Improving Robustness of a Popular Probabilistic Clustering Algorithm Against Insider Attacks

Sayed M. Saghaian N. E.[1(✉)], Tom La Porta[1], Simone Silvestri[2], and Patrick McDaniel[1]

[1] School of EECS, Pennsylvania State University, State College, USA
{sms676,tlp,mcdaniel}@cse.psu.edu
[2] Computer Science Department, University of Kentucky, Lexington, USA
silvestri@cs.uky.edu

Abstract. Many clustering algorithms for mesh, ad hoc and Wireless Sensor Networks have been proposed. Probabilistic approaches are a popular class of such algorithms. However, it is essential to analyze their robustness against security compromise. We study the robustness of EEHCA, a popular energy efficient clustering algorithm as an example of probabilistic class in terms of security compromise. In this paper, we investigate attacks on EEHCA through analysis and experimental simulations. We analytically characterize two different attack models. In the first attack model, the attacker aims to gain control over the network by stealing network traffic, or by disrupting the data aggregation process (integrity attack). In the second attack model, the inducement of the attacker is to abridge the network lifetime (denial of service attack). We assume the clustering algorithm is running periodically and propose a detection solution by exploiting Bernoulli CUSUM charts.

Keywords: Probabilistic clustering algorithm · Anomaly detection · CUSUM test

1 Introduction

Clustering algorithms are widely used in wireless ad hoc and sensor networks to help improve efficacy of performing functions such as routing and data aggregation. Clustering provides scalability, efficient communication, and energy conservation, and prolongs the network lifetime [1,26].

A Wireless Sensor Network (WSN) typically consists of low cost sensor nodes which are not tamper resistant, and are typically left unattended. Consequently, they are susceptible to physical and cyber-attacks. In particular, they are vulnerable to *insider* attacks in which a compromised node retains its full credentials, and is able to operate in compliance with security rules within the network. Likewise, in many ad hoc and mesh network applications nodes are vulnerable to insider attacks. Attackers have different incentives including stealing traffic,

N. Park et al. (Eds.): SecureComm 2020, LNICST 335, pp. 381–401, 2020.
https://doi.org/10.1007/978-3-030-63086-7_21

agitating the data aggregation process, changing routing information, or diminishing the network lifetime. Specifically, the damage from an attack may be more extensive if the compromised node plays the clusterhead role. In this paper, we analyze the impact of node compromise on a probabilistic clusterhead election protocol and propose an algorithm to detect compromised nodes as soon as possible.

Each cluster has a leader called a clusterhead. As opposed to ordinary network nodes which are mainly responsible for sensing or generating data, clusterheads have more responsibilities. Ordinary members of a cluster send their gathered data to their clusterhead. A clusterhead may perform some initial processing on the gathered data, and then forward the data to the base station or possibly other clusterheads. Clusterheads are further responsible for organizing activities within the cluster, maintaining routing tables and paths to ordinary nodes as well as other clusterheads and the base station. Accordingly, the energy resources of a clusterhead are depleted more quickly than ordinary nodes.

In this paper we focus on the robustness of probabilistic clusterhead formation protocols which are very popular. We use EEHCA as an example. We consider two types of attacks. In the first attack, malicious nodes try to gain control over the network by stealing network traffic, or by disrupting the data aggregation process. This attack is known as an *integrity attack* where the attacker inserts itself into the data path and manipulates data. To illustrate the impact of this attack, we derive the percentages of legitimate ordinary nodes served by a malicious clusterhead as a function of the number of compromised nodes. We will observe that even if only a small fraction of nodes are compromised, a considerable number of legitimate ordinary nodes would follow an anomalous clusterhead.

The second attack we consider is called *battery drain attack* under which malicious nodes try to make the energy spent by legitimate ordinary nodes increase. They might further aim to increase the traffic going through legitimate clusterheads. For this attack, we compute the ratio of the expected number of nodes in each cluster under this attack model to the expected number of nodes in each cluster in an honest system. Moreover, since legitimate ordinary nodes in this attack scenario most likely have to join a cluster with a clusterhead positioned at a farther distance, we compute the ratio of the expected energy spent by legitimate ordinary nodes in a cluster under this attack over the expected energy spent by ordinary nodes in a cluster in an attack free environment.

We investigate the effectiveness of these attacks and propose a detection strategy against them which aims to detect malicious nodes as soon as possible. We exploit Bernoulli CUSUM charts to detect misbehaviors rapidly, and discuss how to design an anomaly detection algorithm with a zero false positive rate.

The main contributions of this paper are:

1. We analyze the robustness of EEHCA, a popular probabilistic energy efficient clustering algorithm for WSNs against security compromise.
2. We introduce two types of attack models and analytically characterize the impact of these attack models on the network as a function of the number

of compromised nodes; the scope of the first attack is to gain control over the network (integrity attack). For this attack, we derive the percentage of legitimate ordinary nodes served by a malicious clusterhead as a function of the number of compromised nodes. In the second attack model, the incentive is to abridge the network lifetime (denial of service attack). For this attack, we consider the ratio of the expected energy spent by legitimate ordinary nodes in a cluster under this attack model over the expected energy spent by ordinary nodes in a cluster in an attack free environment.
3. We propose a detection method by exploiting Bernoulli CUSUM charts that results in a rapid anomaly detection while preserving a zero false positive rate. In addition to CUSUM test, we also consider different statistical techniques to detect anomalous sensors including Score test and Likelihood-ratio test.

We begin in the following Section with a review of several key related work.

2 Related Work

Many algorithms for clustering in WSNs have been proposed [1,3,13,26]. One popular class of clustering algorithm takes a probabilistic approach in which nodes become clusterhead with some probability. This class of clustering enables a rapid cluster formation with a low overhead as nodes independently define their role. The objective in this class is to find the optimal probability which results in minimizing the energy spent in the network. This feature is particularly suitable for WSNs where energy conservation is vital to enhancing the network lifetime. LEACH [10], EEHCA [6], HEED [25], and the algorithm proposed by Choi and Lee [8] are examples of this class.

[24] surveyed anomaly detection in WSNs. Perrig et al. [15] proposed a prevention-based scheme which exploits cryptographic primitives such as secret key management, encryption, and authentication. [23] presented a relatively efficient access control method in a sensor network based on public-key and Elliptic Curve Cryptography. However, prevention-based approaches such as cryptographic primitives cannot address security threats due to *insider* attackers. [12] proposed an anomaly detection algorithm that captures insider attackers with a high detection rate and a low false positive rate only when as many as 25% of sensor nodes are compromised and misbehaving. However, we will show that even if only 10% of sensors are compromised, a considerable percentage of legitimate ordinary nodes will follow an anomalous clusterhead.

