A Trust Management based Framework for Fault-tolerant Barrier Coverage in Sensor Networks

Shibo He†, Yuanchao Shu†, Xianbin Cui†, Chunjuan Wei‡, Jiming Chen$^{\dagger *}$ and Zhiguo Shi†
† Zhejiang University, Hangzhou, China
‡ Shanghai University of Electric Power, Shanghai, China
Email: {s18he, ycshu, cjm, shizg}@zju.edu.cn; cuixianbin@gmail.com; weichunjuan@shiep.edu.cn

Abstract—**Barrier coverage has been widely adopted to prevent unauthorized invasion of important areas in sensor networks. As sensors are typically placed outdoors, they are susceptible to getting faulty. Previous works assumed that faulty sensors are easy to recognize, e.g., they may stop functioning or output apparently deviant sensory data. In practice, it is, however, extremely difficult to recognize faulty sensors as well as their invalid output. We, in this paper, propose a novel fault-tolerant intrusion detection algorithm (TrusDet) based on trust management to address this challenging issue. TrusDet comprises of three steps: i) sensor-level detection, ii) sink-level decision by collective voting, and iii) trust management and fault determination. In the Step i) and ii), TrusDet divides the surveillance area into a set of fine-grained subareas and exploits temporal and spatial correlation of sensory output among sensors in different subareas to yield a more accurate and robust performance of barrier coverage. In the Step iii), TrusDet builds a trust management based framework to determine the confidence level of sensors being faulty. We implement TrusDet on HC-SR501 infrared sensors and demonstrate that TrusDet has a desired performance.**

I. Introduction

Barrier coverage [1], [2] (a.k.a. intrusion detection [3]) is one of the most important applications of wireless senor networks [4], [5]. It is concerned with the detectability of intruders when they are crossing the border of an important surveillance area. Due to its easy and low-cost deployment as well as efficient intrusion detection, it has been widely employed to prevent the unauthorized invasion in many application scenarios [6], [7].

As we all know, sensors typically are placed outdoor for intrusion detection. They are exposed to sunshine, wind, rain and other environmental forces, and thus are vulnerable to getting fault. Previous works assumed that faulty sensors are easy to recognize, e.g., they may stop functioning or output apparently deviant sensory data [8]. In such case, we could facilely remove the sensory data of faulty sensors and fuse valid sensory data to attain a good detection [4]. Nevertheless, in practice there exist multiple types of fault. A faulty sensor may output intrusion alarm, keep silent (i.e., no intruder is reported) or output results randomly [9]. Further, faulty sensors may have varying types of fault or recover to normal during operation. What is worse, the detection accuracy of normal sensors is dependent of the surrounding changes. Even a normal sensor could report a wrong detection result. It is, thereby, extremely difficult to recognize faulty sensors as well as their invalid output. This makes fault-tolerant intrusion detection a radically new and extremely challenging problem, which can not be solved by existing approach, e.g., simply providing redundant coverage (i.e., multiple sensors can simultaneously cover the same area).

We take the first attempt to address the fault-tolerant intrusion detection by adopting the barrier coverage model. We consider five typical types of fault [10], [11]: constant alarm fault, constant silent fault, shifted output fault, random output fault, and instantaneous output fault (detailed description can be found in Sec. II-B). A faulty sensor could send a measurement of any possible types of faulty data. Without knowledge about which type of fault each sensor is having, it is, therefore, extremely difficult to tell whether a sensor is functioning normally by the output of measurement. The challenge lies in how to dynamically identify the faulty sensors by current and historical collective decisions from multiple sensors and fuse the sensory data from normal sensors to yield a high intrusion detection probability.

We propose a novel fault-tolerant intrusion detection algorithm (TrusDet) based on trust management in this paper. TrusDet is based on the rationale that the sensory data among close sensors exhibit strong temporal and spatial correlation and thus a faulty sensor can be identified if it behaves statistically inconsistently from other normal sensors. It comprises of three components: i) sensor-level detection, ii) sink-level decision by collective voting, and iii) trust management and fault determination. Summarizing, our contributions in this paper are three-fold:

1) We formulate the problem of fault-tolerant barrier coverage in wireless sensor networks, by taking into account multiple types of fault and detection uncertainty.
2) We design a novel trust management based intrusion detection algorithm (TrusDet) to tackle the problem. TrusDet exploits temporal and spatial correlation of measurement data among sensors to attain a high-accuracy and robust detection probability and builds a trust management framework to dynamically update the confidence level of sensors being faulty.
3) We implement TrusDet on HC-SR501 infrared sensor. We validate the coverage model adopted in this work and demonstrate that TrusDet has a very low false alarm rate in practice.

