodes organized in a hierarchy where the non-leaf nodes serve as the aggregators of the data values sensed at the leaf level and the Base Station corresponds to the root node of the hierarchy. To detect compromised nodes, we use neural network based learning techniques where the nets are used to predict the sensed data at any node given the data reported by its neighbors in the hierarchy. The differences between the predicted and the reported values is used to update the reputation of any given node. We compare a Q-learning schemes with the Beta reputation management approach for their responsiveness to compromised nodes. We evaluate the robustness of our detection schemes by varying the members of compromised nodes, patterns in sensed data, etc.

1 Introduction

Sensors in wireless sensor networks are used to cooperatively monitor physical and environmental conditions specially in regions where human access is limited. Current research on sensor networks propose data aggregation protocol where the sensor nodes reside at the leaf level and the non-leaf nodes act as the aggregator nodes. If a large number of nodes become damaged or compromised, the entire data gathering process may be jeopardized. Hence, the detection of faulty nodes and protecting the security and integrity of the data is a key research challenge.

In our work, the sensor nodes are assumed to be deployed in a terrain where the data being sensed follows a time varying pattern over the entire sensed area. In such scenarios standard outlier detection mechanisms will fail as the data values sensed may vary widely over the sensor field. We propose a neural net learning based technique where regional patterns in the sensor field can be learned offline from sufficient number of observations and thereafter used online to predict and monitor data reported by a node from data reported by neighboring nodes.

We assume that the nodes and the network will function without error for an initial period of time after deployment (provides data for offline training of

S. Rao et al. (Eds.): ICDCN 2008, LNCS 4904, pp. 339–344, 2008.

the net). Next the trained neural nets are used online to predict the output of the nodes given the reported values of the neighboring sensors. The difference between the predicted and reported values is used to measure error. Such errors are used by a couple of incremental reputation update mechanisms, Q-learning and Beta reputation scheme,which take sequence of errors to decide if a node is compromised or not. If the updated reputation falls below a specified threshold, the node is reported to be faulty. We have successfully used these reputation schemes to quickly detect erroneous nodes for different network sizes and data patterns over the sensor field without any false positives and false negatives.

2 Experimental Framework

We assume a sensor network with n nodes, where the nodes are distributed over a region with (x_i, y_i) representing the physical location of the sensing node i[1]. The n nodes are arranged in a tree hierarchy with the base station as the root node. Each non-leaf node in the L-level[2] hierarchy aggregates data reported to it by its k children and forwards it to its own parent in turn.

We model fluctuations of the sensed data in the environment by adding noise to the function value $f(x_i, y_i, t)$ for the $i-$th node at time interval t. So, the sensed value at position (x, y) at time t is given by $f(x, y, t) = g(x, y) + h(t) + N(0, \sigma)$, where h maps a time to the range $[l, h]$ and $N(0, \sigma)$ represents a 0 mean, σ standard deviation Gaussian noise. We have used two different g functions, $e^{-(x^2+y^2)}$ and $\frac{(x+y)}{2}$ and refer to these two environments as E1 and E2 respectively. In our experiments, we assume that each sensor node adds a randomly generated offset in the range $[0, \epsilon]$ to the data value it senses and vary the number of compromised nodes only at the leaf level,though our mechanism, is capable of detecting faulty nodes at any position in the hierarchy except the root node, assumed as base station. The initial error-free data reporting interval is assumed to be D and the threshold for malicious node detection is taken as a fraction $p = 0.03$ of the maximum reputation a sibling possesses at a particular iteration. E stands for the entire data set including offline and online data.

2.1 Learning Technique

To form the predictor for a given node i in the sensor network, we use a three-level feed-forward neural network with $k - 1$ nodes in the input layer which receives data reported by the siblings of this node. Each such neural network has one hidden layer with H nodes and the output layer has one node that corresponds to the predicted value for this sensor node. A back-propagation training algorithm, of learning rate η and momentum term γ is used with sigmoid activation function

[1] The index of node i is calculated as $index_i = c * k + i$, c is the index of the parent node and k is the number of its children. c $0 \leq c \leq \lfloor \frac{n}{k} \rfloor$.

[2] There are k^{l-1} nodes in level l and $n = \sum_{l=1}^{L} k^{l-1}$ where k^{L-1} leaf-level nodes are sensing nodes.

$f(y) = \frac{1}{1+e^{-y}}$ for the neural network units. The outputs are restricted to the $[0, 1]$ range.

We experiment with two representative sensor networks, each organized in an m-ary tree:

Network 1 (N1): The smaller network with $m = 3$ has a total of $n = 40$ nodes, of which 27 leaf level nodes sense data from the environment. The neural networks used for learning node predictors have the following parameters: $\eta = 1.0, \gamma = 0.7$, $H = 6$, $D = 4500$, and $E = 5000$. The output of each node is predicted by taking inputs from two of its siblings. The prediction efficiencies of the net for the functions $e^{-(x^2+y^2)}$ and $\frac{(x+y)}{2}$ are 94.45% and 92.6% respectively. We ran experiments with 3, 5 and 7 malicious nodes for this network.