Our problem of detecting anomalous sensor nodes in the EEHCA clustering algorithm is an instance of a change detection problem. The change detection problem has found many applications from quality control and economics [4] to network security [22] and fraud detection [21]. CUSUM (CUmulative SUM) chart [9,14], sometimes called CUSUM test, is a quickest detection [16] algorithm. CUSUM test is more effective and more popular than other algorithms such as Shewhart control chart [11], Sets method, CUSCORE method and SHDA method [20]. In this paper, in addition to CUSUM test, we also consider different parametric statistical tests to detect anomalous nodes in EEHCA including Score

test and Likelihood-ratio test [2], and compare their performance in terms of detection rate and percentages of falsely removed legitimate nodes.

3 Background Material

In this Section, we review EEHCA, a popular probabilistic energy efficient clustering algorithm, two common operations on point processes that we will exploit in our attack model scenarios, namely *thinning* and *superposition*. We further review two parametric statistical tests and Bernoulli CUSUM test as tools and techniques to detect anomalous behavior in the clusterhead formulation process.

3.1 EEHCA

Authors in [6] proposed a distributed energy efficient clustering mechanism named EEHCA in which they assume sensor nodes are distributed as a homogeneous spatial Poisson process in a 2-D square plane. In their scheme, sensor nodes become a clusterhead voluntarily with a same fixed probability, p, independent of their location and each other. These nodes are called *volunteer clusterheads*. Volunteer clusterheads then advertise themselves to nodes within at most k hops.

Nodes that are not volunteer clusterheads themselves join the cluster with the closest volunteer clusterhead within k hops. These nodes are called *ordinary* nodes. As a result, a Voronoi tessellation is formed. A Voronoi tessellation is a partition of a plane into cells (clusters) such that each cell contains only one generating point (volunteer clusterhead) and the distance of other points (ordinary nodes) in a given cell to the corresponding generating point is smallest among the distances to other generating points.

Any other nodes that are neither volunteer clusterheads nor ordinary nodes become *forced clusterheads*. These nodes only serve themselves.

A routing infrastructure is assumed already to exist so that to send data from one node to another node, only the nodes on the routing path forward the data. Moreover, the communication environment is error-free and the nodes do not deal with retransmitting data. In their energy model the energy spent is proportional to the distance and radio range (r) directly and inversely, respectively.

The authors exploit results from independent homogeneous spatial Poisson processes [5] and derive the optimal probability of becoming clusterhead (p_{opt}) and the optimal k that lead to the minimum energy spent in the network.

Below, we review techniques related to point processes as we use them in our performance evaluation of the clustering algorithm against security compromise.

3.2 Two Common Operations on Point Processes

A Point Process [7] is a collection of points randomly scattered in some compact set W. It can be viewed either as a random set or as a random counting measure. A point process is called spatial point process when $W \subset \mathbb{R}^d$ for $d = 2$ or 3. In our setting, each point represents the location of a sensor node.

Poisson point process is the most basic and important point process. A stationary Poisson point process has two properties: The number of points of the process which fall into a bounded Borel set B has a Poisson distribution with mean of $\lambda\|B\|$ for some constant λ, where $\|B\|$ denotes Lebesgue measure of B. Furthermore, the number of points of the process in B_1 is independent from the number points of the process in B_2 for disjoint Borel sets B_1 and B_2.

In the following, we review two main operations on point processes: thinning and superposition. We exploit these operations to characterize our attack models.

3.2.1 Thinning

Thinning is an operation of removing points from a basic point process Φ_b which has intensity of λ_b by some definite rule. *p-thinning* is the simplest thinning operation in which points are removed from Φ_b with probability $(1-p_r)$ independent of location and possible removal of other points in Φ_b. The points remaining after the p-thinning operation are also a point process (*p-thinned* process) that is stationary if the basic point process is stationary. p-thinned process intensity (λ) is related to the intensity of the basic point process (λ_b) by:

$$\lambda = p_r \lambda_b \tag{1}$$

3.2.2 Superposition

Given two stationary non-overlapping point processes Φ_1 and Φ_2 with intensities of λ_1 and λ_2 respectively, define:

$$\Phi = \Phi_1 \cup \Phi_2$$

Then clearly, Φ is point process with intensity of:

$$\lambda = \lambda_1 + \lambda_2 \tag{2}$$

Next, we review some statistical techniques that can be adapted to detect possible anomalous behaviour of the compromised nodes.

3.3 Parametric Statistical Tests

We now provide some background on two main parametric statistical tests: Score and Likelihood-ratio tests. These tests require a fixed number of samples. The larger the sample size, the more accurate results are achieved. When the sample size is large enough, their statistic follow a chi-square distribution, and if the samples are derived from binomial distribution, the degree of freedom is one.

– Score Test: The score statistic for binomial proportion is:

$$S^2 = \left[\frac{\hat{p} - p_{opt}}{\sqrt{\frac{p_{opt}(1-p_{opt})}{n}}} \right]^2$$

where n is the total the number of samples.

– Likelihood-ratio Test: The Likelihood-ratio statistic for binomial proportion is:

$$LR = 2\left[y \log\left(\frac{\hat{p}}{p_{opt}}\right) + (n-y)\log\left(\frac{1-\hat{p}}{1-p_{opt}}\right)\right]$$

where $\hat{p}$ is the estimated probability of success from the observed sample, and y is the number of successes (number of times a node under test volunteered to be clusterhead) out of n trials (election rounds).

3.4 Bernoulli CUSUM Test

When the observations follow a Bernoulli distribution, the Bernoulli CUSUM test [17–19] can be exploited. Define p_0 as the in-control probability (p_{opt}) and p_1 as the out-of-control probability ($p_m = \gamma p_{opt}$). Given a sequence of independent Bernoulli observations $X_1, ..., X_n$, where $X_i = 1$ if the node under test volunteered to be clusterhead in the i^{th} round of clusterhead formation and 0 otherwise, Bernoulli CUSUM aims to detect a shift from p_0 to p_1 as soon as possible.

One advantage of CUSUM test is that it does not need to wait for a fixed number of samples (here, the status of neighbor nodes after each clusterhead formation) to perform the detection process. In contrast, as soon as a sample is available, CUSUM can check whether any changes have been occurred.

To detect an increase ($\gamma > 1$) in the Bernoulli parameter, a sensor node computes the *Increase* CUSUM statistic corresponding to each of its neighbor nodes in each round of clusterhead election:

$$C_i^+ = \max(0, C_{i-1}^+) + (X_i - k) \tag{3}$$

where k is the reference value and is defined by $k = \frac{r_1}{r_2}$ where

$$r_1 = -\log\frac{1-p_1}{1-p_0} \quad and \quad r_2 = \log\frac{p_1(1-p_0)}{p_0(1-p_1)}. \tag{4}$$

Then, it compares the computed result with a control limit (h_h). If $C_i^+ > h_h$, it signals the corresponding neighbor node is declaring to be clusterhead with some probability greater than p_{opt}.