[*] Jiming Chen is the corresponding author.

This research was supported in part by Zhejiang provincial NSF under grant LR16F020001, and NSFC under grant 61402405, 61672458 and 61528105.

The remainder of the paper is organized as follows. We introduce the fault-tolerant barrier coverage in Sec. II. We design a trust management based intrusion detection algorithm (TrusDet) in Sec. III. We implement TrusDet on HC-SR501 infrared sensor and perform experiments to evaluate the performance of TrusDet in Sec. IV. We conclude the paper in Sec. V.

II. Problem Formulation

In this section, we first provide the coverage model and sensor fault model, and then elaborate on the network model.

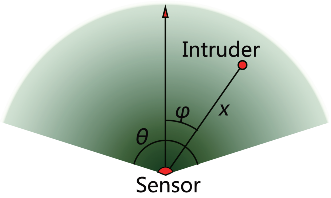


Fig. 1. The coverage model.

A. Coverage Model

We adopt a directional coverage model. Specifically, the sensing region of each sensor i is characterized by a circular sector centered at i of radius R and the central angle θ, as illustrated by Fig. 1. Within the sensing region, the detection p_d is probabilistic since sensors are typically placed outdoor and the sensing accuracy could be impacted by various environmental factors:

$$p_d = \begin{cases} 1, & \text{if } x \leq R', \varphi \leq \frac{\theta}{2} \\ e^{-\lambda x^\beta}, & \text{if } R' < x \leq R, \varphi \leq \frac{\theta}{2} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where λ and β are constants related to the sensor type, R' is the range within which an intruder can be detected deterministically, R is the maximum sensing range, and φ is the orientation angle of the sensor. We note that the directional coverage model is a good approximation to the realistic sensing region of some passive sensors and has been widely employed in the literature [12], [13]. We assume that sensors are homogeneous, i.e., each sensor has the same coverage model. In practice, this can be guaranteed by purchasing the same type of sensors for the same application scenario and adjusting their sensitivity to the same level.

B. Sensor Fault Model

A variety of faults may occur to sensors during operation. Most existing literature assumed that sensors will stop functioning once they get faulty. Based on such assumption, it is easy to recognize the faulty sensors. In practice, faulty sensors may continue to contribute sensory data. For example, if a sensor is partially damaged, it may capture very weak signal strength regardless of the existence of intruders. Consequently, the sensor could not detect any intrusion and will keep silent constantly. To the contrary, if normally-closed switch output is adopted at a sensor [10], the sensor will keep sending alarm message of intrusion when a cable is broken. Common faults occurring to sensors can mainly be categorized into the following five types [10], [11]:

- **Constant alarm fault**: the faulty sensor will output constant alarm message regardless of the existence of intrusion;
- **Constant silent fault**: the faulty sensor will keep silent constantly regardless of the existence of intrusion;
- **Shifted output fault**: the faulty sensor will output message which is opposite to what it captures;
- **Random output fault**: the faulty sensor will output silent and alarm messages randomly regardless of the existence of intrusion;
- **Instantaneous output fault**: the outputs of the faulty sensor are incorrect for a short time, after which it becomes normal.

C. Network Model

We consider that a wireless sensor network, comprising of a set $\mathcal{N} = \{1, 2, \cdots, n\}$ of sensors, is deployed around the perimeter of an important area (such as precious infrastructure or military base) to detect intrusion. For easy presentation, we assume that the perimeter of the important area is a line segment. We evenly distribute n sensors along the straight perimeter so that the distance between every two adjacent sensors along the perimeter is the same. Denote the distance between two adjacent sensors by d. The connectivity of the network can be guaranteed when the communication range is larger than d [14]. Otherwise, we can reduce d to ensure the network connectivity.

The whole operation of the sensor network is divided into time slots. The duration of one slot is denoted by T_0. In each slot, each sensor monitors its surrounding to tell if there is an intruder. It sends an alarm message of intrusion to the sink if the received signal strength exceeds a constant threshold; otherwise, it does nothing, i. e., keep silent. Since detection output by single sensor is unreliable, it is desirable to provide multiple coverage. We begin with formally defining (k, δ)-coverage.

