Anomaly Detection Based Secure In-Network Aggregation for Wireless Sensor Networks

Bo Sun, *Member, IEEE,* Xuemei Shan, Kui Wu, *Senior Member, IEEE,* and Yang Xiao, *Senior Member, IEEE*

***Abstract*—Secure in-network aggregation in wireless sensor networks (WSNs) is a necessary and challenging task. In this paper, we first propose integration of system monitoring modules and intrusion detection modules in the context of WSNs. We propose an extended Kalman filter (EKF) based mechanism to detect false injected data. Specifically, by monitoring behaviors of its neighbors and using EKF to predict their future states (actual in-network aggregated values), each node aims at setting up a normal range of the neighbors' future transmitted aggregated values. This task is challenging because of potential high packet loss rate, harsh environment, and sensing uncertainty. We illustrate how to use EKF to address this challenge to create effective local detection mechanisms. Using different aggregation functions (average, sum, max, and min), we present how to obtain a theoretical threshold. We further apply an algorithm of combining cumulative summation and generalized likelihood ratio to increase detection sensitivity. To overcome the limitations of local detection mechanisms, we illustrate how our proposed local detection approaches work together with the system monitoring module to differentiate between malicious events and emergency events. We conduct experiments and simulations to evaluate local detection mechanisms under different aggregation functions.**

***Index Terms*—Cumulative summation (CUSUM), extended Kalman filter, generalized likelihood ratio (GLR), in-network aggregation, intrusion detection systems, wireless sensor networks (WSNs).**

I. Introduction

WIRELESS sensor networks (WSNs) can provide effective and economically viable solutions for a large variety of applications, such as health monitoring, scientific data collection, environmental monitoring, and military operations. However, sensor nodes in these applications could easily be compromised and can inject arbitrarily falsified values into the networks.

In-network aggregation has been proven to be an important primitive to reduce the communication overhead and to save energy for WSNs. Many aggregation protocols have been proposed and their performance has been evaluated [2]–[14]. However, only a few protocols consider secure in-network aggregation based on a prevention-based scheme, in which encryption, authentication, and key management are used. Once a sensor node is compromised, all its associated secrets become open to attackers, rendering prevention-based techniques helpless. To solve this problem, intrusion detection systems (IDSs), which serve as the second wall of protection, can effectively help identify malicious activities. Unfortunately, there is very little work that aims at addressing the secure in-network aggregation problem from an intrusion detection perspective.

In this paper, to enhance WSN security, we propose that system monitoring modules (SMM) should be integrated with intrusion detection modules (IDM) in the context of WSNs. In practice, WSNs are often deployed to monitor important emergency events, such as forest fires and battlefield monitoring. This integration can facilitate classification between malicious events and important emergency events. For example, using IDM, when node A raises an alert on node B because of an event E, A can further initiate investigation on E with the help of SMM. Specifically, A can wake up relevant sensor nodes around B and request their opinions about E. If the majority of sensor nodes think that E could happen, A can make a decision that E is triggered by some emergency event. Otherwise, A can suspect that E is malicious.

We first propose an extended Kalman filter (EKF) based mechanism to detect false injected data. Specifically, by monitoring behaviors of its neighbors and using EKF to predict actual in-network aggregated values (states), each node aims at setting up a normal range of neighbors' future transmitted aggregated values. This task is challenging because of potential high packet loss rate [18], harsh environment, sensing inaccuracy, time asynchrony between children and parents' nodes, and so on. All these factors contribute to uncertainties. By utilizing a state-space model [25], an EKF-based mechanism is suitable for WSN nodes because this mechanism may address those incurred uncertainties in a lightweight manner and compute relatively accurate estimates of aggregated values, based on which a normal range can be approximated. Utilizing a threshold-based mechanism, a promiscuously overheard value is then compared with a locally computed normal range to decide whether they are significantly different. We then analyze how to decide the thresholds under different aggregation functions (average, sum, max, and min).

Manuscript received July 19, 2011; revised April 2, 2012; accepted May 8, 2012. Date of publication November 16, 2012; date of current version February 20, 2013. This work was supported in part by the U.S. National Science Foundation under Grants CNS-0922888, CNS-0716211, and CCF-0829827.

B. Sun is with the Department of Computer Science, Lamar University, Beaumont, TX 77710 USA (e-mail: bsun@lamar.edu).

K. Wu is with the Department of Computer Science, University of Victoria, Victoria, BC V8W 3P6, Canada (e-mail: wkui@cs.uvic.ca).

Y. Xiao is with the Department of Computer Science, University of Alabama, Tuscaloosa, AL 35487 USA (e-mail: yangxiao@ieee.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSYST.2012.2223531

To increase detection sensitivity when malicious values have small deviations, we further apply an algorithm of combining cumulative summation (CUSUM) and generalized likelihood ratio (GLR) [1]. However, this incorporation incurs computation overhead to resource-constrained sensor nodes. To decrease this overhead, we simplify the computation and make the CUSUM GLR algorithm suitable for sensor nodes.

We conduct experiments and simulations to evaluate EKF based and CUSUM GLR based local detection mechanisms using different aggregation functions. Our implementation of EKF and CUSUM GLR on representative sensor node MICA2 motes [23] demonstrates that our proposed scheme is practical on resource stringent hardware.

The remainder of this paper is organized as follows. Section II describes the related work. In Section III, we present our motivations, network model, and assumptions. In Section IV, we present our security model. In Section V, we propose secure in-network aggregation algorithms based on EKF and CUSUM GLR under different aggregation functions. We further elaborate on how IDM and SMM work together to differentiate between malicious events and emergency events. In Section VI, we evaluate our proposed mechanisms. Section VII concludes this paper and points out important future work.

II. Related Work

There are many research efforts that address aggregation problems [10]–[14] in WSNs. However, none of the aforementioned protocols considers secure aggregation problems until recently. Hu and Evans [7] tackled the problem of information aggregation in which one node is compromised. Their protocol might be vulnerable if both a child node and its parent node are compromised. Yang *et al.* [8] proposed a secure hop-by-hop data aggregation protocol based on principles of divide-and-conquer and commit-and-attest. Przydatek *et al.* [6] proposed an aggregate-commit-prove framework to design secure data aggregation protocols. Chan *et al.* [17] presented an optimally secure aggregation scheme for arbitrary aggregator topologies and multiple malicious nodes. Wagner [2] used statistical estimation to design more resilient aggregation schemes against malicious data injection attacks. In his work, a mathematical framework is presented to formally evaluate security of different aggregation algorithms. However, no detailed simulations and experiments are carried out in [2]. Moreover, [2] does not consider in-network aggregation. Our work improves over [2] in these aspects. Wu *et al.* [9] proposed a secure aggregation tree to detect and prevent cheating in WSNs, in which the detection of cheating is based on topological constraints in a constructed aggregation tree.

There are some resilient aggregation algorithms aiming to increase the likelihood of accurate results when WSNs are prone to message loss and node failure [14]–[16]. Also, a number of proposed protocols aim to ensure the secrecy and authentication of data [3]–[5] in WSNs.

Several protocols are proposed to filter false data in WSNs [29]–[31]. Generally, they utilize different key distribution mechanisms to develop filtering capabilities. In these research efforts, different sensing reports are validated by message authentication codes along the way to the sink. The sink can further filter out remaining false reports that escape the filtering en route.

There are also some research efforts using statistical approaches like a Bayesian algorithm and decision fusion [36]–[38] to take into account the possibility of sensor measurement faults. These research efforts have turned out to be important for applications, including target detection, data query, and event region detection. Kalman filter (KF) and CUSUM GLR have also been widely used in various applications. For example, in the context of WSNs, KF was used to enable accurate target tracking [40]. Based on nonparametric CUSUM, [41] proposed two local detector algorithms from sequential sensor readings to enable distributed detection in WSNs. However, to the best of our knowledge KF and CUSUM have not yet been applied to secure WSN aggregation services.

Unlike existing techniques, our work aims at addressing secure in-network aggregation problems from an intrusion detection perspective. Our work relies on predicted aggregated values in an efficient online manner and can complement existing aggregation protocols to considerably enhance WSN security.

III. Motivations, Network Model, and Assumptions

A. Motivations

Consecutive observations of sensor nodes are usually highly correlated in time domains [21]. This correlation, along with the collaborative nature of WSNs, makes it possible to predict future observed values based on previous values. This motivates our proposed local detection algorithms. Furthermore, since WSNs are usually densely deployed, nodes close to each other can have spatially correlated observations, which can facilitate the collaboration of sensor nodes in proximity to differentiate between malicious events and important emergency events. This motivates us to integrate SMM and IDM in order to achieve accurate detection results.

B. Network Model

To utilize data aggregation, an aggregation tree is often built first. Fig. 1(a) is one example of such an aggregation tree. In Fig. 1(a), A, B, C, and D perform sensing tasks, obtain values and transmit them to their parent node H. H aggregates (min, max, sum, average, etc.) the received values from A, B, C, and D, and transmits the aggregated value further up to node K. The same is true for operation $(E, F, G) \rightarrow I \rightarrow J$ and operation $(M, N) \rightarrow L \rightarrow J$. These aggregation operations are performed based on the established parent–child relationship, which can be modeled using Fig. 1(b). In Fig. 1(a), the base station collects all these data and, if necessary, can transmit them across the Internet.

C. Assumptions

WSNs are often deployed to monitor emergency events such as forest fires. We do not assume time synchronization among nodes. Our proposed approach can tolerate the time inaccuracy caused by child nodes and parent nodes. In the context of WSNs, time synchronization still incurs expensive operations.