Similarly, to detect a decrease ($\gamma < 1$) in the Bernoulli parameter, a sensor node computes the *Decrease* CUSUM statistic corresponding to each of its neighbor nodes in each round of clusterhead election:

$$C_i^- = \min(0, C_{i-1}^-) + (X_i - k) \tag{5}$$

Then, it compares the computed result with a control limit (h_l). If $C_i^- < h_l$, it signals the corresponding neighbor node is volunteering to be clusterhead with some probability smaller than p_{opt}.

To exploit Markov chains in the performance analyses of CUSUM, it is required to select k such that $k = \frac{1}{m}$, where m is an integer. In practice, one can tolerate a small change from p_1 to $p_{1,a}$ so that m is an integer ($m = \text{round}(\frac{1}{k})$).[1]

The CUSUM statistic (C_i) can be initialized in three different ways. In a *zero-start*, the CUSUM statistic initially starts from 0 ($C_0 = 0$). In the *Fast Initial Response* (FIR), a faster anomaly detection can be achieved by giving the CUSUM statistic a head-start with the cost of a small increase in the false alarm rate. $h/4$ or $h/2$ (fractions of the control limit) are usually used for initial value of the CUSUM statistic. Finally, one can assume that the CUSUM statistic has reached a steady-state or stationary distribution by the time a shift occurs. Before reaching the steady-state, if the CUSUM statistic exceeds the control limit, the generated signal is ignored and the statistic is restarted from 0.

Typically, the Average Number of Observations before Signal (ANOS) is used as a performance metric for CUSUM. $ANOS(p)$ is the expected number of observations needed to signal when the node under test volunteers to be clusterhead with probability of p. The in-control ANOS ($ANOS(p_0)$) measures the average number of observations between two successive false positives. The out-of-control ANOS ($ANOS$ $(p \neq p_0)$) indicates the speed of change detection and is defined as the average number of clusterhead election rounds until an alarm is given, indicating a neighbor node is becoming clusterhead with some probability other than the optimal probability. It is desired to have a sufficiently large in-control ANOS and as small as possible out-of-control ANOS. In the case of the steady-state, ANOS is called the Steady State ANOS, $SSANOS$. Simulation results show that convergence to the steady-state distribution occurs long before $1.5ANOS(p_0)$ [20].

A general approach in designing CUSUM is to first select a desired $ANOS(p_0)$ (average number of observations between two successive false positives) and then, find the control limit h that approximately achieves the desire $ANOS(p_0)$. There are two approaches in approximating $ANOS(p_0)$; one method is Corrected Diffusion (CD) approximation and the other approach is by using Markov chain formulation. In the CD approximation, first h^* is computed from[2]:

$$ANOS(p_0) = \frac{\exp(h^* r_2) - h^* r_2 - 1}{|r_2 p_0 - r_1|} \tag{6}$$

Once h^* is obtained, h_h for Increase CUSUM and h_l for Decrease CUSUM can be computed from:

$$h_h = \frac{\text{floor}\left[m\left(h^* - \epsilon_{p_0}\sqrt{p_0(1-p_0)}\right)\right]}{m} \tag{7}$$

[1] To find such a $p_{1,a}$, Newton-Raphson method with starting point of p_1 for solving the nonlinear equation quickly converges to a solution.

[2] Newton-Raphson method with starting point h_0^* in the range $4 \leq h_0^* \leq 8$ for increase detection scenario or starting point h_0^* in $-4 \leq h_0^* \leq -8$ for decrease detection case can be adopted to solve the nonlinear equation.

$$h_l = \frac{\text{floor}\left[m\left(h^* + \epsilon_{p_0}\sqrt{p_0(1-p_0)}\right)\right]}{m} \tag{8}$$

where ϵ_p is approximated from:

$$\epsilon_p = \begin{cases} 0.410 - 0.0842(\ln(p)) - 0.0391(\ln(p))^3 - \\ 0.00376(\ln(p))^4 - 0.000008(\ln(p))^7 & \text{if } 0.01 \leq p \leq 0.5 \\ \\ \frac{1}{3}\left(\sqrt{\frac{1-p}{p}} - \sqrt{\frac{p}{1-p}}\right) & \text{if } 0 < p < 0.01 \end{cases} \tag{9}$$

4 Attack Models

We analyze the performance of EEHCA against two attack models by exploiting Thinning and Superposition operations. In the first, the attackers aim to gain control over the network by setting their probability to become volunteer clusterheads to a value greater than p_{opt}. Consequently, they become volunteer clusterheads more often and can gain control over the network by stealing the traffic or by deleting the data sent by ordinary nodes depending on their incentives.

In the second attack model, attackers try to avoid being volunteer clusterheads. In this attack model, the malicious node sets its probability to become volunteer clusterhead to a value less than p_{opt} for the sake of increasing the energy spent by legitimate ordinary nodes as well as overwhelming legitimate clusterheads by increasing the amount of traffic passing through them.

4.1 Attack Model 1

In this subsection, we analyze an attack model where the attacker volunteers to be a clusterhead with a fixed probability (p_m) greater than p_{opt}. This attack is known as *integrity attack*. Clearly, the higher the p_m it chooses, the higher the control it may gain over the network. However, if the attacker becomes a clusterhead too frequently, it may exhaust its own battery lifetime more rapidly. Furthermore, detection of such an extreme attack is easier for the legitimate nodes. Therefore, if the incentive of the attacker is to gain the maximum control over the network during its lifetime (assuming no anomaly detection exits), it would select a p_m equal to 1. However, if the attacker wishes to gain control over the network while conserving its energy for a longer period of time, it will become a volunteer clusterhead with some probability not significantly greater than the optimal. We consider the attacker to have an arbitrary fixed p_m for this attack model in our analyses.

Assume the number of sensor nodes (N) is known and nodes are distributed according to a homogeneous spatial Poisson point process Φ_b with intensity of λ_b. In the following we analyze the percentage of legitimate ordinary nodes that belong to clusters with malicious clusterheads under attack model 1.

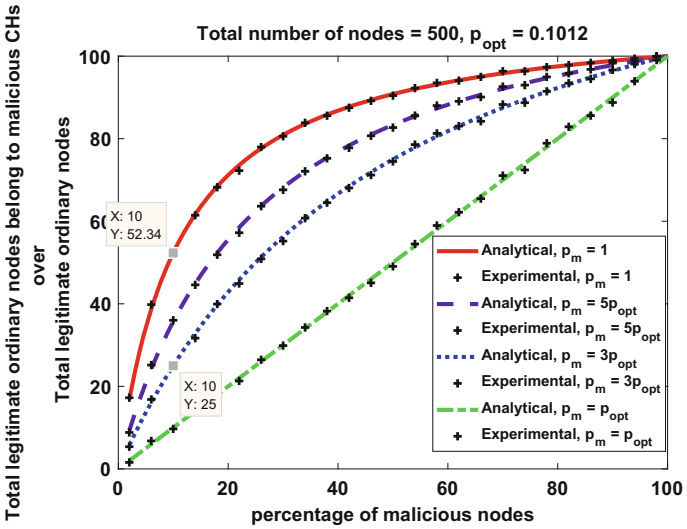

Fig. 1. Percentage of legitimate nodes served by malicious clusterheads for different values of p_m under attack model 1 for a network with 500 sensor nodes (intensity of 5) and $p_{opt} = 0.1012$.