***Definition** 1 ((k,δ)-Coverage):* A point P is said to be (k, δ)-covered if there are at least k sensors, each of which can detect the intruder at P with a detection probability no less than δ. Equivalently, we say the sensor network provides (k, δ)-coverage to point P. An area Ω is (k, δ)-covered if every point in Ω is (k, δ)-covered.

***Definition** 2 (The Detectability of a (k,δ)-covered area):* Given an area Ω, which is (k, δ)-covered, its detectability, denoted by $D(\Omega)$, is defined as the shortest distance for an intruder to cross Ω.

Given k, δ and the application requirement $D(\Omega)$, it is easy to calculate d. The radius is denoted by r_δ when detection probability equals to δ. We have

$$d = \frac{2}{k} \times \frac{-D(\Omega)\tan\frac{\pi-\theta}{2} + \sqrt{r_\delta^2 tan^2\frac{\pi-\theta}{2} + r_\delta^2 - D^2(\Omega)}}{tan^2\frac{\pi-\theta}{2} + 1} \tag{2}$$

We proceed to define the neighbors of each sensor i.

***Definition** 3 (Neighbors):* Given a (k, δ)-covered area Ω, the neighbors of sensor i, denoted by $\mathcal{N}_i$, is the set of sensors which cover the same Ω as sensor i.

The definition of neighbors defined in this paper is quite different from those in literature. After sensor deployment, we have a (k, δ)-covered area with detectability $D(\Omega)$. When an intruder comes, k sensors could possibly report intrusion alarm messages, which could be utilized to yield a robust detection. However, due to the presence of sensor fault, a sensor may output intrusion alarm, keep silent (i.e., no intruder present) or output results randomly. Since the detection by normal sensors is probabilistic, even a normal sensor could send a wrong detection result. A critical problem that comes is how to recognize the faulty sensors and fuse the sensory data to attain a high-accuracy and robust detection. Our goal in this paper is to detect the intrusion and recognize the faulty sensors with maximum accuracy, simultaneously.

III. Fault-tolerant Intrusion Detection

In this section, we design a trust management based intrusion detection algorithm (TrusDet) to solve the fault-tolerant barrier coverage. The main rationale behind TrusDet is that we exploit the temporal and spatial correlation of alarm outputs among sensors to relieve the impact of detection uncertainty and leverage historical cumulative performance to build a trust evaluation framework.

Specifically, TrusDet consists of three components: 1) sensor-level detection, 2) sink-level decision by collective voting, and 3) trust management and fault determination. In the first component, each sensor first collects the signal strength of intruder and obtains an output based on the detection. Then it fuses the decision by using historical detection to get a detection result. In the second component, after collecting all the alarm messages from sensors, the sink exploits the tempo-spatial correlation of alarm messages to make a collective decision, based on the observation that nearby sensors will report similar outputs. In the third component, the cumulative performance of each sensor is evaluated and a trust management framework is built. The faulty sensors can be recognized by the trust value due to their inconsistent behaviors. The whole process can be represented by the Fig. 2.

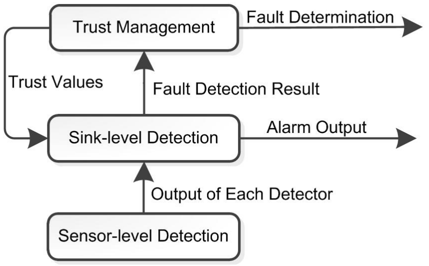


Fig. 2. Schematic diagram of TrusDet.

A. Sensor-level Detection

Since one-time detection may cause false alarm, we introduce sliding window to increase the reliability of intrusion detection. After a sensor i takes a measurement result $c_i(t)$ at slot t, it selects past L slots as a sliding window to obtain a fused result $\overline{c_i(t)}$ [15], i.e., $\overline{c_i(t)} = \sum_{l=0}^{L} \alpha(t - lT_0)c_i(t - lT_0)$, where T_0 is the duration of a time slot, $\sum_{l=0}^{L} \alpha(t - lT_0) = 1$, and $\alpha(t - l_1T_0) \geq \alpha(t - l_2T_0)$ when $l_1 < l_2$, which means a closer previous result has a greater impact on the data fusion.