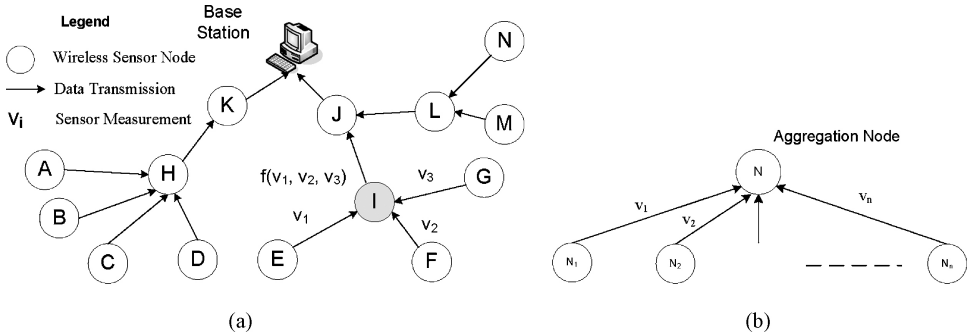


Fig. 1. Wireless sensor networks. (a) Example aggregation tree. (b) Aggregation model.

We assume that promiscuous mode is supported by sensor nodes. By enabling promiscuous mode, when one node, e.g., *F* in Fig. 1(a), is within the radio transmission range of another node, e.g., *I*, node *F* can overhear node *I*'s transmissions. This facilitates our proposed neighbor monitoring mechanisms.

For the purpose of saving node energy, there have been extensive research efforts on various kinds of sensor node scheduling policies, in which a minimum number of nodes remain awake to satisfy a certain degree of coverage. Therefore, we assume that sensor nodes may go to sleep. However, we also assume that necessary sensor nodes could be woken up anytime once required. We realize that in a real system, it may need nonzero time to allow a node to become fully functional. Therefore, our proposed scheme may not work very well if the period of the attackers' false injection is very short. This can be an open problem, and will be explored in our our future work.

In this paper, to simplify the study, and similar to many scheduling papers in sensor networks, we assume that any node can be woken up anytime immediately. However, in a real system in practice, it may need nonzero time to allow node to become fully functional so that the proposed waking-up-other node scheme may not work very well if the period of the attackers' false injection is very short. Such a problem can be an open topic that needs further study.

Please note that if one node is in sleep mode, this node does not need to be in promiscuous mode. Existing research in self-monitoring for sensor networks (such as [39]) can be integrated with our solution so that each active communication link can be monitored by nodes in the WSN. Moreover, in order to monitor sensor behaviors, there is an inevitable tradeoff of adopting promiscuous mode. Actually, any related work in this aspect cannot avoid this.

IV. Security Model

When a sensor node is compromised by an adversary, this adversary can take full control of the compromised node. It may inject falsified data readings or nonexistent readings into the WSN. We also assume that falsified data transmitted by a compromised node is significantly different from the state (the actual value, for example, the actual monitored average temperature) so that falsified data can effectively disrupt aggregation operations.

If the adversary only injects a limited number of falsified data that are slightly different from true aggregated values, this will not cause significant impact on deployed applications. Therefore, we will also consider an attack model that an adversary continuously forges falsified data with small deviations.

We assume that the majority of nodes around some unusual events are not compromised. It will become an open research problem if this assumption does not hold.

V. Secure In-Network Aggregation

Our proposed protocol is equipped with two modules: IDM and SMM. The functionality of the IDM is to detect whether monitored nodes are malicious insider nodes, while the functionality of the SMM is to monitor important emergency events. Note that SMM is a necessary component for most of WSN applications. IDM and SMM need to be integrated with each other to work effectively. Relying on local detection alone is not desirable because each node has only very limited information available. Furthermore, since sensor nodes are prone to failure, it is very difficult to differentiate between emergency events sent by good nodes and malicious events. In our proposed scheme, whenever IDM and SMM detect some abnormal events, they need to request the collaboration of more sensor nodes around the events to make a final decision.

For the IDM, our general idea is like the mechanism proposed in [27]. Node *A* promiscuously overhears its neighbor's transmitted aggregated value and compares it with the predicted normal range. If the overheard value lies outside the normal range, either an event *E* happens or the neighbor *N* then becomes a suspect. To tell whether node *N* is a malicious node or *E* is an important emergency event like the breakout of a forest fire, *A* initiates the collaboration between IDM and SMM by waking up relevant sensor nodes around *N* and requesting their opinions about *E*. Please note that our proposed detection solution and the solution adopted in [27] are completely different.

A. Challenges

Many challenges exist when we try to predict the normal range of in-network aggregated values in a lightweight manner. First, it is difficult to achieve actual aggregate values because of many sources of potential uncertainties. WSNs suffer from a high packet loss rate. For example, based on [18], in an

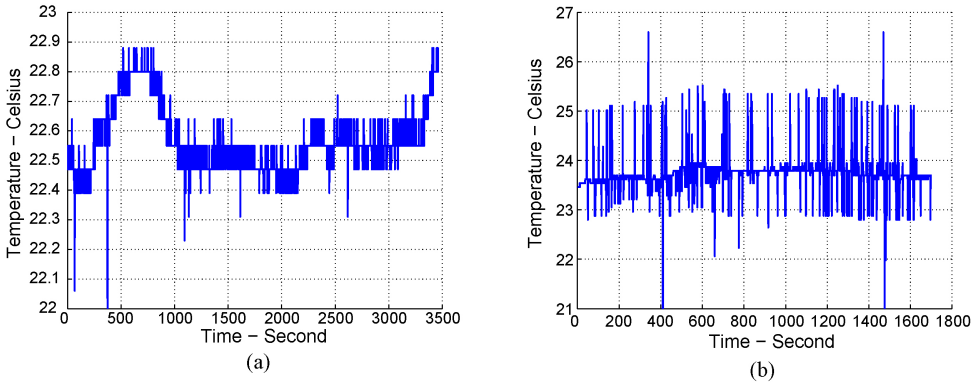


Fig. 2. Lab setting measurement. (a) Sensing inaccuracy of WSN nodes. (b) Output of the average aggregation.

in-building environment, with 62 motes deployed with the granularity of one mote per office, at a low load of 0.5 packet per second, there is around 35% of links whose packet loss is worse than 50% at a medium access control layer. Therefore, even a reasonable link layer loss recovery is unable to mask high packet losses.

For aggregation protocols, the lack of time synchronization among children and parent nodes may make aggregation nodes use different sets of values for aggregation. The complexity of existing aggregation protocols also contributes to the challenges of modeling in-network aggregated values. In [10], it shows that for periodic aggregation, timing, i.e., how long a node waits to receive data from its children (downstream nodes in respect to the information sink) before forwarding data onto the next hop plays a crucial role in the performance of aggregation algorithms in the context of periodic data generation.

Furthermore, individual sensor readings are subject to environmental noise. To demonstrate this, we set up a simple one-hop WSN testbed, in which node A periodically transmits sensed values to a base station. Node A consists of a MICA2 mote and a MTS310 sensor board [24]. In a lab setting, we measure the collected data, as shown in Fig. 2(a). We conduct a further experiment to demonstrate the uncertainty of the average aggregation function. In this experiment, we deploy four sensors to send their sensed temperature to an aggregation node B. B periodically computes the average of the received values. The average values are illustrated in Fig. 2(b). Fig. 2(a) and (b) illustrates that data captured from a physical world and the aggregated values based on these data tend to be noisy.

Sensor nodes suffer from stringent resources, which prevent the usage of some powerful yet expensive estimation and prediction approaches. To enable neighbor monitoring mechanisms, we need a lightweight scheme that can be efficiently executed by sensor nodes. In this respect, we use an approach based on EKF for each node to predict and estimate future values of its neighbors, as we detail in the next section.

B. Extended Kalman Filder Based Local Detection

1) *Extended Kalman Filder:* Based on a state-space model, KF [25] addresses a general problem of trying to estimate a state of a dynamic system perturbed by Gaussian white noise, using measurements that are linear functions of the system state, but corrupted by additive Gaussian white noise. Extended with linear estimation theory, EKF can be applied to many nonlinear applications by approximating effects of small perturbations linearly. By setting a proper process model and measurement model for a specific WSN application and utilizing time update and measurement update equations to recursively process data, we can use EKF to obtain a relatively accurate estimate of state [25].

TABLE I
NOTATIONS FOR EKF

Symbol	Meaning
x_k	State: the actual value at time t_k
F	Function relating x_{k+1} to x_k
$\hat{x}_k^-$	*A priori* estimate of x_k
$\hat{x}_k^+$	*A posteriori* estimate of x_k
z_k	Measurement (measured value) at time t_k
H	Function relating x_k to z_k
w_k	Process noise at time t_k
Q_k	Variance of w_k at time t_k
v_k	Measurement noise at time t_k
R_k	Variance of v_k at time t_k
P_k^-	*A priori* estimate error at time t_k
P_k^+	*A posteriori* estimate error at time t_k
K_k	Kalman gain at time t_k

In our case, state represents an actual value to be measured. State at a given instant of time is characterized by instantaneous values of an attribute of interest, for example, actual temperature monitored by WSNs. In the following sections, we make state and actual exchangeable. Because actual temperature is perturbed by various uncertainties in WSNs, it is highly possible for aggregation nodes to obtain values other than state values. Aggregation nodes can only obtain measured values and estimate actual values. Therefore, proper models are necessary when EKF is applied to WSNs under different applications.