Theorem 1. *In a sensor network with* n *nodes where nodes are distributed as a homogeneous spatial Poisson point process, in order to cause more than* 50% *of legitimate ordinary nodes to belong to clusters with malicious clusterheads, an attacker needs to be become volunteer clusterhead with probability of* p_m *satisfying:* $p_m \geq (\frac{n}{m} - 1)p_{opt}$ *where* m *is the number of compromised nodes.*

Proof. For proof, see Appendix A.

Figure 1 illustrates the effectiveness of launching an attack under attack model 1 for the sake of gaining control over the network for a network with 500 sensor nodes ($\lambda_b = 5$) where according to [6], $p_{opt} = 0.1012$. In this figure, we show the percentages for 5 different values of p_m selected by malicious nodes. For example, if only 10% of nodes are compromised, 25% of legitimate ordinary nodes will follow a malicious clusterhead if $p_m = 3p_{opt}$. Whereas if the attacker selects $p_m = 1$, the percentage of legitimate ordinary nodes served by a malicious clusterhead would be more than 50%. By setting $p_m = 1$, the attacker can gain the most control over the network with the cost of running out of power rapidly and increasing the chance of being detected by legitimate nodes. Note that the curve is linear for the case where $p_m = p_{opt}$.

We further simulate a network with 500 nodes placed in a square plane with an area of 100 (unit square) and compare the analytical versus experimental percentage of nodes served by malicious clusterheads for different values of p_m selected by the attacker for 100 trials. Our simulation results demonstrate agreement between experimental results and analytical results.

4.2 Attack Model 2

Clusterheads have more responsibilities than ordinary nodes. Consequently, their energy depletes more rapidly. In the following, we analyze the case in which the attackers aim to launch a denial of service attack or alternatively a battery drain attack by abridging the battery-lifetime of legitimate nodes. To achieve this goal, they refuse to play the role of volunteer clusterheads when they are supposed to volunteer. Gaining a "free-ride" and taking advantage of clustering without paying its cost may be another incentive for attackers. They set their probability of becoming clusterhead (p_m) to a value less than p_{opt}. As a result of this attack, legitimate ordinary nodes have to spend more energy to send their data to their clusterhead since they have to join a cluster where its clusterhead is positioned at a farther distance relative to an attack-free environment. Moreover, more traffic is sent to legitimate clusterheads in this scenario.

We compute the ratio of the expected number of nodes in each cluster under this attack model to the expected number of nodes in each cluster in an honest attack free system. Furthermore, we compute the ratio of the expected energy spent by legitimate ordinary nodes in a cluster under attack model 2 over the expected energy spent by ordinary nodes in a cluster in an attack free system.

Theorem 2, characterizes the increase in the number of nodes served by legitimate clusterheads under attack model 2:

Theorem 2. *In a sensor network with* n *nodes where nodes are distributed as a homogeneous spatial Poisson point process, if an attacker who has compromised* m *nodes* $(m \neq n)$ *never becomes a volunteer clusterhead, then:*

$$\mathbb{E}[N_{Ord,Amodel2}|N = n] \approx \frac{1}{(1 - \frac{m}{n})}\mathbb{E}[N_{Ord,Afree}|N = n] \tag{10}$$

where the random variable $N_{Ord,Afree}$ *is the number of ordinary nodes in each cluster in an attack-free system and* $N_{Ord,Amodel2}$ *denotes the number of ordinary nodes in each cluster under attack model 2.*

Proof. For proof, see Appendix B.

Theorem 3, characterizes the increase in the expected energy spent by legitimate ordinary nodes under attack model 2:

Definition 1. *Let* C_1^{Afree} *be the total energy to send 1 unit of data to the clusterhead by ordinary nodes of a cluster in an attack-free system. Similarly, let* $C_1^{Amodel2}$ *be the total energy spent by legitimate ordinary nodes to send a unit of data to the clusterhead when nodes are compromised according to attack model 2.*

Theorem 3. *In a sensor network with* n *nodes where nodes are distributed as a homogeneous spatial Poisson point process, if an attacker who has compromised* m *nodes* $(m \neq n)$ *never becomes a volunteer clusterhead, then:*

$$\frac{\mathbb{E}[C_1^{Amodel2}|N = n]}{\mathbb{E}[C_1^{Afree}|N = n]} \approx \frac{1}{(1 - \frac{m}{n})^{1/2}} \tag{11}$$

Proof. For proof, see Appendix C.

5 Anomaly Detection

In Sect. 4, we observed that a considerable percentage of legitimate nodes may be affected even if only a small fraction of nodes are compromised. Hence, we would like to detect any anomalous behavior in the sense that a node volunteers with some probability other than p_{opt}. On one hand, we want a high detection rate. On the other hand, we desire no false positives. Furthermore, the detection should be done *as soon as possible*.

This problem can be formulated as an anomaly detection problem. *Normal*, in our problem, means nodes volunteer with probability of p_{opt}. Deviation from normal means change in the probability of becoming volunteer clusterhead. Therefore, our problem reduces to detecting changes in the probability distribution. Hence, any detection mechanism for this problem should detect changes in the probability of success p_{opt} of a Bernoulli random variable.

In this Section, we present our strategy to detect malicious nodes that volunteer to be clusterheads with some probability other than p_{opt} as soon as possible. Considering that the clustering algorithm is run periodically, with the existence of even a small false positive rate, falsely removed legitimate nodes accumulate and at some point a significant number of legitimate nodes will be eliminated from the network. Removing legitimate nodes falsely results in degradation in the performance and in the optimality of the clustering algorithm.

At a high level, our detection strategy works as follows: Given in EEHCA, each node becomes a clusterhead voluntarily independent of the other nodes, each legitimate node monitors all of its neighbor nodes by recording their status (clusterhead or ordinary node) for each round of election. Using the observation sequence corresponding to the status of each neighbor node and applying some *quickest detection* algorithm, a node can detect malicious nodes in its neighborhood rapidly. By comparing different statistical tests performance for detecting anomalies, we choose CUSUM chart as our quickest detection method. Legitimate nodes will stop sending traffic or providing services to the detected nodes.

In the following, we provide the design details of CUSUM test, Score test and Likelihood-ratio test. We then compare their performances in terms of detection rate and percentages of legitimate nodes removed incorrectly.