Given that the output of a detection is binary in general, we choose Th_D as a threshold to determine the binary output $O_i(t)$ of sensor i at slot t, i.e.,

$$O_i(t) = \begin{cases} 1, \overline{c_i(t)} \geq Th_D \\ 0, \overline{c_i(t)} < Th_D \end{cases} \tag{3}$$

where 1 means that there exists an intruder while 0 indicates no intruder present.

In such a scheme, a correct detection result can be diluted by previous detection results in the data fusion. This may result in missing alarm. To this end, the detectability $D(\Omega)$ of the (k, δ)-covered area should be reasonably large such that an intruder should take at least $L+1$ slots to cross the area. That is

$$D(\Omega) \geq (L + 1) \times v_{max}T_0, \tag{4}$$

where v_{max} is the maximum estimated velocity of an intruder.

For the sensors which yield an alarm of intrusion, they send the alarm messages (i.e., digital 1) to the sink with the predefined routing path [16]. Sensors which do not detect the intruder keep silent. This reduces the traffic load since most of sensors keep silent at each slot.

B. Sink-level Decision by Collective Voting

After receiving alarm messages from sensors, the sink tries to make a final decision. Note that if sensor i detects an intruder in slot t, then with high probability its neighbors $\mathcal{N}_i$ also detect the intruder in slot t. Also, its neighbors may detect the intruder in the last slot since the intruder move continuously in the surveillance area. Therefore, there is temporal and spatial correlation among the detection output of sensors, which is exploited to yield a better fusion decision in TrusDet.

Notice that when an intruder is present at the (k, δ)-covered area, k sensors could detect its existence with a probability no less than δ. Clearly, an intruder can not be present at multiple locations simultaneously. Thus, we divide (k, δ)-covered area further into a collection of subareas, so that all points in one subarea are (k, δ)-covered by the same set of sensors. Local data fusion can be performed among sensors which cover the same subarea. In this way, we have a fine-grained collective voting, resulting in a more reliable fusion decision.

Specifically, we redefine the sensing area of each sensor as the area within which an intruder can be detected with a probability larger than or equal to δ. Then the coverage area is divided into several subareas $\mathcal{SR} = \{SR_1, SR_2, \cdots\}$. Obviously, in each subarea, all points are (k, δ)-covered by the same set of sensors. Denote the set of sensors that cover the same subarea SR with probability larger than δ by $\mathcal{N}(SR)$.

To include the case of sensor fault, we differentiate the contribution of each sensor in the same subarea. The contribution is measured by the trust weight $r_i(t)$ of each sensor i in slot t, a higher trust weight implying higher reliability of the sensor's output (trust management will be elaborated in next subsection). Initially, all sensors are taken as normal and assigned with the same trust weight.

Denote the collection of (k, δ)-covered area associated with sensor i by V_i. For each subarea $SR \cap V_i \neq \emptyset$, we calculate the fusion result $O_{i,spa}(SR)$ by collective voting from sensors in $\mathcal{N}(SR)$ as $O_{i,spa}(SR) = \sum\limits_{j \in \mathcal{N}(SR \cap V_i), j \neq i} r_j(t) \times O_j(t)$.

Then the subarea with the largest value has the highest probability of intrusion, i.e.,

$$SR^*_{i,spa} = \arg \max_{SR \cap V_i \neq \emptyset} O_{i,spa}(SR). \tag{5}$$

We proceed to include the temporal correlation of alarm message in the data fusion. We first find all possible subareas $\mathcal{SR}(SR^*_{i,spa})$ from which an intruder can travel to the current subarea $SR^*_{i,spa}$. Since an intruder can travel at most v_{max} in one slot, $\mathcal{SR}(SR^*_{i,spa})$ can be computed as

$$\mathcal{SR}(SR^*_{i,spa}) = \{SR \in \mathcal{SR} : |SR - SR^*_{i,spa}| \leq v_{max} T_0\},$$

where $|SR - SR^*_{i,spa}| = \min_{P_1 \in SR, P2 \in SR^*_{i,spa}} |P_1 - P_2|$ is the distance between two set of points. As illustrated in Fig. 3, the area enclosed by the small red curve is $SR^*_{i,spa}$ and the area enclosed by the large red curve is where the intruder may be in slot $t - T_0$.