Aggregation nodes calculate aggregated values periodically. Therefore, we adopt a discrete-time EKF [25], in which a system state is estimated at a discrete set of times t_k, where $k = 0, 1, \ldots,$ These discrete times correspond to the times at which a value is measured and a state is estimated.

2) *System Dynamic Model:* Table I lists the notations that are used in the following sections. Actual aggregated values form a dynamic process, and a process model, given in (1), governs the evolution of this process

$$x_{k+1} = F(x_k) + w_k. \tag{1}$$

In (1), function F relates the state at time step t_k to the state at time step t_{k+1}. Obviously, $F(x_k)$ is application dependent and its discussion is beyond the scope of this paper. For example, if WSNs are deployed to monitor the average indoor temperature of a building, $F(x_k)$ may be set to x_k. If the monitored temperature decreases gradually in a time period, one possible $F(x_k)$ may be set to δx_k, where δ is a positive value less than 1. In practice, because of the complex nature of monitored phenomena, $F(x_k)$ may take a complicated form. Therefore, necessary simplifications of $F(x_k)$ may be needed after careful analysis of deployed applications. One further problem is that for a given application, different forms of $F(x_k)$ may be used to characterize the state change. One possible solution to this problem, if needed, is that base stations may broadcast a new form of $F(x_k)$ over the network to adjust the process model.

As a dynamic system, the state of an application has variation, which is reflected in w_k. w_k is the process noise at time t_k and is usually modeled as a normal random variable. We further assume that w_k follows normal distribution $N(0, Q)$, where Q, a constant parameter, is the variance of w_k. Note that Q may also be broadcasted over the network by the base station to adapt to changing environment.

Measurement model is given in

$$z_k = H(x_k) + v_k = x_k + v_k. \tag{2}$$

z_k is the measured value at time t_k. For example, in Fig. 1(a), node I sends out an aggregated value z_k at time t_k, node E, F, and G can overhear this value. $x_k \in \Re$ ($\Re$ denotes the set of real numbers) is the state to be monitored at time t_k and represents the actual aggregated value of the area that aggregation node I covers. v_k is the measurement noise, representing noisy sensor measurements and various uncertainties in WSNs. Again, for a specific application, we assume that v_k follows a normal distribution with mean 0 and variance R, denoted as $N(0, R)$, where R is the variance of v_k. Note that R can also be adjusted by the base station.

3) *System Equations:* In the following, we list important system equations based on the system models, as presented in (1) and (2). For details about the derivation of these equations, please refer to [25].

A time update-state estimate equation is used to predict the state $\hat{x}_{k+1}^-$ at time t_{k+1}

$$\hat{x}_{k+1}^- = F(\hat{x}_k^+). \tag{3}$$

A time update-error update equation is used to predict estimate errors at time t_{k+1}

$$P_{k+1}^- = \frac{\partial F}{\partial x}|_{x=\hat{x}_k^+} P_k^+ \frac{\partial F}{\partial x}|_{x=\hat{x}_k^+}^T + Q_k = P_k^+ + Q_k. \tag{4}$$

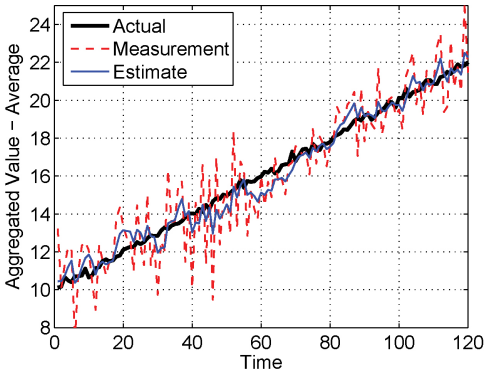


Fig. 3. Prediction accuracy of discrete-time EKF.

A measurement update-Kalman gain equation is used to compute the Kalman gain at time t_{k+1}

$$K_{k+1} = P_{k+1}^-(P_{k+1}^- + R_k)^{-1} = \frac{P_{k+1}^-}{P_{k+1}^- + R_k}. \tag{5}$$

A measurement update-estimate update with measurement z_{k+1} equation is used to update estimate with measurement z_{k+1}

$$\hat{x}_{k+1}^+ = \hat{x}_{k+1}^- + K_{k+1}(z_{k+1} - \hat{x}_{k+1}^-). \tag{6}$$

A measurement update-error covariance update equation is used to update estimate errors

$$P_{k+1}^+ = (I - K_{k+1}H_{k+1})P_{k+1}^{-1} = (1 - K_{k+1})P_{k+1}^-. \tag{7}$$

The time update equations (3) and (4) are responsible for predicting the state ($\hat{x}_{k+1}^-$) and estimate error (P_{k+1}^-) at time t_{k+1}. In (4), applying a first-order Taylor series approximation to $F(x)$, $\frac{\partial F}{\partial x}|_{x=\hat{x}_k^+}$ is the value of the first-order partial derivative of F with respect to x at $x = \hat{x}_k^+$. Because the state is a scalar variable in our case, $P_{k+1}^- = P_k^+ + Q_k$.

The measurement update equations (5) and (6) are responsible for incorporating the new measured value z_{k+1} into the *a priori* estimate $\hat{x}_{k+1}^-$ to compute *a posteriori* estimate ($\hat{x}_{k+1}^+$). Note that because $H(x_k) = x_k$, $H_k = \frac{\partial H}{\partial x} = 1$. Therefore, H_k can be ignored in (6). Equation (7) is used to update the estimate error.

Basically, at time t_k, to predict the actual value x_{k+1}, a node needs two values: 1) the *a priori* estimate $\hat{x}_{k+1}^-$, which can be obtained based on (3); and 2) the measured value z_{k+1} that can be promiscuously overheard. Then, based on (6), a relatively accurate estimate of x_{k+1} can be computed. The second part of (6) indicates the adjustment of $\hat{x}_{k+1}^-$ based on the difference between the *a priori* estimate $\hat{x}_{k+1}^-$ and the measured value z_{k+1}.

EKF can provide a relatively accurate prediction of neighbors' future aggregated values. To illustrate this point, using the network topology shown in Fig. 1(b) and the average as an example aggregation function, we plot the actual, measurement, and estimate value using the system equations described in Section V-B3. Specifically, we simulate environment whose actual temperature increases smoothly and is perturbed by Gaussian white noise. Sensor nodes N_i in Fig. 1(b) sense the temperature (to simulate sensing inaccuracy, sensed temperature is the actual temperature

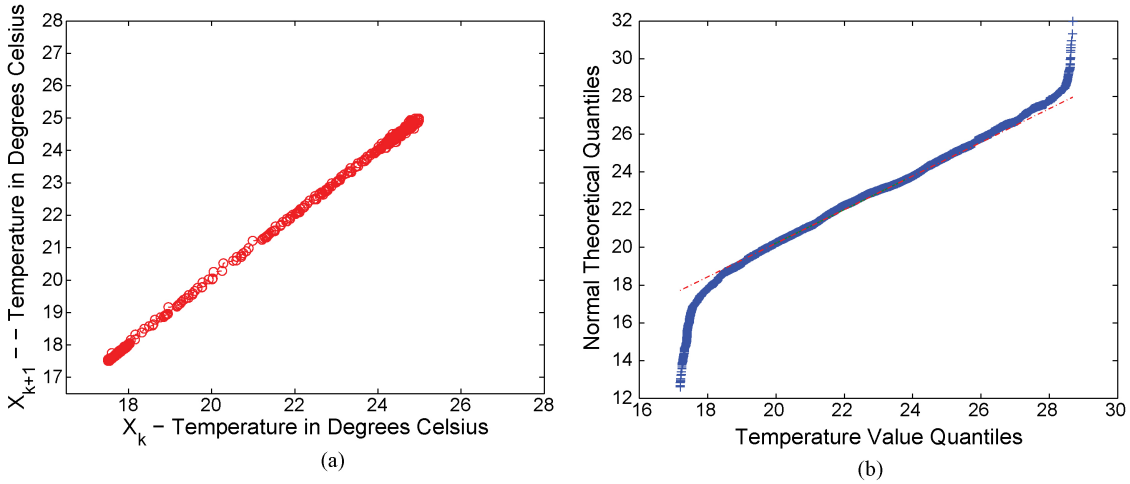


Fig. 4. Intel Lab data. (a) Relationships between x_k and x_{k+1}. (b) Q-Q plot. Most of data points fall almost perfectly along the line, which is a good indicator that w_k is normally distributed.

perturbed by random noise) and periodically transmit sensed values to their parent node, node N. Node N periodically calculates the average of the received values, which are measured values. Estimated values are the estimates of measured values by each N_i using EKF. In Fig. 1(b), we assume that a packet sent from N_i to N has 0.3 loss probability.

The result is illustrated in Fig. 3. From Fig. 3, we can see that although many kinds of uncertainties exist, an EKF based approach can still provide a relatively accurate prediction of the actual value.

4) *Model Fitting Using Real Data Set:* F in (1) is application dependent. For example, we can use Intel Lab Data [33], a commonly used data set, to plot the relationship F between x_k and x_{k+1} in an environment similar to the Intel Berkeley Research Laboratory.

We randomly pick one sensor node, filter out its faulty readings (i.e., those readings that deviate much from both immediately previous and following readings), and select one time period in which temperature readings keep increasing. Based on the readings in this time period, we plot the relationship between x_k and x_{k+1}, as illustrated in Fig. 4(a). From Fig. 4(a), we can tell that a linear function form for F, $x_{k+1} = x_k + w_k$, is reasonable. To illustrate whether w_k follows a normal distribution with a fixed variance Q, we make a quantile–quantile plot (Q–Q plot) between a normal distribution and $(x_{k+1} - x_k)$, as illustrated in Fig. 4(b). As seen in Fig. 4(b), most of data points fall almost perfectly along the line, which is a good indicator that our w_k is normally distributed.