5.1 Design Details and Simulation Results

5.1.1 Parametric Statistical Test

Since the underlying distribution is known here, and samples are expected to come from a Bernoulli distribution with probability of success of p_{opt}, we consider *parametric* statistical tests. Our *null hypothesis* (H_0) and *alternate hypothesis* (H_a) are:

H_0 : nodes volunteer with probability of p_{opt}

H_a : nodes volunteer with some probability other than p_{opt}

Based on the observed data for each neighbor node (the status of neighbor nodes after each clusterhead formation), we find the P-value. We reject the null hypothesis if P-Value is less than significance level of 5%.

We have two goals; first, a rapid anomaly detection is needed. Second, no legitimate node should be removed incorrectly. In the ideal situation where we have a large number of samples available (e.g. over 100 samples), the parametric statistical tests would have a false positive rate of 5%. However, since we cannot wait until 100 rounds of election to act on detecting malicious nodes, for the parametric statistical tests, we progressively add samples to the tests. Initially, we gather 9 samples before running the tests. From the 10^{th} round on, we perform the tests on all the available data up to that point. Lack of enough sample size would result in a higher false positive rate than the nominal 5%.

5.1.2 CUSUM Test

Setting ANOS to a higher value, results in a higher control limit h_h for Increase CUSUM (or, a lower control limit h_l for Decrease CUSUM) and hence, achieves a lower false positive rate. However, a higher ANOS results in a slower detection rate. We further observed that the zero-start CUSUM has the lowest detection rate as well as the smallest possible false alarm rate relative to the head-start approaches. Moreover, even if a node acts normal and becomes a volunteer clusterhead with probability of p_{opt}, after, on average, $ANOS(p_0)$ clusterhead formation rounds, it will be detected and marked as a malicious node assuming zero-start is used. Removing legitimate nodes falsely results in degradation in the performance and in the optimality of the clustering algorithm. Hence, it is essential to deal with the generated false alarms.

Resetting the CUSUM statistic to zero after every few rounds of cluster formation is one approach to reduce false positives. We call this approach the *resetting method*. A candidate value for the number of rounds before restarting the CUSUM statistic is determine as follows: Suppose we may tolerate up to $x\%$ of traffic to be controlled by an adversary. Considering $\frac{m}{n}\%$ of the sensor nodes are compromised, we find the p_m that results in $x\%$ of legitimate ordinary nodes belonging to a cluster with a malicious clusterhead by using Eq. (22). We set the restarting value to $ANOS(p_m) + 1$.

An obvious drawback of this approach is that it results in a slower anomaly detection. Moreover, our simulation results show that the above approach is not very effective in preventing high false alarm rates even if we increase the tolerance up to 25% which requires $p_m = 3p_0$.

A second approach to reduce the false positives, which we call it the *self-monitoring method*, is to have each legitimate node not only monitor its neighbors by recording their status (clusterhead or ordinary node) and applying the CUSUM method (Increase and Decrease), but also to have it monitor its own status. The intuition behind this method is to prevent legitimate nodes from getting marked as malicious nodes because of crossing the control limits in the CUSUM tests performed by the other nodes.

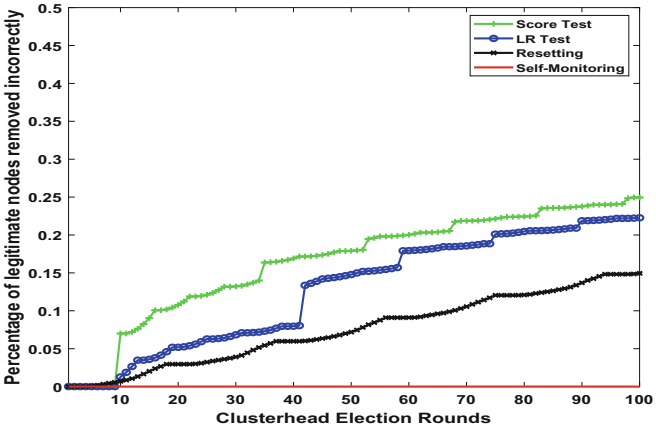

Fig. 2. Percentage of legitimate nodes removed incorrectly under resetting method assuming tolerance of up to 25% ($p_m = 3p_0$) for 100 rounds of elections.

The self-monitoring method works as follows: In each election round, a legitimate node tentatively decides to become a volunteer clusterhead with probability of p_{opt}. If the tentative result is to not become clusterhead, the node computes its own Decrease CUSUM statistic. If its Decrease CUSUM statistic is less than the Decrease CUSUM control limit (h_l), it tentatively becomes a clusterhead. If a legitimate node is tentatively a clusterhead (either as a result of the original decision or as a result of Decrease CUSUM), it computes its Increase CUSUM statistic. Finally, if the Increase CUSUM statistic is less than Increase CUSUM control limit (h_h), then it definitively becomes a volunteer clusterhead for that round of clusterhead formation.

One concern with the self-monitoring method is that a legitimate node might now become a volunteer clusterhead with some probability considerably lower than the optimal probability. However, we find this is not the case. Our simulation results show that as the number of clusterhead election rounds increase, legitimate nodes become clusterhead with some probability very close to the optimal probability. We recorded the status of legitimate nodes in the system for 500 election rounds. On average, a legitimate node volunteered to be a clusterhead with probability of $\overline{p_{leg}} = 0.0994$ (as opposed to $p_{opt} = 0.1012$) where the standard deviation is 0.0072. Hence, the normalized error of the probability of becoming a volunteer clusterhead for a legitimate node is $\frac{(p_{opt} - \overline{p_{leg}})}{p_{opt}} = 0.0783$.

Figure 2 illustrates the percentage of legitimate nodes removed incorrectly for a network with $p_0 = 0.1012$ when 10% nodes are compromised for 100 clusterhead election rounds. For Bernoulli CUSUM we set $ANOS(p_0) = 208$. Assuming tolerance of 25% ($p_m = 3p_0$), the CUSUM statistic is reset to 0 after 19 clusterhead formations. From Fig. 2, we observe that the self-monitoring method eliminates the false alarms. For the other methods, the number of nodes

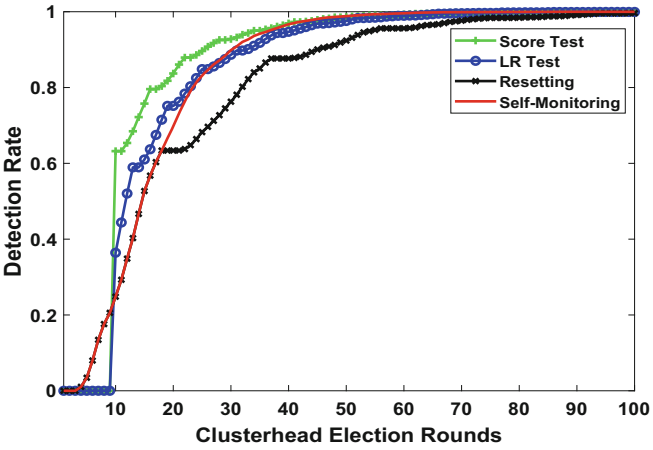

Fig. 3. Comparing detection rates. The network has 500 nodes in which 10% of them are compromised.

incorrectly characterized as compromised nodes accumulates over the rounds of clusterhead formations. Moreover, the parametric statistical tests have a higher percentage of incorrectly removed nodes than the CUSUM test. Although resetting the CUSUM statistic slows the false positive rate, false alarms still result in incorrectly removing a substantial fraction of legitimate nodes.