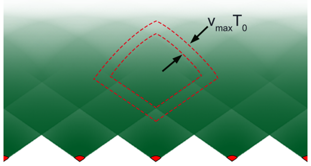


Fig. 3. An illustration of $\mathcal{SR}(SR^*_{i,spa})$.

We then calculate the fusion result $O_{i,tem}(SR)$ from sensors which have temporal correlation with sensor i. For each $SR \in \mathcal{SR}(SR^*_{i,spa})$, we have $O_{i,tem}(SR) = \sum\limits_{j \in \mathcal{N}(SR)} r_j(t - T_0) \times O_j(t - T_0)$.

Therefore, collective voting from sensors of temporal correlation with sensor i can be obtained by

$$SR^*_{i,tem} = \arg \max_{SR \cap V_i \neq \emptyset} O_{i,tem}(SR). \tag{6}$$

Finally, the collective voting can be performed in the following way:

$$\overline{O_i(t)} = O_i(t) + O_{i,spa}(SR^*_{i,spa}) + O_{i,tem}(SR^*_{i,tem}). \tag{7}$$

Denote by $A_i(t)$ the system alarm output, which can now be decided in the following way:

$$A_i(t) = \begin{cases} 1, \overline{O_i(t)} \geq AlarmTh \\ 0, \overline{O_i(t)} < AlarmTh \end{cases} \tag{8}$$

where $AlarmTh$ is the threshold value, $A_i(t) = 1$ means system alarm for intruder presence, and $A_i(t) = 0$ means system silence.

The final decision $A_i(t)$ is then used to evaluate the detection performance of sensors, which have participated in the voting process. For sensors which send alarm messages to the sink, its alarm indicator $I^a_i(t) = 1$ when there exists $A_j(t) = 1$ that matches $i \in \mathcal{N}(SR^*_{j,spa})$, and $I^a_i(t) = 0$, otherwise. For sensors which keep silent, its silence indicator $I^s_i(t) = 0$ when there exists $A_j(t) = 1$ that matches $i \in \mathcal{N}(SR^*_{j,spa})$, and $I^s_i(t) = 1$, otherwise.

When multiple sensors send alarm messages, the sink performs the collective voting for each sensor, and evaluates the detection performance of participating sensors accordingly. If $A_i = 1$ for some sensor and only one subarea is found in the spatial voting process, sink outputs one alarm message to warn that there is one intruder; If there are multiple $A_i = 1$ and more than one subareas are found in the spatial voting process, the sink outputs multiple alarm messages.

C. Trust Management and Fault Determination

With the presence of environmental noise, normal sensors could have incorrect outputs. Taking this into account, we introduce a trust management mechanism in this section, which evaluates the trust values of sensors based on their historical performance [17]. In such a way, normal sensors would not be taken as faulty ones just because they have made a few wrong detections. We also consider the recovery mechanism of trust value so that the trust value of a faulty sensor (e.g., sensors with instantaneous output fault) has the chance to be taken as normal when it becomes normal again.

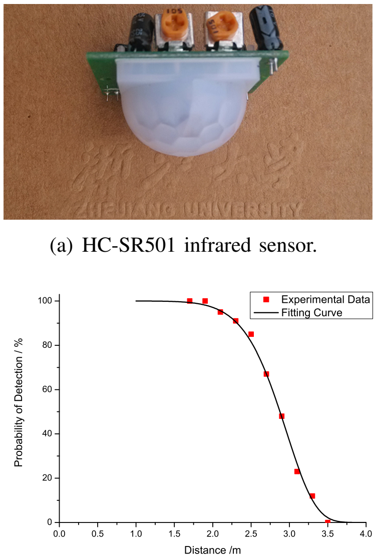
In trust management process, we adjust the trust value of each sensor by comparing its individual decision and the collective decision obtained at the sink. We increase the trust value when a sensor's decision is consistent with the collective decision and decrease the trust value otherwise. There are four cases: 1) $I^a_i(t) = 1$, i.e., sensor i had a right detection, 2) $I^a_i(t) = 0$, i.e., sensor i had a false alarm, 3) $I^s_i(t) = 1$, i.e., sensor i kept silent while no intruder was present, and 4) $I^s_i(t) = 0$, i.e., sensor i missed the intruder.