Network 2 (N2): The larger network with $m = 4$ has a total of $n = 85$ nodes, of which 64 leaf level nodes sense data from the environment. The neural networks used for learning node predictors have the following parameters: $\eta = 0.8$, $\gamma = 0.7, H = 8, D = 4500$, and $E = 5000$. The output of each node is predicted by taking inputs from three of its siblings. In this case the prediction efficiencies of the net for the functions $e^{-(x^2+y^2)}$ and $\frac{(x+y)}{2}$ are 93.30% and 90.6% respectively. We ran experiments with 5, 10, and 15 malicious nodes for this network. Algorithm 1 is used online to update the reputation for each node i at each data reporting time interval based on relative error $\varepsilon_i^t = \left|1 - \frac{reported_i^t}{predicted_i^t}\right|$, where $predicted_i^t$ and $reported_i^t$ are the values predicted for and the actual output by the node i at time t respectively. From this relative error, an error statistic $\aleph_i^t = e^{-K*\varepsilon_i^t}$[3] is computed for updating node reputation. Reputation updates are performed by the Q-learning and Beta-reputation schemes. As performance metric, we use the iterations taken by these mechanisms to detect the first and last erroneous nodes. The latter value corresponds to the time taken to detect all faulty nodes.

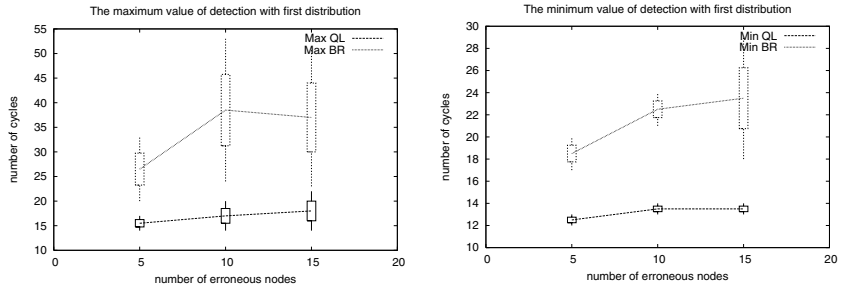
Q-Learning Framework: The reputation of every node i is updated as follows: $Reputation_{QL_i}^t \leftarrow (1-\alpha) * Reputation_{QL_i}^{t-1} + \alpha * \aleph_i^t$. We use a learning rate, α, of 0.2 and an initial reputation, $Reputation_{QL_i}^0 = 1, \forall i$.

RFSN Framework: In Reputation Based Framework for Sensor Networks (RFSN) [1] framework the corresponding reputation update equation is given by $Reputation_{\beta_i^t} = \frac{\gamma_i^t+1}{\gamma_i^t+\beta_i^t+2}$, where γ_i^t and β_i^t are the cumulative cooperative and non-cooperative responses received from node i until time t. We assume $\gamma_i^0 = \beta_i^0 = 0$ and these values are subsequently updated as $\gamma_i^t \leftarrow \gamma_i^{t-1} + \aleph_i^t$ and $\beta_i^t \leftarrow \beta_i^{t-1} + (1 - \aleph_i^t)$.

[3] We have used $K = 10$. The results are robust to K values of this order but too high or too low K value would respectively be inflexible or will not sufficiently penalize errors.

Algorithm 1: DetectMalicious(n, N)

Data: The trained neural net N with set of given parameters, number of nodes n
Result: Detection of malicious nodes
initialization: $Reputation\_Threshold = 0.03$, $\forall i, Reputation^0_{QL_i} = 1$, $Reputation^0_{\beta_i} = 0$;
for *t=0;;t++* **do**
  for *each sensor node* $node_i$ **do**
    Compute relative_error: ε;
    Compute error_statistic: $f(\varepsilon)$;
    Update $Reputation^t_{QL_i}$;
    Update $Reputation^t_{\beta_i}$;
    if $Reputation^t_{QL_i} \leq Reputation\_Threshold * \max_{k \in Neigh_i} Reptutation^t_{QL_k}$ **then**
      $node_i$ is malicious according to Q-learning based reputation mechanism;
    end
    if $Reputation_{\beta_i} \leq Reputation\_Threshold * \max_{k \in Neigh_i} Reptutation^t_{\beta_k}$ **then**
      $node_i$ is malicious according Beta-Reputation mechanism;
    end
  end
end

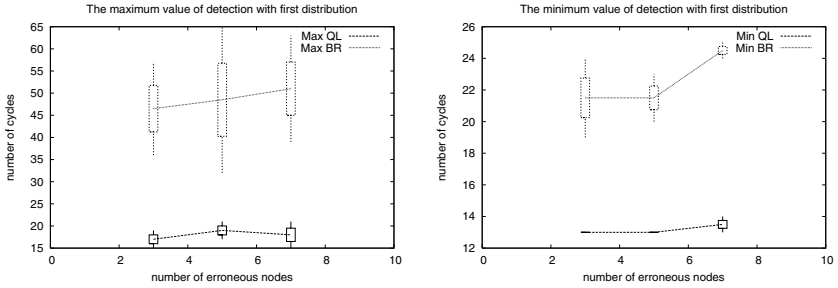


(a) Maximum time to detect malicious nodes (b) Minimum time to detect malicious nodes