In Fig. 3, we compare the detection rate of the discussed anomaly detection methods. As expected, the CUSUM self-monitoring approach detects malicious nodes with a higher rate than the resetting approach. Recall, the CUSUM resetting approach, and the parametric tests result in false positives as shown in Fig. 2 while the CUSUM self-monitoring approach eliminates false positives. Although Score and LR tests have slightly higher detection rates from the round 10 to 40, their high false positive rates offset this advantage as discussed above.

Because of the high detection rate and zero false positive rate, we select the CUSUM self-monitoring approach as our anomaly detection method. To illustrate the performance of our detection method, we run an experiment for 100 trials on a network with $p_{opt} = 0.1012$ in which 10% of nodes are randomly compromised. We set the out-of-control probability $p_{1,a} = 0.1517$ for Increase CUSUM and 0.0478 for Decrease CUSUM. For Increase CUSUM, we set $ANOS_h = 208$ resulting in a control limit of $h_h = 3.1250$. For the Decrease CUSUM, we set $ANOS_l = 195$, which results in a control limit of $h_l = -2.2143$. We evaluate the performance of the proposed anomaly detection method under two different probabilities selected by an attacker, $p_m = 0.9512$ and $p_m = 3p_{opt}$. We depict the percentage of the detected malicious nodes versus the number of the clusterhead election rounds when exploiting zero-start and h/4 head-start CUSUM tests.

Figure 4 exhibits the performance of the self-monitoring approach considering two different probabilities, namely $p_m = 0.9512$ and $p_m = 3p_{opt}$, for an

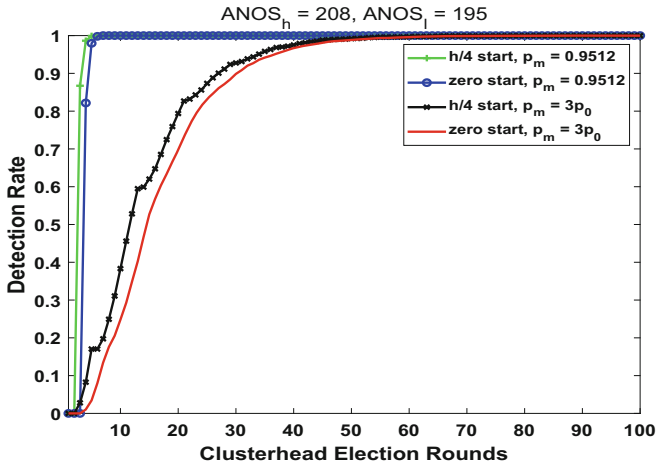

Fig. 4. Detection rate of the Self-Monitoring approach considering two different p_m's for an attacker when CUSUM charts with two different head-starts are exploited. The network has 500 nodes in which 10% of them are compromised.

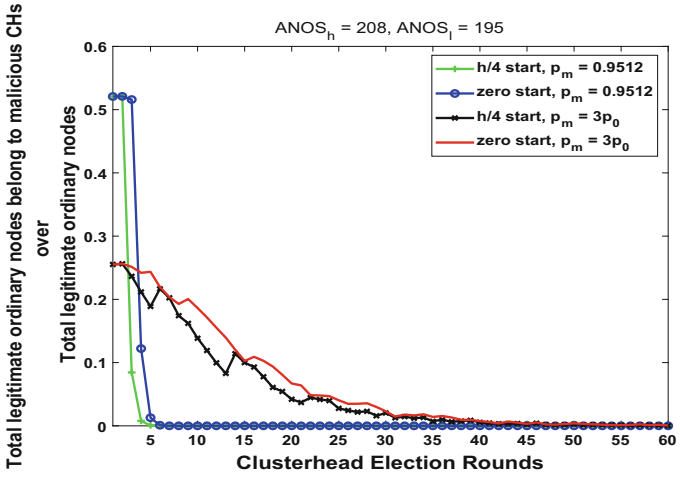

Fig. 5. Percentage of legitimate nodes served by a malicious clusterhead for different values of p_m after integrating CUSUM test into the clustering algorithm.

attacker to become a volunteer clusterhead. As Fig. 4 shows, the more vigorous the attacker, the higher the detection rate. We further compare the performance of CUSUM with different head-starts. CUSUM with a head-start of $h/4$ has a higher detection rate than CUSUM with zero-start because of the provided head-start. Using CUSUM with a head-start of $h/4$, 87% of malicious node were detected in only three rounds of clusterhead formation when the attacker becomes a volunteer clusterhead with $p_m = 0.9512$. On the other hand, it takes

21 election rounds to detect 83% of malicious nodes when $p_m = 3p_{opt}$. This result might be misleading in the sense that one might think it takes too long to detect malicious nodes. However, it should be noted that when attackers chose a p_m relatively close to p_{opt}, they are in fact, not behaving maliciously in the first few election rounds and volunteer as expected. It is in the later rounds that they start to deviate from the protocol and act maliciously. Once they start to act maliciously, our proposed anomaly detection algorithm can capture they misbehavior effectively.

The ultimate goal of the proposed detection strategy is to eliminate any legitimate traffic toward malicious nodes without incorrectly removing legitimate nodes. Our proposed solution effectively detects malicious nodes misbehaviors without introducing any false positives. The self-monitoring approach can almost immediately detect extreme attacks in which the attacker volunteers to be a clusterhead with a high probability.

In Fig. 5, we illustrate the result of applying CUSUM combined with the self-monitoring method. Attackers that become volunteer clusterheads with a probability significantly greater than the optimal probability are almost immediately detected. These attackers gain a higher percentage of traffic control initially. However, they lose their control dramatically after only a few rounds of cluster formation. On the contrary, detection of the attackers that volunteer with a probability close to the optimal probability requires more time, however, these types of attacker have a low impact.

6 Conclusions

We analyzed the performance of EEHCA, a probabilistic energy based clustering algorithm against security compromise. Our results demonstrate a significant vulnerability in EEHCA performance when compromised nodes exist. We then presented a detection strategy to detect anomalous nodes effectively. We showed that when CUSUM test is combined with a self-monitoring approach, anomalous nodes are detected quickly without removing legitimate nodes falsely.