We further use maximum likelihood estimation (MLE) [34] to estimate $\sqrt{Q}$. Using $N(\mu, \sigma^2) \sim X$ to denote a random process X that is normally distributed with mean μ and variance σ^2, we have $N(0, Q) \equiv N(0, \sigma^2) \sim w_k = x_{k+1} - x_k$, the likelihood function becomes

$$\prod_{k=1}^{n} p(w_k|0, \sigma^2) = \frac{1}{\sqrt{(2\pi\sigma^2)^n}} \exp(-\frac{1}{2\sigma^2} \sum_{k=1}^{n} {w_k}^2)$$
$$= \frac{1}{\sqrt{(2\pi\sigma^2)^n}} \exp(-\frac{1}{2\sigma^2} \sum_{k=1}^{n} (x_{k+1} - x_k)^2)$$

Algorithm 1 EKF based local detection algorithm

Assumption Node A can overhear node B's transmission. A thinks that B is a normal node at and before time t_k
Input z_{k+1} transmitted by node B and overheard by node A
Output Whether A raises an alert on z_{k+1}
Procedure

1: At time t_k, A computes $\hat{x}_k^+$ based on (6) (note that $\hat{x}_k^-$ is stored in node A);
2: A computes $\hat{x}_{k+1}^-$ based on $\hat{x}_k^+$ using (3);
3: A computes $Diff = |\hat{x}_{k+1}^- - z_{k+1}|$;
4: **if** $(\Delta < Diff)$ **then**
5: A raises an alert on B;
6: **else**
7: A thinks that B functions normally;
8: **end if**

where n is the number of readings that we have. Therefore, the MLE estimate of Q is $\hat{Q} = \frac{1}{n-1} \sum_{k=1}^{n-1} (x_{k+1} - x_k)^2$. Applying this to the selected Intel Laboratory data, we can estimate that w_k follows a $N(0, 0.2809)$ distribution.

5) *Threshold Based Anomaly Detection Mechanisms:* Now, we present our EKF based local detection algorithm. A sensor node monitors its neighbor's behavior and establishes a normal range of the neighbor's future aggregated values. The creation of the normal range is centered on estimated values using EKF. An alert can be raised if the monitored value lies outside of the predicted normal range. This scheme is illustrated in Algorithm 1. Here Δ is a predefined threshold.

In Algorithm 1, A's role is to decide whether z_{k+1} is abnormal or not. Node A can overhear node B's transmission z_{k+1} at time t_{k+1}. After estimating $\hat{x}_k^+$ at time t_k, A can predict node B's transmitted value $\hat{x}_{k+1}^-$ at time t_{k+1} based on (3). At time t_{k+1}, A overhears B's transmitted value z_{k+1} and compares $\hat{x}_{k+1}^-$ with z_{k+1} to decide whether B is acting normally or not. If the difference between $\hat{x}_{k+1}^-$ and z_{k+1} (denoted as $Diff$ in Algorithm 1) is larger than Δ, a predefined threshold, A then raises an alert on B. Otherwise, A thinks that B functions

normally. Apparently, Δ is a very important parameter here. We will provide the analysis of Δ in Section V-B6.

In practice, anomaly based IDSs suffer from a high false positive rate. We can use a post-processing scheme to reduce potential false alarms. For example, we can modify Algorithm 1 at line 4 so that A can raise an alert on B after several continuous observations of $\Delta < Diff$. The intuition here is that intrusion sessions usually demonstrate locality, i.e., many alarms within a short time window. In this way, many alerts can be used to generate one intrusion report and false alarms can be effectively reduced.

6) *Threshold Analysis:* In WSNs, various factors, such as packet loss, packet collision, time asynchrony, in aggregation protocols may contribute to uncertainties of aggregated values. Let U denote the variance of this uncertainty. Based on three-sigma control limits in Shewchart control charts [26], Δ can be set to $3\sqrt{U}$. We provide the analysis of U in the following.

In Fig. 1(b), each N_i represents one sensor node and each N_i transmits value v_i to its parent node N based on a predefined aggregation protocol. Suppose that the expectation of each v_i is $E[v_i] = \mu_i$ and the variance of each v_i is $var(v_i) = {\sigma_i}^2$. Suppose that with a probability $0 < p < 1$ (p is the probability that N does not receive the packet from its child because of packet loss, packet collision, etc.), a packet on each link is lost. Let a random variable X denote the aggregated value at node N. We analyze the variance of X considering different packet loss probabilities.

a) *Average:* Let E_m denote an event that there are m ($m >= 0 \wedge m < n$) packets lost. Let V_m denote the corresponding aggregated value when these m packets are lost. Let $p(E_m)$ denote the probability of the event E_m. $\sum p(E_m)V_m = \frac{(n-1)!}{m!(n-m)!}(1-p)^{n-m}p^m\sum_{i=1}^{n} v_i$.

Then, the probability function of X, P_X, is

$$P_X = \begin{cases} (1-p)^n, & \text{if } X = \frac{\sum_{i=1}^{n} v_i}{n} \\ (1-p)^{n-1}p, & \text{if } \frac{\sum_{i=1 \wedge i \neq 1}^{n} v_i}{n-1} \\ (1-p)^{n-1}p, & \text{if } \frac{\sum_{i=1 \wedge i \neq 2}^{n} v_i}{n-1} \\ \cdots & \cdots \end{cases}$$

To save space, we omit the deduction process. For applications where monitored values v_i are similar when aggregation operations are performed (for example, the monitored temperature over an area does not have much difference), denote $E(v_i) = \mu$, and $var(v_i) = \sigma^2$. Therefore, considering the impact of packet loss, collision, etc., $E(X) = \sum_{m=0}^{n}(1-p)^{n-m}p^m\frac{n!}{m!(n-m)!}\mu$.

$$E(X^2) = \sum_{m=0}^{n}[\frac{(1-p)^{n-m}p^m}{(n-m)^2}((C_n^m - C_{n-1}^{m-1})n(\mu^2+\sigma^2) + 2(C_n^m - 2C_{n-1}^{m-1} + C_{n-2}^{m-2})\frac{n(n-1)\mu^2}{2})],$$

where $C_n^m = \frac{n!}{m!(n-m)!}$. We omit the deduction procedure because of the space limitation. We have $var(X) = E[X^2] - E^2[X]$. U can be set to $\sqrt{var(X)}$.

b) *Sum:* For the aggregation sum, when there are m ($m >= 0 \wedge m <= n$) packets lost, we have $\sum p(E_m)V_m = \frac{(n-1)!}{m!(n-m-1)!}(1-p)^{n-m}p^m\sum_{i=1}^{n} v_i$.

Then P_X is

$$P_X = \begin{cases} (1-p)^n, & \text{if } X = \sum_{i=1}^{n} v_i \\ (1-p)^{n-1}p, & \text{if } X = \sum_{i=1 \wedge i \neq 1}^{n} v_i \\ (1-p)^{n-1}p, & \text{if } X = \sum_{i=1 \wedge i \neq 2}^{n} v_i \\ \cdots & \cdots \end{cases}$$

For applications where the monitored values v_i are similar,

$$E(X) = \sum_{m=0}^{n}(1-p)^{n-m}p^m\frac{n!}{m!(n-m-1)!}\mu.$$

$$E(X^2) = \sum_{m=0}^{n}[(1-p)^{n-m}p^m((C_n^m - C_{n-1}^{m-1})n(\mu^2+\sigma^2) + 2(C_n^m - 2C_{n-1}^{m-1} + C_{n-2}^{m-2})\frac{n(n-1)\mu^2}{2})].$$

U can then be computed.

c) *Min/Max:* The analysis of aggregation max is similar to that for the min. To save space, we only provide the analysis of min. For the set of values v_i ($1 \le i \le n$), without loss of generality, after sorting these v_i, we have $v_1 \le v_2 \le \ldots \le v_n$. Then X is defined as follows.

Then, P_X is

$$P_X = \begin{cases} (1-p), & \text{if } X = v_1 \\ p(1-p), & \text{if } X = v_2 \\ p^2(1-p), & \text{if } X = v_3 \\ \cdots & \cdots \end{cases}$$

For applications where v_i are similar, assume that the probability density function (pdf) for v_i is $f(x)$ and the cumulative distribution function for v_i is $F(x)$. Based on order statistics [35], we can compute the pdf of X

$$f_X = \frac{n!}{(r-1)!(n-r)!}[F(x)]^{r-1}[1-F(x)]^{n-r}f(x) = n[1-F(x)]^{n-1}f(x) \quad (8)$$

where r denotes the rth order statistic. For min aggregation, $r = 1$. Based on (8), we can further compute $E(X)$ and $E(X^2)$. U can be derived.

C. CUSUM GLR Based Local Detection

An EKF based approach illustrated in Section V-B does not consider the fact that attacks launched at different times are not always independent. Therefore, an EKF based approach ignores the information given by the entire sequence of measured values. For example, in Algorithm 1, if an attacker continuously forges z_{k+1} with small deviations, this leads to a small $Diff$. A relatively large Δ can make an EKF based approach insensitive to these kinds of attacks because this approach only

TABLE II
NOTATIONS FOR CUSUM GLR

Symbol	Meaning
θ_0	Parameter before change
θ_1	Parameter after change
$p_\theta(y)$	Parameterized probability density - θ: parameter
y_k	Observation at time t_k
ζ_k	Estimate error of EKF at time t_k
ξ_k	Anomaly incurred at time t_k, incurred by attacks
w	Window size used to estimate parameters

uses the information available at a previous time instant. When injected falsified values have small deviations, an EKF based approach alone may not achieve desirable performance.