Specifically, we set the reward of each correct output as β ($\beta > 0$). When $I^a_i(t) = 1$, the reward is $\beta_a = \beta / P(A_i(t) = 1)$; when $I^s_i(t) = 1$, the reward is $\beta_s = \beta / P(A_i(t) = 0)$. Likewise, we set the punishment of trust value as α ($\alpha > 0$). When $I^s_i(t) = 0$ happens, we should subtract $\alpha_s = \alpha / P(A_i(t) = 1)$ from the current trust value; when $I^a_i(t) = 0$, the punishment is $\alpha_a = \alpha / P(A_i(t) = 0)$.

Denote by $tr_i(t)$ the trust value of sensor i in slot t. The trust value of each sensor is updated as follows:

$$tr_i\left(t + T_0\right) = \begin{cases} tr_i\left(t\right) - \alpha_a, \text{if } I^a_i(t) = 0 \\ tr_i\left(t\right) - \alpha_s, \text{if } I^s_i(t) = 0 \\ tr_i\left(t\right) + \beta_a, \text{if } I^a_i(t) = 1 \\ tr_i\left(t\right) + \beta_s, \text{if } I^s_i(t) = 1 \end{cases} \tag{9}$$

A sensor may have an extremely high trust value after working normally for a long time. It takes long time for



(a) HC-SR501 infrared sensor.

(b) Experimental results.

Fig. 4. Detection model verification.

TrusDet to locate a faulty sensor if its trust value is high enough to afford several punishments. Thus, we set an upper bound TR_0 on the trust value. Similarly, a lower bound with value of 0 is set.

$$tr_i\left(t\right)=\left\{\begin{array}{l} TR_0, \text{if } tr_i\left(t\right)>TR_0,\\ 0, \text{if } tr_i\left(t\right)<0,\\ tr_i\left(t\right), \text{else.}\end{array}\right. \tag{10}$$

A threshold of trust value TR_{th} is set to recognize the faulty sensors. We introduce $m_i(t)$ as the indicator: $m_i(t)=1$ means that sensor i is faulty, and $m_i(t)=0$ otherwise.

$$m_i\left(t\right)=\left\{\begin{array}{l} 1, \text{if } tr_i(t)<TR_{\text{th}},\\ 0, \text{otherwise.}\end{array}\right. \tag{11}$$

We proceed to calculate the trust weight of each sensor, that we use in the collective voting in step 2). As sensors with higher trust value are more reliable, their outputs are more trustworthy. We do not include the outputs of faulty sensors in the collective voting unless their trust values become high enough again. The trust weight of each sensor is updated according to the following:

$$r_i(t)=\left\{\begin{array}{c} \frac{tr_i(t)}{TR_0}, if\, m_i(t)=0,\\ 0, if\, m_i(t)=1.\end{array}\right. \tag{12}$$

IV. Performance Evaluation

In this section, we first conduct a set of experiments to verify the probabilistic detection model. Then we carry out experiments to evaluate the performance of TrusDet.

*1) **Model Verification**:* We chose HC-SR501 (shown in Fig. 4(a)) infrared sensor for intruder detection. Main component of HC-SR501 is an infrared ray based LHI778 probe. We conduct a set of experiments to quantify the detection probability of HC-SR501 in this section.

Specifically, a person acts as an intruder, walking in the detection area of the sensor, considering that HC-SR501 does not capture the stationary object. We conduct a set of experiments in which the distance between the person and the sensor increases from 0.1m to 3.5m with an increment of 0.2m. In each experiment, the person stays 3 minutes at the location. The probability of detection is estimated by the ratio of alarm duration and total time interval. Obviously, it is easy to decide R', within which intruders can be detected deterministically. We collect all the data when the distance is larger than R'. To decide the parameter θ, we fix the distance between the sensor and the person, and vary the orientation direction of the sensor. θ is found when the sensor detects the person at a very low probability.

We fit probabilistic detection model by the experiment data (see Eq. 1). Fig. 4(b) shows the fitting curve, and specific parameters in the detection model are obtained: $\lambda=7.010e-5$, $\beta=8.707$, θ=132.7°, $R'=1.9m$, $R=3.5m$. Clearly, the experiment results fit the detection model well.

*2) **Performance Verification**:* In this section, we set $k=3$, $\delta=30\%$, $L=19$, and $v_{max}=10m/s$ in the experiments. This leads to $d=1.01m$ and $D(\Omega)=2m$ (referring to Eq. (4) and Eq. (2)).