Acknowledgments. Research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-13-2-0045 (ARL Cyber Security CRA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

A Proof of Theorem 1

Proof. Suppose a realization of the random point process Φ_b with number of nodes equal to $N = n$ is given. Arbitrarily compromise m nodes independent of location and each other under attack model 1. One can consider compromising

nodes as a p-thinning operation with $p_r = 1 - \frac{m}{n}$. As a result, legitimate nodes and malicious nodes are distributed independently according to a homogeneous spatial Poisson point processes (PPP):

$$PPP_{leg} : \lambda_{leg} = p_r \lambda_b \tag{12}$$

$$PPP_{mal} : \lambda_{mal} = (1 - p_r)\lambda_b \tag{13}$$

Legitimate nodes become volunteer clusterheads with probability of p_{opt}. Hence, legitimate clusterheads and legitimate ordinary nodes are distributed as independent homogeneous spatial Poisson point processes:

$$PPP_{CH,leg} : \lambda_{CH,leg} = p_{opt} \lambda_{leg} \tag{14}$$

$$PPP_{Ord,leg} : \lambda_{Ord,leg} = (1 - p_{opt})\lambda_{leg} \tag{15}$$

Malicious nodes become volunteer clusterheads with probability of p_m and therefore, malicious clusterheads are distributed as a homogeneous spatial Poisson point process:

$$PPP_{CH,mal} : \lambda_{CH,mal} = p_m \lambda_{mal} \tag{16}$$

It should be noted that PPP_{mal} and PPP_{leg} are non-overlapping and independent and hence, $PPP_{CH,mal}$ and $PPP_{CH,leg}$ are independent non-overlapping point processes. Consequently, one can apply the Superposition operation and define:

$$PPP_{CH} = PPP_{CH,leg} \cup PPP_{CH,mal} \tag{17}$$

which is a spatial Poisson point process with intensity of:

$$\lambda_{CH} = \lambda_{CH,leg} + \lambda_{CH,mal} = p_{opt}(1 - \frac{m}{n})\lambda_b + p_m \frac{m}{n} \lambda_b \tag{18}$$

Similarly, one can derive the intensity of $PPP_{Ord,leg}$ as

$$\lambda_{Ord,leg} = (1 - p_{opt})(1 - \frac{m}{n})\lambda_b \tag{19}$$

Denoting the number of $PPP_{Ord,leg}$ process points in each Voronoi cell by the random variable $N_{Ord,leg}$ and applying the results of [5], we get:

$$\mathbb{E}[N_{Ord,leg}|N = n] \approx \mathbb{E}[N_{Ord,leg}] = \frac{\lambda_{Ord,leg}}{\lambda_{CH}} = \frac{(1 - p_{opt})(n - m)}{p_{opt}(n - m) + m p_m} \tag{20}$$

where $\mathbb{E}$ denotes expected value operation.

Since there are m malicious nodes and each of them becomes a clusterhead with probability p_m, there are on expectation mp_m cells having a malicious clusterhead. As a result, the expected number of legitimate ordinary nodes belonging to clusters with a malicious clusterhead is:

$$L_m = m p_m \mathbb{E}[N_{Ord,leg}|N = n] \tag{21}$$

On the other hand, there are a total of $L_{leg} = (n-m)(1-p_{opt})$ legitimate ordinary nodes on expectation. Hence, one can compute the percentage of legitimate ordinary nodes served by a malicious clusterhead under attack model 1 from:

$$q_{affected} = \frac{L_m}{L_{leg}} = \frac{mp_m}{p_{opt}(n-m) + mp_m} \tag{22}$$

From (22), more than 50% of legitimate ordinary nodes would belong to clusters with malicious clusterheads if the fraction of the compromised nodes satisfies:

$$\frac{m}{n} \geq \frac{p_{opt}}{p_m + p_{opt}} \tag{23}$$

Alternatively, for a given n and m, an adversary causes more than 50% of legitimate ordinary nodes to belong to clusters with malicious clusterheads if it sets its probability of becoming volunteer clusterhead to:

$$p_m \geq (\frac{n}{m} - 1)p_{opt} \tag{24}$$

B Proof of Theorem 2

Proof. Before any attack has been launched (i.e. nodes become clusterhead with probability of p_{opt}), clusterheads and ordinary nodes form two homogeneous, independent, non-overlapping spatial Poisson point processes with intensities of $p_{opt}\lambda_b$ and $(1-p_{opt})\lambda_b$, respectively. Consequently, one can derive the expected number of ordinary nodes in each cluster in an attack-free system from [5]:

$$\mathbb{E}[N_{Ord,Afree}|N=n] \approx \frac{\lambda_{Ord,Afree}}{\lambda_{CH,Afree}} = \frac{(1-p_{opt})}{p_{opt}} \tag{25}$$

Now consider a system under attack model 2. Similar to the analysis presented in Appendix A, legitimate clusterheads form a spatial Poisson process:

$$PPP_{CH,leg} : \lambda_{CH,leg} = p_{opt}(1 - \frac{m}{n})\lambda_b \tag{26}$$

Legitimate ordinary nodes and malicious ordinary nodes are two independent non-overlapping point processes, and hence, their union is also a spatial Poisson point process, PPP_{ord}:

$$PPP_{ord} = PPP_{ord,leg} \cup PPP_{ord,mal} \tag{27}$$

And hence:

$$\lambda_{Ord} = \lambda_{ord,leg} + \lambda_{ord,mal} = (1-p_{opt})(1-\frac{m}{n})\lambda_b + (1-p_m)\frac{m}{n}\lambda_b \tag{28}$$

Denote the number of ordinary nodes in each cluster under attack model 2 by the random variable $N_{Ord,Amodel2}$. Then:

$$\mathbb{E}[N_{Ord,Amodel2}|N=n] \approx \frac{\lambda_{Ord}}{\lambda_{CH,leg}} = \frac{(1-p_{opt})(n-m) + (1-p_m)m}{p_{opt}(n-m)} \tag{29}$$

Therefore, the ratio is computed from:

$$\frac{\mathbb{E}[N_{Ord,Amodel2}|N=n]}{\mathbb{E}[N_{Ord,Afree}|N=n]} = \frac{(n-m)+(1-p_m)m}{(n-m)} \tag{30}$$

According to [6], p_{opt} is very small for networks with density higher than 10. Since the attacker selects a p_m even smaller than p_{opt} under attack model 2, one can approximate this ratio by:

$$\frac{\mathbb{E}[N_{Ord,Amodel2}|N=n]}{\mathbb{E}[N_{Ord,Afree}|N=n]} \approx \frac{1}{(1-\frac{m}{n})}, \quad m \neq n \tag{31}$$