Therefore, in this section, based on EKF, we further apply an algorithm of combining CUSUM and GLR [1], which utilizes the cumulative sum of the deviations between measured values and estimated values. Table II lists notations that we use.

1) *Basic Principles of CUSUM GLR:* Consider a sequence of observed random variables $y_0, y_1, \ldots, y_k$ with a probability density $p_\theta(y)$ depending on only one scalar parameter θ. Before the unknown change time t_0, θ is constant and equal to θ_0. After the change at t_0, we have $\theta = \theta_1$. To detect the change of θ, we form the following hypothesis about the parameter θ:

$$H_0 : \theta = \theta_0$$
$$H_1 : \theta = \theta_1.$$

If H_0 is accepted based on predefined decision rules, we can think of that θ has no change.

To form decision rules to detect the change of θ, we apply CUSUM GLR because it has illustrated overall desirable performance [32]. We first define the log-likelihood ratio as

$$s_k = \ln \frac{p_{\theta_1}(y_k)}{p_{\theta_0}(y_k)}. \tag{9}$$

Intuitively, s_k shifts from a negative value to a positive one when a change occurs in parameter θ. We further define

$$S_N = \sum_{i=0}^{N} s_i. \tag{10}$$

Typically, S_N first decreases with N, and then begins to increase after θ is changed from θ_0 to θ_1. Because S_N incorporates the cumulative sum of s_k, S_N can be used to detect the change in y_k [1].

2) *Combination of Extended Kalman Filtering and CUSUM GLR:* We define a random process $y_k = z_k - \hat{x}_k^-$, where z_k and $\hat{x}_k^-$ are denoted in Table I. Intuitively, y_k consists of two parts: ζ_k and ξ_k. ζ_k represents EKF estimate errors when there is no anomaly happening. ξ_k represents the anomaly incurred at time t_k. Therefore, in another form, we have $y_k = \zeta_k + \xi_k$.

Based on Section V-B, P_k^+ denotes *a posteriori* estimate error of EKF at time t_k. Therefore, we assume that the sequence ζ_k forms a Gaussian process with mean 0 and variance P_k^+. If there is no attack incurred to y_k, ξ_k is 0. $y_k = \zeta_k$ can be modeled as a zero-mean Gaussian process.

That is, before the incurred anomaly, the mean of the random process y_k (i.e., ζ_k), denoted as μ_0, should center around 0. After the incurred anomaly, the mean of y_k, denoted as μ_1, becomes an unknown value that is decided by attack intensity.

Considering all these into (9), if we use θ_0 and θ_1 to represent μ_0 and μ_1, respectively, we have $\theta_0 \approx 0$. We also need to further estimate θ_1 (i.e., μ_1).

We use values over the past window of size w to estimate μ_1. Denote the list of values over the past window of size w as $y_{k-w+1}, y_{k-w+2}, \ldots, y_k$, where $k \geq w - 1$. Therefore, μ_1 can be estimated as follows:

$$\hat{\mu}_1 = \frac{1}{w} \sum_{i=k-w+1}^{k} y_i. \tag{11}$$

Plugging (11) into (9), we have

$$s_k = \ln \frac{p_{\mu_1}(y_k)}{p_{\mu_0}(y_k)} = \ln \frac{p_{\mu_1}(z_k - \hat{x}_k^-)}{p_{\mu_0}(z_k - \hat{x}_k^-)}.$$

Before the change, s_k is a value roughly close to 0 because θ_1 and θ_0 actually represent the same parameter. After the change, s_k becomes a positive value. Correspondingly, S_N should center around 0 before the change, and shows a positive shift after the change.

Based on this, the decision rule is then given by

$$d = \begin{cases} H_0, & \text{if} \quad S_N < h \\ H_1, & \text{if} \quad S_N \geq h \end{cases} \tag{12}$$

where h is a predefined threshold. That is, when $S_N \geq h$, the decision is in favor of H_1 and anomalies are incurred into y_k. Therefore, an alarm can be raised.

The length of time that it can take to generate alarms depends on attack intensity. The more intense the attack is, the more quickly S_N can reach the predefined threshold h to generate alarms, as we show in Section VI.

3) *CUSUM GLR Based Anomaly Detection:* Due to resource constraints on sensor nodes, it is difficult for sensor nodes to carry out complex operations such as *ln* in (9). Also, it consumes much memory to store $p_\theta(y)$ in sensor nodes. Therefore, necessary simplifications are needed.

We assume that the standard variation of y_k before the anomaly, σ_0, and the standard variation of y_k after the anomaly, σ_1, are roughly the same. Note that if σ_1 is much larger than σ_0, Algorithm 1 can be effective in detecting the anomaly incurred by ξ_k. Therefore, the standard variations of y_k before and after anomaly are roughly the same and thus can be represented by one parameter, denoted as σ. Further assuming that y_k follows a normal distribution, we have $p_{\theta 0}(y_k) \sim N(\mu_0, {\sigma_0}^2) \sim N(0, \sigma^2)$ and $p_{\theta 1}(y_k) \sim N(\mu_1, {\sigma_1}^2) \sim N(\mu_1, \sigma^2)$. Because both $p_{\theta 0}(y_k)$ and $p_{\theta 1}(y_k)$ have the same parameter σ, which can be decided based on specific applications in an offline manner, we can use μ_0 and μ_1 to represent θ_0 and θ_1, respectively. Therefore, we have

$$p_{\theta_i}(y_k) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y_k - \mu_i)^2}{2\sigma^2}} \quad i = 0 \text{ or } 1. \tag{13}$$

Plugging this into (9) and (10), we have

$$S_N = \frac{b}{\sigma} \sum_{k=0}^{N} (y_k - \mu_0 - v/2) = \frac{b}{\sigma} \sum_{k=0}^{N} (y_k - v/2)$$

Algorithm 2 CUSUM GLR based local detection algorithm

Assumption Node A can overhear node B's transmission. A thinks that B is a normal node at and before time t_k

Input A sequence of z_{k+1} transmitted by node B and overheard by node A

Output Whether node A raises an alert on z_k

Procedure

1: Compute $y_k = z_k - \hat{x}_k^-$ at time t_k
2: Compute $\hat{\mu}_1 = \frac{1}{w}\sum_{i=k-w+1}^{k} y_i$ when $k \geq w-1$
3: Compute $S_N = \frac{b}{\sigma}\sum_{i=0}^{N}(y_i - \mu_0 - v/2) = \frac{b}{\sigma}\sum_{i=0}^{N}(y_i - v/2)$
4: **if** $(S_N > h)$ **then**
5: A raises an alert on B;
6: **else**
7: A thinks that B functions normally;
8: **end if**

where $v = \mu_1 - \mu_0 = \mu_1$ and $b = \frac{\mu_1 - \mu_0}{\sigma} = \frac{\mu_1}{\sigma}$. In practice, we can consider the sum of those s_k that fall in any arbitrary window $\sum_{k=N_1}^{N_2} s_k$, where $N_1 \leq k \leq N_2$, to check whether there are any anomaly incurred between N_1 and N_2.

In this way, we can simplify the computation of s_k. Summarizing all these, the CUSUM GLR based detection scheme is illustrated in Algorithm 2. In Algorithm 2, we assume that node A overhears a sequence of z_{k+1} transmitted by node B. Following the above-mentioned approach, node A computes S_N and compares S_N with the threshold value h to decide whether to raise an alert or not.

Here, h is a predefined threshold. By selecting a tradeoff between false positive rate and detection sensitivity, h can be empirically determined in an offline training manner. How to systematically analyze the value of h based on environmental dynamics will be our important future work.

D. Collaboration Between IDM and SMM

Local detection alone is not enough. WSNs are often deployed to monitor emergency phenomena (like the breakout of a forest fire), about which good nodes can trigger important events and generate unusual yet important information. Also, the error prone nature of sensor nodes may make even normal sensor nodes faulty and generate abnormal information. Therefore, local detection alone suffers from a high false positive rate. Node collaboration is necessary for sensor networks to make correct decisions about abnormal events.

Therefore, for WSNs, IDM and SMM need to integrate with each other to work effectively. When node A raises an alert on node B because of some event E, to decide whether E is malicious or emergent, A may initiate a further investigation on E by collaborating with existing SMMs. WSNs are usually densely deployed to collaboratively monitor some events. To save energy, some sensor nodes are periodically scheduled to sleep. Based on this, node A can wake up those sensor nodes (denoted as codetectors in Fig. 5) around B and request from these nodes their opinions on the behavior of E. Because the majority of sensor nodes around the investigated event E are not compromised, after A collects the information from these nodes, if A finds that the majority of sensor nodes think that event E may happen, A then makes a decision that E is triggered by some emergency events.

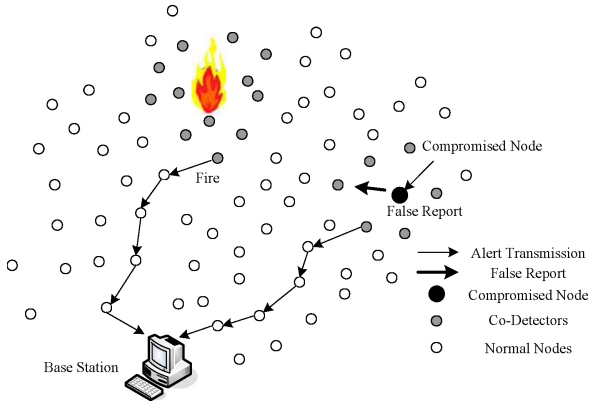


Fig. 5. Collaboration between IDM and SMM to differentiate malicious events from emergency events.