Fig. 1. Maximum and minimum number of cycles required to detect compromised nodes by the Q-learning (QL) and RFSN (BR) approaches in environment E1 ($n = 85$) for distribution $e^{-(x^2+y^2)}$

2.2 Observations

We ran experiments with 10 random orderings of data reporting sequences and average our results over these runs. Figures 1, 2 show the average time taken to detect the first and last malicious node taken by these reputation schemes for the two problem sizes. We omit the figures for distribution $\frac{x+y}{2}$ for both the networks due to space constraints. The result for $\frac{x+y}{2}$ corroborates the view



(a) Maximum time to detect malicious nodes

(b) Minimum time to detect malicious nodes

Fig. 2. Maximum and minimum number of cycles required to detect compromised nodes by the Q-learning (QL) and RFSN (BR) approaches in environment E2 ($n = 40$) for distribution $e^{-(x^2+y^2)}$

observed for $e^{-(x^2+y^2)}$. The standard deviations of these metrics, centered around the mean, are also shown. We highlight the following observations:

Observation 1: For both environments and problem sizes, the time taken to detect the first malicious node is less for the Q_Learning based approach than that of the RFSN based approach. This value remains between 12 to 15 iterations irrespective of the environment and number of erroneous nodes in the network. We have experimented with at most 15 and 7 faulty nodes respectively for the 85 and 40 node networks.

Observation 2: The plots show that the mean values of the time taken to detect the last erroneous node is again significantly less for the Q_Learning based reputation scheme compared to the RFSN based approach irrespective of network size and the number of malicious nodes (see Figures 1(a),and 2(a)).

Observation 3: We did not observe any false positives (normal nodes identified as malicious) or false negatives (undetected malicious nodes) in our experiments.

We conclude that for the class of environments that we have considered, the Q-Learning scheme detects malicious nodes in the network more expediently compared to the Beta-reputation approach. The significance of the past reputation values are exponentially discounted in the Q-Learning scheme whereas the Beta-reputation scheme gives equal weight to early and recent experiences.

3 Related Work

Recent work on securing sensor networks use techniques like key establishment, authentication, secure routing, etc. Symmetric key cryptography is preferred for protecting confidentiality, integrity and availability [2,3]. Intrusion Tolerant secure routing protocols in wireless Sensor Networks (INSENS) [4] tries to bypass malicious nodes and nullifies the effect of compromised nodes in the vicinity of

malicious nodes. Existing literature on intrusion detection mechanisms in sensor networks use statistical approaches like outlier detection schemes, where data is assumed to be sampled from the same distribution [5]. Such mechanisms, however, cannot be used when sensor fields span a wide area and can have significant variation in the sensed data.Our proposed neural net based approach learns such patterns from reported data and can be combined with reputation management schemes to detect malicious nodes online.

4 Conclusion

For the environments, where standard outlier detection mechanisms are ineffective, we propose a combination of a neural network based offline learning approach and online repuation update schemes to identify nodes reporting inconsistent data. We experimentally evaluate our scheme for two different network sizes and two different data patterns over the sensor field. Results show that our approach is successful in identifying multiple colluding malicious nodes without any false positives and false negatives. The approach scales well and is robust against attacks even when as much as 25% of the sensor nodes are corrupted. The Q-learning based approach is found to detect malicious nodes faster than the Beta-repuation based scheme. In the future we plan to extend our work to incorporate the analysis of more sophisticated collusion, and prevent malicious nodes from using false identities to report spurious, multiple false data.

Acknowledgement. Research supported in part by a DOD-ARO Grant #W911NF-05-1-0285.

References

1. Ganeriwal, S., Srivastava, M.B.: Reputation-based framework for high integrity sensor networks. In: SASN 2004. Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks, pp. 66–77. ACM Press, New York (2004)
2. Eschenauer, L., Gligor, V.D.: A key-management scheme for distributed sensor networks. In: Proceedings of the 9th ACM conference on Computer and communications security, pp. 41–47 (November 2002)
3. Perrig, A., Szewczyk, R., Tygar, J.D., Wen, V., Culler, D.E.: Spins: security protocols for sensor networks. Wirel. Netw. 8(5), 521–534 (2002)
4. Deng, J., Han, R., Mishra, S.: Insens: Intrusion-tolerant routing in wireless sensor networks (2002)
5. Yang, Y., Wang, X., Zhu, S., Cao, G.: Sdap: A secure hop-by-hop data aggregation protocol for sensor networks. In: MobiHoc 2006. Proceedings of the 7th international symposium on Mobile ad hoc networking and computing, pp. 356–367. ACM Press, New York (2006)