C Proof of Theorem 3

Proof. By applying the results of [6], the total energy to send 1 unit of data to the clusterhead by ordinary nodes of a Voronoi cell in an attack-free system (C_1^{Afree}) is computed from:

$$\mathbb{E}[C_1^{Afree}|N=n] \approx \frac{\lambda_{Ord,Afree}}{2r\lambda_{CH,Afree}^{3/2}} = \frac{(1-p_{opt})}{2rp_{opt}^{3/2}\lambda_b^{1/2}} \tag{32}$$

Likewise, for a system under attack model 2, one can calculate the expected value of the total energy spent by legitimate ordinary nodes in a cluster to send a unit of data to the clusterhead, $C_1^{Amodel2}$ by:

$$\mathbb{E}[C_1^{Amodel2}|N=n] \approx \frac{\lambda_{Ord,leg}}{2r\lambda_{CH}^{3/2}} = \frac{(1-p_{opt})(1-\frac{m}{n})}{2r(p_{opt}(1-\frac{m}{n})+p_m\frac{m}{n})^{3/2}\lambda_b^{1/2}} \tag{33}$$

By assuming p_m is much smaller than p_{opt}, we approximate the total energy spent by legitimate ordinary nodes in a cluster under attack model 2 by:

$$\mathbb{E}[C_1^{Amodel2}|N=n] \approx \frac{(1-p_{opt})}{2rp_{opt}^{3/2}[(1-\frac{m}{n})\lambda_b]^{1/2}} \tag{34}$$

Hence, the ratio of $\mathbb{E}[C_1^{Amodel2}|N=n]$ to $\mathbb{E}[C_1^{Afree}|N=n]$ is computed from:

$$\frac{\mathbb{E}[C_1^{Amodel2}|N=n]}{\mathbb{E}[C_1^{Afree}|N=n]} \approx \frac{1}{(1-\frac{m}{n})^{1/2}}, \quad m \neq n \tag{35}$$

References

1. Abbasi, A.A., Younis, M.: A survey on clustering algorithms for wireless sensor networks. Comput. Commun. **30**(14), 2826–2841 (2007)
2. Agresti, A.: An Introduction to Categorical Data Analysis. Wiley Series in Probability and Statistics. Wiley, Hoboken (2007)

3. Al-Karaki, J.N., Kamal, A.E.: Routing techniques in wireless sensor networks: a survey. Wireless Commun. **11**(6), 6–28 (2004)
4. Andersson, E., Bock, D., Frisén, M.: Some statistical aspects of methods for detection of turning points in business cycles. J. Appl. Stat. **33**(3), 257–278 (2006)
5. Baccelli, F., Zuyev, S.: Poisson-Voronoi spanning trees with applications to the optimization of communication networks. Oper. Res. **47**(4), 619–631 (1999)
6. Bandyopadhyay, S., Coyle, E.J.: An energy efficient hierarchical clustering algorithm for wireless sensor networks. In: INFOCOM 2003, Twenty-Second Annual Joint Conference of the IEEE Computer and Communications, IEEE Societies, vol. 3, pp. 1713–1723. IEEE (2003)
7. Chiu, S.N., Stoyan, D., Kendall, W.S., Mecke, J.: Stochastic Geometry and Its Applications. Wiley, Hoboken (2013)
8. Choi, J., Lee, C.: Energy consumption and lifetime analysis in clustered multihop wireless sensor networks using the probabilistic cluster-head selection method. EURASIP J. Wireless Commun. Netw. **2011**(1), 1–13 (2011)
9. Hawkins, D.M., Olwell, D.H.: Cumulative Sum Charts and Charting for Quality Improvement. Statistics for Engineering and Physical Science. Springer, New York (1998). https://doi.org/10.1007/978-1-4612-1686-5
10. Heinzelman, W.R., Chandrakasan, A., Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In: Proceedings of the 33rd Annual Hawaii International Conference on System Sciences 2000, p. 10. IEEE (2000)
11. Kang, C.W., Kvam, P.H.: Shewhart control charts. In: Basic Statistical Tools for Improving Quality, pp. 97–124 (2011)
12. Liu, F., Cheng, X., Chen, D.: Insider attacker detection in wireless sensor networks. In: INFOCOM 2007, 26th IEEE International Conference on Computer Communications, pp. 1937–1945. IEEE (2007)
13. Liu, X.: A survey on clustering routing protocols in wireless sensor networks. Sensors **12**(8), 11113–11153 (2012)
14. Page, E.: Continuous inspection schemes. Biometrika **41**, 100–115 (1954)
15. Perrig, A., Szewczyk, R., Tygar, J.D., Wen, V., Culler, D.E.: SPINS: security protocols for sensor networks. Wireless Netw. **8**(5), 521–534 (2002). https://doi.org/10.1023/A:1016598314198
16. Poor, H.V., Hadjiliadis, O.: Quickest Detection, vol. 40. Cambridge University Press, Cambridge (2009)
17. Reynolds, M.R.: The Bernoulli CUSUM chart for detecting decreases in a proportion. Qual. Reliab. Eng. Int. **29**(4), 529–534 (2013)
18. Reynolds, M.R., Stoumbos, Z.G.: A general approach to modeling CUSUM charts for a proportion. IIE Trans. **32**(6), 515–535 (2000)
19. Reynolds Jr., M.R., Stoumbos, Z.G.: A CUSUM chart for monitoring a proportion when inspecting continuously. J. Qual. Technol. **31**(1), 87–108 (1999)
20. Sego, L.H.: Applications of control charts in medicine and epidemiology. Ph.D. thesis, Virginia Polytechnic Institute and State University (2006)
21. Stoto, M.A., et al.: Evaluating statistical methods for syndromic surveillance. In: Wilson, A.G., Wilson, G.D., Olwell, D.H. (eds.) Statistical Methods in Counterterrorism, pp. 141–172. Springer, New York (2006). https://doi.org/10.1007/0-387-35209-0_9
22. Tartakovsky, A.G., Rozovskii, B.L., Shah, K.: A nonparametric multichart CUSUM test for rapid intrusion detection. In: Proceedings Joint Statistical Meetings, 7–11 August 2005

23. Wang, H., Sheng, B., Li, Q.: Elliptic curve cryptography-based access control in sensor networks. Int. J. Secur. Netw. **1**(3–4), 127–137 (2006)
24. Xie, M., Han, S., Tian, B., Parvin, S.: Anomaly detection in wireless sensor networks: a survey. J. Netw. Comput. Appl. **34**(4), 1302–1325 (2011)
25. Younis, O., Fahmy, S.: HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. IEEE Trans. Mob. Comput. **3**(4), 366–379 (2004)
26. Zhang, Y., Yang, L.T., Chen, J.: RFID and Sensor Networks: Architectures, Protocols, Security, and Integrations. CRC Press, Boca Raton (2009)