On the other hand, if A finds that the majority of sensor nodes think that event E should not happen, A then thinks that E is triggered by either a malicious node or a faulty yet good node. In this way, A can continue to wake up those nodes around event E and their opinions about the behavior of E. If A keeps finding that the majority of sensor nodes think that event E should not happen, A then suspects that E is malicious.

After A makes a final decision, A can report this event to base stations. No matter whether it is an emergency event or a malicious event, the event can be taken care of by human operators.

In practice, there may exist efficient approaches for SMM to collect information from those sensor nodes around event E. For example, Wang *et al.* [22] proposed an efficient approach to construct a dominating tree to cover all the neighbors of a suspect (node B in our example). Their approach includes those nodes that have more neighbor codetectors (nodes that can provide useful information). By doing so, an efficient dominating tree can be constructed and utilized for an initiator (node A in our example) to collect information about the suspect.

Krontiris *et al.* [42] also proposed a voting based mechanism for collaborative intrusion detection in wireless sensor networks. In [42], each node is equipped with a local detector module. A general algorithm consisting of *Initialization Phase*, *Voting Phase*, *Publish Key Phase*, *Exposing the Attacker*, and *External Ring Reinforcement Phase* is proposed to incorporate local alarms. In our future work, we plan to integrate our EKF based location module with this general algorithm, to make our system resilient to more general attacks.

VI. Performance Evaluation

In this section, we use live data and synthetic data to evaluate EKF based and CUSUM GLR based location detection algorithms. The advantage of live data is that they capture real-world situations. However, live data only contain a limited number of situations whose parameters cannot be varied. Furthermore, it may be difficult to obtain real attack data. In this situation, synthetic data, whose parameters under

normal and abnormal situations can be carefully controlled, can offer advantages.

A. Simulation Results on Synthetic Data

1) *Simulation Setup:* We use Fig. 1(b) as an abstract network model to evaluate EKF based and CUSUM GLR based anomaly detection algorithms. For each link, we use different packet loss rates (denoted as P), 0.1, 0.25, and 0.5, respectively. For each packet loss rate, we simulate two sets of sensor values v_i. v_i denotes raw sensor data transmitted from a child sensor node to its parent node. In the first set, we make all v_i randomly distributed between one predefined range $[min, max]$. We select $[min, max]$ so that a reasonable false positive and detection rate can be plotted. This is to simulate WSN applications that are deployed to monitor an area that has same attributes (for example, temperature). In the second set, we set different v_i randomly distributed between different $[min, max]$ pairs. That is, $\forall i, 1 \leq i \leq n$, $[min, max]$ pairs satisfy $min_i < v_i < max_i$, $min_{i+1} = min_i + T$, and $max_{i+1} = max_i + T$. T is a constant parameter. The purpose is to simulate those WSN applications that monitor an area that has different attributes. For example, in practice, because of the impact of geographic conditions, different areas may have different temperatures.

We set the number of children to 6. This is a reasonable number given a densely deployed sensor network. Furthermore, we do not observe a big impact of the number of children on performance results. We use a normal distribution to generate process noise and measurement noise. For each set of parameters, we measure its performance under different aggregation functions.

We set node N as a compromised node. That is, node N can inject falsified aggregated data into a network. One question is how to simulate attack data. Intuitively, the more different attack data are from normal data, the easier attack activities can be detected. Therefore, we introduce a concept degree of damage, denoted as D. D is defined as the difference between attack data and normal data. For example, in Fig. 1(b), if we assume that the correct aggregated value by node N is C and the malicious aggregated value sent out by N is M, then we have $D = M - C$. We evaluate our local detection schemes using different D.

We use the following two metrics to evaluate EKF based Algorithm 1.

1) False positive rate: It is measured over normal data items. Suppose that m normal data items are measured, and n of them are identified as abnormal. False positive rate is defined as n/m.
2) Detection rate: It is measured over abnormal data items. Suppose that m abnormal data items are measured, and n of them are detected. Detection rate is defined as n/m.

Under the same set of simulation parameters, we obtain 5000 normal data items and 5000 malicious data items. For these data items, we use different threshold values and measure the corresponding false positive rate and detection rate. In other words, for a given threshold value, we use the local detection algorithm presented in Algorithm 1 to obtain one false positive rate and detection rate. We then apply different threshold values and obtain a set of false positive rates and detection rates. Based on them, we then plot receive operating characteristic (ROC) curves. Therefore, under the same set of simulation setting, ROC curve can show detection rates against false positive rates by changing different threshold values. For the same aggregation function, we apply the same set of threshold values for the purpose of comparison.

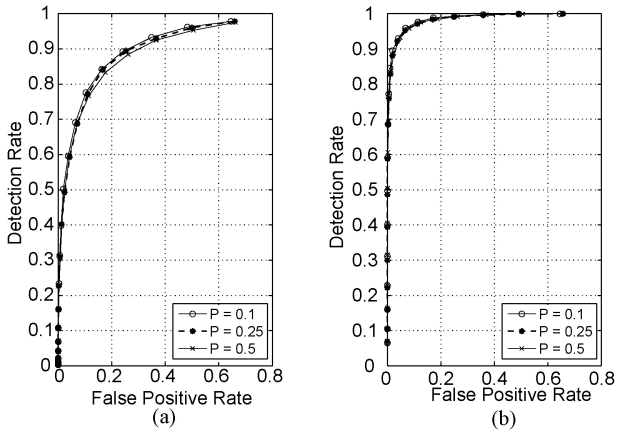


Fig. 6. Same distribution of v_i, average aggregation. (a) D = 10. (b) D = 15.

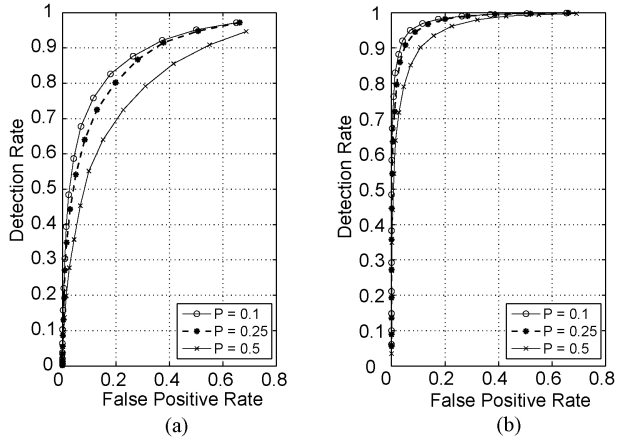


Fig. 7. Different distributions of v_i, average aggregation. (a) D = 10. (b) D = 15.

To evaluate CUSUM GLR based Algorithm 2, we plot S_N based on (10) and s_k based on (9) under normal operations and abnormal operations. Ideally, under normal operations, S_N should center around 0. Under abnormal operations, based on different degrees of damage, S_N should illustrate different trends of changes.

2) *Simulation Results for EKF Based Detection Scheme:* When we evaluate the EKF based detection scheme, in the case of the same distribution of v_i, we make all v_i randomly distributed between one predefined range $[min, max]$. In the case of the different distribution of v_i, we set different v_i randomly distributed between different $[min, max]$ pairs. Since the simulation results of average, sum, maximum, and minimum are similar, we only illustrate the simulation of the average aggregation below I.

a) *Average:* In Fig. 6(a) and (b), we can see that performance (the false positive rate and the detection rate) is not impacted much by packet loss rate. This is because the

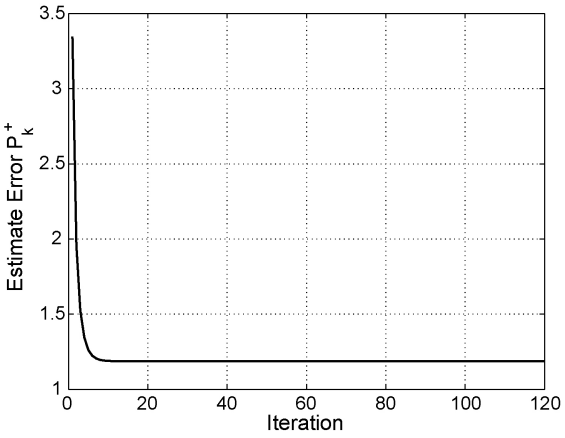


Fig. 8. $P_k{}^+$ stabilize.

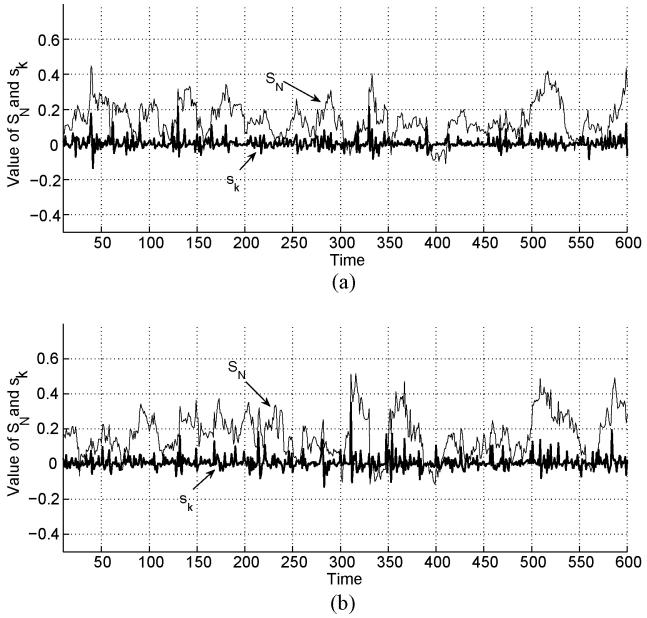


Fig. 9. Same distribution of v_i, average aggregation. (a) P = 0.1. (b) P = 0.5.

aggregation function is average, in which the aggregation node N computes the average of received values. Because all the v_i fall in the same range, the loss of several v_i does not have a big impact on overall performance.