We place 9 sensors evenly in a straight line with distance d to detect intruders. For those at both ends of the deployment line that do not have enough neighbors, we only consider the 5 neighboring sensors on one side. We do 10 experiments for each fault probability, from 0% to 20% with an increment of 1%. In each experiment, the five types of faults are simulated by the program, which are designed to be evenly distributed.

The following miss detection rate P_{md} and false detection rate P_{fd} are used to measure the performance of fault detection: $P_{md}=\frac{|Q-F|}{|Q|}$, $P_{fd}=\frac{|F-Q|}{|N-Q|}$, where Q is the set of actual faulty sensors simulated by program, F is the set of sensors that are decided as faulty by TrusDet, and N is the set of all sensors.

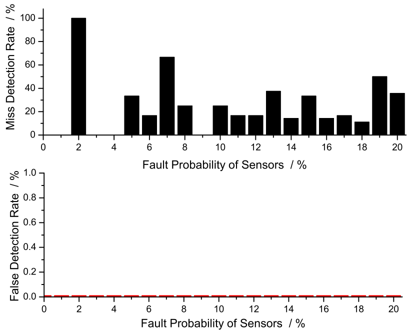


Fig. 5. Miss and false detection rates under different fault probabilities obtained by TrusDet.

At the beginning of each set of experiments, we ensure that TrusDet is assigned with an appropriate initial value (this step can be omitted in application, because newly placed sensors always have max trust value in practice). Then, an intruder (i.e., a person) walks in the area 10 times along a random route, each time being treated as the time slot of an intrusion. Between two time slots of intrusion, a silent slot (i.e., no intrusion happens) is needed. We get 10 groups × 10 times = 100 time slots to analyze miss alarm rate and false alarm rate.

We calculate miss alarm rate and false alarm rate in the following way:

$$P_{ma} = \frac{N_{ma}}{N_{slots}} \times 100\%, P_{fa} = \frac{N_{fa}}{N_{slots}} \times 100\%, \qquad (13)$$

where N_{ma} is the number of miss alarm, N_{fa} is the number of false alarm of all sensors , $N_{slots} = 100$ is the number of time slots in one experiments. We also perform experiments without TrusDet as a baseline.

*3) **Results Analysis**:* Miss detection rate, false detection rate, miss alarm rate and false alarm rate are four main indicators to evaluate the performance of TrusDet. We discuss these four performance metrics of TrusDet in the following.

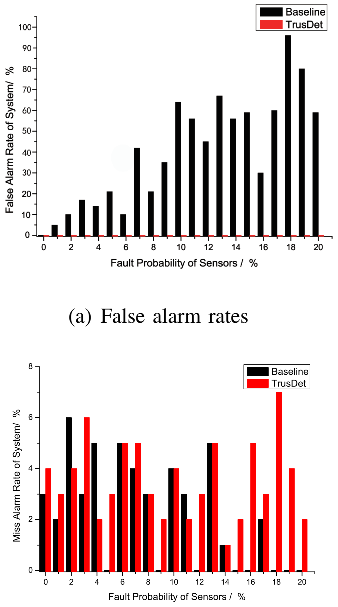
False and Miss Detection Rate. Fig. 5 shows false and miss detection rates in fault detection. We can see that the false detection rate is 0%, which is satisfactory. However, the miss detection rate is a bit high. The main reason is that it is quite difficult for TrusDet to separate the two types of silent behaviors caused by detection uncertainty and faults, especially in the case where the fault probability is very low. This is corroborated by analyzing experiment data, in which most undetected faulty sensors are those with constant silent fault. Fortunately, though sensors with constant silent fault are difficult to be identified, their behaviors are exactly the same as faulty sensors, i.e., it does not send any alarm messages to the sink. Therefore, this will not impact the performance of intrusion detection by other normal sensors.

False and Miss Alarm Rate. Fig. 6(a) shows the false alarm rates. Without TrusDet, the false alarm rate rapidly increases with the fault probability of sensors. It can be seen that TrusDet has zero false alarm rate, which shows a significant advantage of TrustDet.

Fig. 6(b) shows the miss alarm rates. When fault probability is low, miss alarm rates are almost the same for TrusDet and the baseline algorithm. The miss alarm rate obtained by baseline algorithm decreases when fault probability grows. One possible reason is that the baseline has a very high false alarm rate (refer to Fig. 5), which decreases the miss alarm rate. Note that the miss alarm rates are less than 8% in all cases for TrusDet, which is acceptable in practice.