Comparing Fig. 6(b) with (a), we can see that with the increase of the degree of damage in the attack data, the performance improves (the curves move upper left). This is what we have expected.

In Fig. 7(a) and (b), we can observe the impact of different packet loss rates: when the packet loss rate becomes larger, the detection rate becomes smaller, while the false positive rate keeps roughly the same value. This is because with the increase of packet loss rate, the simultaneous loss of smaller v_i increases. This leads to an increase of the measurement value z_k at time t_k. Based on (6), $\hat{x}_k^+$ becomes larger. $\hat{x}_{k+1}^-$, computed based on (3), becomes larger under our simulated environment. Based on Algorithm 1, *Diff* becomes smaller. This leads to a decreased detection rate. Similarly, we can see that Fig. 7(b) has a better overall performance than Fig. 7(a).

b) *Estimate Error:* We use a constant Q and R in our simulation. Because of this, the estimate error P_k^+ stabilizes

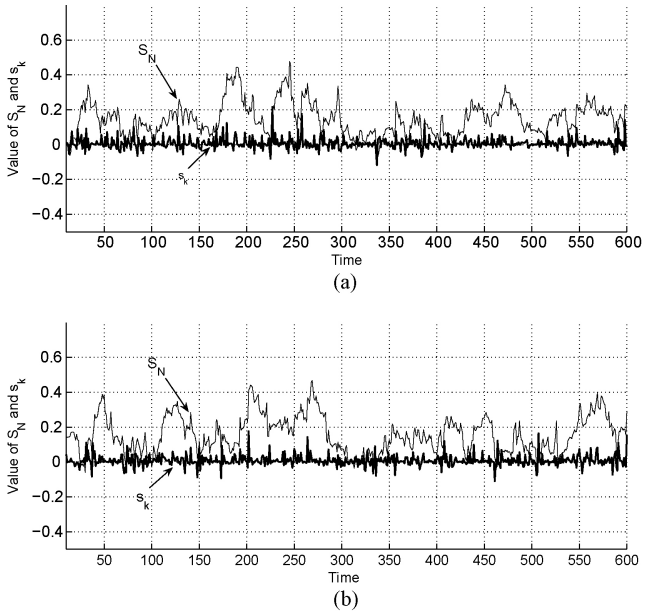


Fig. 10. Different distributions of v_i, average aggregation. (a) P = 0.1. (b) P = 0.5.

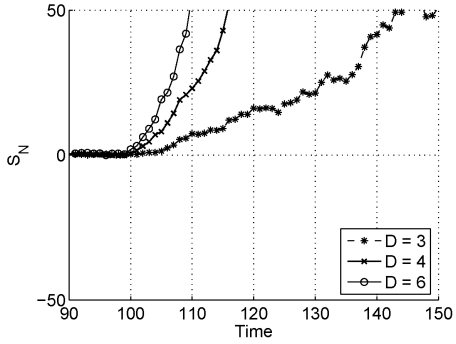


Fig. 11. Detection sensitivity of average aggregation.

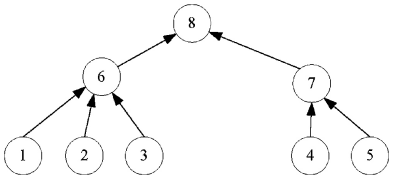


Fig. 12. Testbed.

quickly. Therefore, the initial value of $P_k{}^+$ does not matter very much. Fig. 8(a) illustrates one example run of P_k^+ with an arbitrary initial value.

3) *Simulation Results for CUSUM GLR Based Detection Scheme:* We have similar simulation results and observations between the *average* aggregation and other aggregation functions, such as *sum*, *min*, and *max*. Therefore, we only present the results of the average function in the following.

For the same distribution of v_i under normal operations, the change of S_N and s_k for average aggregations under different packet loss rates is plotted in Fig. 9(a) and (b), respectively. As we expect, because there is no anomaly incurred into y_k, s_k are close to zeros most of the time. Therefore, S_N, based on its definition in (10), are also very close to zeros most of the time.

Fig. 9(a) and (b) can help us to select a proper h in Algorithm 2 and derive a corresponding false positive rate.

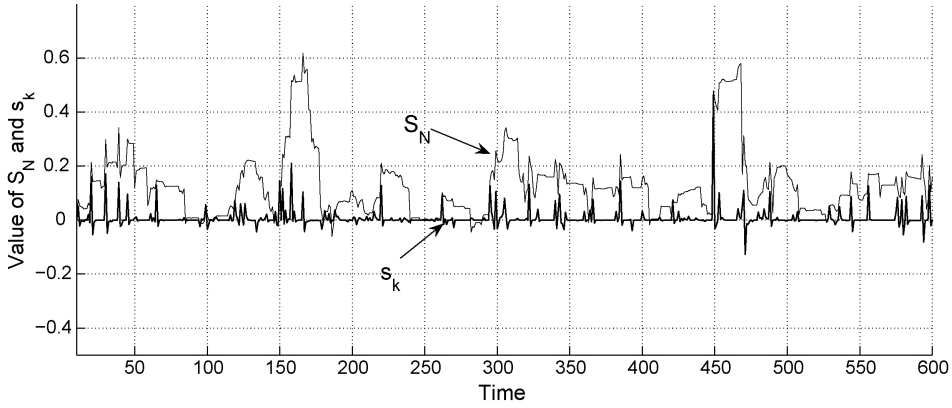


Fig. 13. S_N and s_k for average aggregation—measured in a lab setting.

TABLE III
ROM AND RAM SIZE FOR IMPLEMENTATION

	ROM Size (Bytes)	RAM Size (Bytes)
EKF	1204	32
CUSUM GLR	1888	616

For example, in Fig. 9(a), because most S_N is less than 0.4, a proper h can be set to 0.4.

For the different distribution of v_i under normal operations, the change of S_N and s_k for average aggregations under different packet loss rates is plotted in Fig. 10(a) and (b), respectively. First, for Fig. 10(a) and (b), we have the observations similar to that of Fig. 9(a) and (b). Second, when we compare Fig. 10(a) with 9(a), we do not observe the impact of different v_i on measured performance. This is because in Algorithm 2, σ of different v_i is larger than that of same v_i. Therefore, although different v_i lead to more dynamics, we do not observe a more dynamic S_N and s_k. The same is true when we compare Fig. 10(b) with 9(b).

We then introduce attack data into our simulation, and plot S_N based on different attack intensities. The result is illustrated in Fig. 11. Clearly, more intensive attacks lead to a faster increase of S_N, and thus faster detection of attacks. This is what we expect.

B. Experimental Results on Live Data

We implement EKF and CUSUM GLR on MICA2 motes [23] in TinyOS and evaluate their overhead in terms of ROM size and RAM size. The result is illustrated in Table III.

For one MICA2 mote, EKF needs 1204 Bytes in ROM and 32 Bytes in RAM. CUSUM GLR needs 1888 Bytes in ROM and 616 Bytes in RAM. The reason that CUSUM GLR needs relatively a large RAM is because CUSUM GLR needs to estimate μ_1 based on (11).

We use 8 MICA2 sensor nodes to set up a two-level testbed, as illustrated in Fig. 12. In a lab with roughly constant temperature, leaf nodes (nodes 1–5) take samples of temperature values at a rate of one sample per minute and periodically transmit sensed values to parent nodes (nodes 6 and 7). The parent nodes compute the average of received values and transmit the aggregated data to node 8, which further aggregates the received data.

Using data collected at node 8, we plot the dynamics of S_N and s_k, as illustrated in Fig. 13. We have the same observation as that of average aggregations illustrated in Section VI-A3. For other aggregation functions, we also have the same observations as that illustrated in Section VI-A3. To save space, we omit the introduction of their experimental results.

VII. CONCLUSION AND FUTURE WORK

In this paper, we first proposed that IDM and SMM should work together to provide intrusion detection capabilities for WSNs. We then proposed an EKF based approach to detect false injected data. We illustrated how we use EKF to address various uncertainties in WSNs and created an effective local detection mechanism. To increase detection sensitivity, we further applied an algorithm of combining CUSUM and GLR. We further demonstrated how our proposed IDM can work together with SMM to differentiate between malicious events and emergency events.

We evaluated our proposed schemes using both real experiments based on MICA2 motes and large-scale synthetic data. Experiment results demonstrate that our proposed work is suitable to provide intrusion detection capabilities for secure in-network aggregation in wireless sensor networks.

In the future, we will address more aspects of our proposed system to make it more robust and effective. This includes considering more parameters in the EKF and CUSUM GLR based local detection, how to identify the adversary when the majority of its neighbors have been compromised, and more systematic evaluation of our proposed solutions.

REFERENCES

[1] M. Basseville and I. V. Nikiforov, *Detection of Abrupt Changes: Theory and Application*. Englewood Cliffs, NJ: Prentice-Hall/Simon and Schuster Company, 1993.

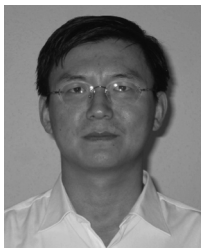
[2] D. Wagner, "Resilient aggregation in sensor networks," in *Proc. ACM SASN*, 2004, pp. 78–87.

[3] S. Zhu, S. Setia, S. Jajodia, and P. Ning, "An interleaved hop-by-hop authentication scheme for filtering false data injection in sensor networks," in *Proc. IEEE Symp. Security Privacy*, May 2004, pp. 260–272.

[4] C. Castelluccia, E. Mykletun, and G. Tsudik, "Efficient aggregation of encrypted data in wireless sensor networks," in *Proc. MOBIQUITOUS*, Jul. 2005, pp. 109–117.

[5] H. Cam, S. Ozdemir, P. Nair, and D. Muthuavinashiappan, "Espda: Energy efficient and secure pattern-based data aggregation for wireless sensor networks," in *Proc. IEEE Sensors*, Oct. 2003, pp. 732–736.

[6] B. Przydatek, D. Song, and A. Perrig, "SIA: Secure information aggregation in sensor networks," in *Proc. ACM Sensys*, 2003, pp. 255–265.
[7] L. Hu and D. Evans, "Secure aggregation for wireless networks," in *Proc. Workshop Security Assurance Ad Hoc Netw.*, Jan. 2003, pp. 384–391.
[8] Y. Yang, X. Wang, S. Zhu, and G. Cao, "SDAP: A secure hop-by-hop data aggregation protocol for sensor networks," in *Proc. ACM MOBIHOC*, May 2006, pp. 356–367.
[9] K. Wu, D. Dreef, B. Sun, and Y. Xiao, "Secure data aggregation without persistent cryptographic operations in wireless sensor networks," *Elsevier Ad Hoc Networks J.*, vol. 15, no. 1, pp. 100–111, 2007.
[10] I. Solis and K. Obraczka, "In-network aggregation trade-offs for data collection in wireless sensor networks," *Int. J. Sensor Netw.*, vol. 1, nos. 3–4, pp. 200–212, Sep. 2006.
[11] N. Shrivastava, C. Buragohain, D. Agrawal, and S. Suri, "Medians and beyond: New aggregation techniques for sensor networks," in *Proc. ACM Sensys*, 2004, pp. 239–249.
[12] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of network density on data aggregation in wireless sensor networks," in *Proc. ICDCS*, 2002, pp. 457–458.
[13] S. Madden, M. J. Franklin, J. Hellerstein, and W. Hong, "TAG: A tiny aggregation service for ad-hoc sensor networks," in *Proc. OSDI*, Dec. 2002, pp. 131–146.
[14] J.-Y. Chen, G. Pandurangan, and D. Xu, "Robust computation of aggregates in wireless sensor networks: Distributed randomized algorithms and analysis," *IEEE Trans. Parallel Distributed Syst.*, vol. 17, no. 9, pp. 987–1000, Sep. 2006.
[15] A. Manjhi, S. Nath, and P. B. Gibbons, "Tributaries and deltas: Efficient and robust aggregation in sensor network streams," in *Proc. ACM SIGMOD*, 2005, pp. 287–298.
[16] S. Roy, S. Setia, and S. Jajodia, "Attack resilient hierarchical data aggregation in sensor networks," in *Proc. ACM SASN*, 2006, pp. 71–82.
[17] H. Chan, A. Perrig, and D. Song, "Secure hierarchical InNetwork aggregation in sensor networks," in *Proc. ACM CCS*, 2006, pp. 278–287.
[18] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," in *Proc. ACM Sensys*, Nov. 2003, pp. 1–13.
[19] R. Pon, M. Batalin, M. Rahimi, Y. Yu, D. Estrin, G. J. Pottie, M. Srivastava, G. Sukhatme, and W. J. Kaiser, "Self-aware distributed embedded systems," in *Proc. IEEE FTDCS*, May 2004, pp. 102–107.
[20] D. Chu, A. Deshpande, J. M. Hellerstein, and W. Hong, "Approximate data collection in sensor networks using probabilistic models," in *Proc. IEEE ICDE*, Apr. 2006, pp. 48–59.
[21] M. C. Vuran *et al.*, "Spatial-temporal correlation: Theory and applications for wireless sensor networks," *Elsevier Comput. Netw.*, vol. 45, no. 3, pp. 245–259, Jun. 2004.
[22] G. Wang, W. Zhang, G. Cao, and T. La Porta, "On supporting distributed collaboration in sensor networks," in *Proc. IEEE Milcom*, vol. 2. Oct. 2003, pp. 752–757.
[23] *Mica2 Mote* [Online]. Available: http://www.memsic.com
[24] *MTS310 Sensor Board* [Online]. Available: http://www.memsic.com
[25] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice Using MATLAB*. New York: Wiley, Jan. 2001.
[26] D. C. Montgomery, *Introduction to Statistical Quality Control*. New York: Wiley, 2009.
[27] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proc. ACM Mobicom*, Aug. 2000, pp. 255–265.
[28] D. J. Abadi, S. Madden, and W. Lindner, "REED: Robust, efficient filtering and event detection in sensor networks," in *Proc. VLDB*, 2005, pp. 769–780.
[29] Z. Yu and Y. Guan, "A dynamic en-route scheme for filtering false data injection in wireless sensor networks," in *Proc. IEEE INFOCOM*, Apr. 2006, pp. 1–12.
[30] F. Ye, H. Luo, S. Lu, and L. Zhang, "Statistical en-route detection and filtering of injected false data in sensor networks," in *Proc. IEEE INFOCOM*, Mar. 2004, pp. 2446–2457.
[31] W. Zhang and G. Cao, "Group rekeying for filtering false data in sensor networks: A predistribution and local collaboration-based approach," in *Proc. IEEE INFOCOM*, 2005, pp. 503–514.
[32] D. Han and F. Tsung, "A generalized EWMA control chart and its comparison with the optimal EWMA, CUSUM, and GLR schemes," *Ann. Statist.*, vol. 32, no. 1, pp. 316–339, 2004.
[33] *Intel Lab Data* [Online]. Available: http://berkeley.intel-research.net/labdata/
[34] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. New York: Wiley, Oct. 2000.
[35] Wolfram MathWorld. *Order Statistic* [Online]. Available: http://mathworld.wolfram.com/OrderStatistic.html
[36] E. Elnahrawy and B. Nath, "Cleaning and querying noisy sensors," in *Proc. ACM WSNA*, Sep. 2003, pp. 78–87.
[37] B. Krishnamachari and S. Iyengar, "Distributed Bayesian algorithms for fault-tolerant event region detection in wireless sensor networks," *IEEE Trans. Comput.*, vol. 53, no. 3, pp. 241–250, Mar. 2004.
[38] T. Clouqueur and K. Saluja, "Fault tolerance in collaborative sensor networks for target detection," *IEEE Trans. Comput.*, vol. 53, no. 3, pp. 320–333, Mar. 2004.
[39] D. Dong, Y. Liu, and X. Liao, "Self-monitoring for sensor networks," in *Proc. ACM MobiHoc*, May 2008.
[40] J. Lin, L. Xie, and W. Xiao, "Target tracking in wireless sensor networks using compressed KF," *Int. J. Sensor Netw.*, vol. 6, nos. 3–4, Nov. 2009.
[41] K. Premkumar, A. Kumar, and J. Kuri, "Distributed detection and localization of events in large ad hoc wireless sensor networks," in *Proc. 47th Annu. Allerton Conf. Communications Control Comput.*, Sep.–Oct. 2009, pp. 178–185.
[42] I. Krontiris, Z. Benenson, T. Giannetsos, F. C. Freiling, and T. Dimitriou, "Cooperative intrusion detection in wireless sensor networks," in *Proc. 6th EWSN*.



Bo Sun (S'01–M'04) received the Ph.D. degree in computer science from Texas A&M University, College Station, in 2004.

He is currently an Associate Professor with the Department of Computer Science, Lamar University, Beaumont, TX. His current research interests include security issues (intrusion detection in particular) of wireless ad hoc networks, wireless sensor networks, cellular mobile networks, and other communications systems.

Dr. Sun was a recipient of the 2006 Texas Advanced Research Program Grant and the National Science Foundation Grants DUE-0633445 and CNS-0922888 for his research.

Xuemei Shan received the Ph.D. degree in industrial engineering from Northwestern University, Evanston, IL.

She currently works in risk management at a leading financial service company.

Kui Wu (SM'08) received the Ph.D. degree in computing science from the University of Alberta, Edmonton, Canada, in 2002.

He joined the Department of Computer Science, University of Victoria, Victoria, British Columbia, Canada, in 2002, and is currently an Associate Professor there. His current research interests include mobile and wireless networks, network performance evaluation, and network security.

Dr. Wu is a member of the ACM.

Yang Xiao (SM'04) was involved in the industry as a Medium Access Control Architect, engaged in IEEE 802.11 standard enhancement work before he joined academia. He is currently with the Department of Computer Science, University of Alabama, Tuscaloosa, Canada. His current research interests include security and communications or networks. He has published more than 200 refereed journal papers, including 50 IEEE and ACM transactions papers, and over 200 refereed conference papers and book chapters related to these research areas.

Dr. Xiao currently serves as the Editor-in-Chief for the *International Journal of Security and Networks* and the *International Journal of Sensor Networks* (SCI-index).