V. Conclusion

In this paper, we have introduced a novel trust management based fault-tolerant intrusion detection algorithm (TrusDet) to improve the performance of intrusion detection system. We have implemented TrusDet on HC-SR501 infrared sensors and perform extensive experiments to demonstrate the performance of TrusDet. It is shown by the experiment results that TrusDet can recognize most faulty sensors and obtain a quite low false alarm rate. Hence, TrusDet is able to promote the performance of intrusion detection system efficiently.



(a) False alarm rates

(b) Miss alarm rates

Fig. 6. False alarm rates and miss alarm rates under different fault probabilities obtained by baseline and TrusDet, respectively.

References

[1] S. Kumar, T. H. Lai, and A. Arora, “Barrier coverage with wireless sensors,” in *Proceedings of ACM MobiCom*, 2005.

[2] S. He, X. Gong, J. Zhang, J. Chen, and Y. Sun, “Curve-based deployment for barrier coverage in wireless sensor networks,” *IEEE Transactions on Wireless Communications*, vol. 13, no. 2, pp. 724–735, 2014.

[3] Y. Liu, C. Li, Y. He, J. Wu, and Z. Xiong, “A perimeter intrusion detection system using dual-mode wireless sensor networks,” in *Proceedings of ChinaCom*, 2007.

[4] A. Saipulla, B. Liu, G. Xing, X. Fu, and J. Wang, “Barrier coverage with sensors of limited mobility,” in *Proceedings of ACM MobiHoc*, 2010.

[5] J. Chen, W. Xu, S. He, Y. Sun, P. Thulasiraman, and X. Shen, “Utility-based asynchronous flow control algorithm for wireless sensor networks,” *IEEE JSAC*, vol. 28, no. 7, pp. 1116–1126, 2010.

[6] S. He, J. Chen, X. Li, X. Shen, and Y. Sun, “Mobility and intruder prior information improving the barrier coverage of sparse sensor networks,” *IEEE Transactions on Mobile Computing*, vol. 13, no. 6, pp. 1268–1282, 2014.

[7] X. Zhang, M. L. Wymore, and D. Qiao, “Optimized barrier location for barrier coverage in mobile sensor networks,” in *Proceedings of IEEE WCNC*, 2015.

[8] L. Fu, P. Cheng, Y. Gu, J. Chen, and T. He, “Minimizing charging delay in wireless rechargeable sensor networks,” in *Proceedings of IEEE INFOCOM*, 2013.

[9] M. Ding, D. Chen, K. Xing, and X. Cheng, “Localized fault-tolerant event boundary detection in sensor networks,” in *Proceedings of IEEE INFOCOM*, 2005.

[10] M. Hugh and R. Gerald, “Intrusion alarm system,” Oct 1972, US Patent. [Online]. Available: https://www.google.com/patents/US3696359.

[11] N. Mehranbod, M. Soroush, and C. Panjapornpon, “A method of sensor fault detection and identification,” *Journal of Process Control*, vol. 15, no. 3, pp. 321–339, 2005.

[12] G. Xing, R. Tan, B. Liu, J. Wang, X. Jia, and C.-W. Yi, “Data fusion improves the coverage of wireless sensor networks,” in *Proceedings of ACM MobiCom*, 2009.

[13] Y. Zou and K. Chakrabarty, “Sensor deployment and target localization based on virtual forces,” in *Proceedings of IEEE INFOCOM*, 2003.

[14] X.-Y. Li, P.-J. Wan, Y. Wang, and C.-W. Yi, “Fault tolerant deployment and topology control in wireless networks,” in *Proceedings of ACM MobiHoc*, 2003.

[15] B. Yao and Q. Chen, “On the temporal-spatial correlation based fault-tolerant dynamic event region detection scheme in wireless sensor networks,” in *Proceedings of the International Conference on Mobile Ad-Hoc and Sensor Networks*, 2007.

[16] X. Cao, J. Chen, Y. Zhang, and Y. Sun, “Development of an integrated wireless sensor network micro-environmental monitoring system,” *ISA transactions*, vol. 47, no. 3, pp. 247–255, 2008.

[17] S. Lin, G. Zhou, K. Whitehouse, Y. Wu, J. A. Stankovic, and T. He, “Towards stable network performance in wireless sensor networks,” in *Proceedings of IEEE RTSS*, 2009.