Detection of hidden data attacks combined fog computing and trust evaluation method in sensor-cloud system

Guangxue Zhang1 | Tian Wang1,4 | Guojun Wang2 | Anfeng Liu3 | Weijia Jia4

1 College of Computer Science and Technology, Huaqiao University, Xiamen, China
2 School of Computer Science and Educational Software, Guangzhou University, Guangzhou, China
3 School of Information Science and Engineering, Central South University, Hunan, China
4 State Key Laboratory of Internet of Things for Smart City, University of Macau, Macau, China

Correspondence
Tian Wang, College of Computer Science and Technology, Huaqiao University, Xiamen, China; or State Key Laboratory of Internet of Things for Smart City, University of Macau, Macau, China.
Email: cs_tianwang@163.com

Present Address
Tian Wang, 668 Jimei Avenue, Jimei District, Xiamen, China.

Funding information
National Natural Science Foundation of China (NSFC), Grant/Award Number: 61872154, 61772148, and 61672441; Natural Science Foundation of Fujian Province of China, Grant/Award Number: 2018J01092, Fujian Provincial Outstanding Youth Scientific Research Personnel Training Program, National China 973 Project, Grant/Award Number: 2015CB352401; Chinese National Research Fund (NSFC) Key Project, Grant/Award Number: 61532013, and 61872239; FDCT/0007/2018/AMJ; University of Macau, Grant/Award Number: MYRG2018-00237-RTO, CPG2018-00032-FST, and SRG2018-00111-FST; Subsidized Project for Cultivating Postgraduates Innovative Ability in Scientific Research of Huaqiao University, Grant/Award Number: 1611314018

Summary
With the popularity of Sensor-Cloud, its security issues get more attention from industry and academia. Especially, Sensor-Cloud underlying network is very vulnerable to internal attacks due to its limitations in computing, storage, and analysis. Most existing trust evaluation mechanisms are proposed to detect internal attack issues from the behavior level. However, there are some special internal attacks in the data level such as hidden data attacks, which are normal in the behavior level but generate malicious data to lead user to make wrong decisions. To detect this type of attacks, we design a fog-based detection system (FDS), which is based on the trust evaluation mechanism in the behavior level. In this paper, three types of scenes (the redundant data, the parameter curve characteristic, and the data validation) are defined, and three detection schemes are given. Some experiments are conducted, which manifest that FDS has certain advantages in detecting hidden data attacks.

KEYWORDS
fog computing, hidden data attacks, internal attacks, sensor-cloud, trust evaluation mechanism

Abbreviations
ANA, anti-nuclear antibodies; APC, antigen-presenting cells; IRF, interferon regulatory factor
1 | INTRODUCTION

Sensor-Cloud is applied in many fields, such as industry, agriculture, transportation, military, and so on. It is an effective combination of Wireless Sensor networks (WSNs) and Cloud Computing, which liberates and expands WSNs’ capabilities by advantages of Cloud Computing in computing, storage, parallel processing, management technology, and so on. To improve the data share and physical sensor usage efficiency, Sensor-Cloud virtualizes physical sensors into Cloud as virtual sensors, which means that many applications can share physical sensors by access to virtual sensors. However, Sensor-Cloud faces some internal attack threats that malicious attackers participate in network activities and realize some illegal purposes with legal identifications. Under this circumstance, many applications using virtual sensors will be influenced and the economic loss will be immeasurable.

The trust evaluation mechanism is proposed to solve internal attack issues, and it complements each other with the traditional security mechanism (encryption, authorization, and audit). It is hard for the traditional security mechanism to detect internal attacks since the attacker has a legal identification and some rights in WSNs. However, there are still many problems that need to be solved for existing trust evaluation mechanisms such as the resource consumption in WSNs, the attack against the recommendation trust, the misjudgment of normal sensor, and so on. In our previous study, we have dealt with the above problems. In this paper, we mainly focus on the hidden data attack in the data level.

The hidden data attack is a special internal attack type from the data level, where malicious sensors behave normally in the communication with others, but they can induce user to do wrong decisions by providing biased data. The hidden data attack detection ensures sensor data credible in the data level, and it is the extension of general trust evaluation mechanisms that focus more on whether the behavior of sensor is abnormal. However, the hidden data attack detection is based on a series of data analyses that consume more resource (computing, storage and so on), which affects network performance and lifetime. Thus, it is not suitable to be taken on resource-limited WSNs.

In existing studies, many anomaly detection techniques are proposed to deal with the data injection attack and the data modification attack. However, the judgment whether a sensor is malicious should be after a long-term observation, since the malicious sensor has a legal identification. WSNs cannot undertake these global, long-term and lots of data analyses. Moreover, if these analyses perform in cloud, the timeliness cannot be guaranteed. Fog Computing has some advantage, such as low latency, certain processing power, adequate storage capacity and so on. Therefore, Fog platform can be built as an upper layer to manage the trust evaluation mechanism in WSNs and support the hidden data attack detection system.

In this paper, we combine Fog Computing and a fog-based hierarchical trust evaluation mechanism to solve the hidden data attack problem. The infrastructure is shown in Figure 1, where the trust evaluation mechanism is divided into three parts: the direct trust layer in WSNs, the preliminary decision layer among underlying fog devices, and the data analysis layer in the fog platform. Also, the detection of hidden data attack is in the fog platform, which consists of three basic parts: the redundant data part, the parameter curve characteristic part, and the data validation part. Of course, only when are information from WSNs with greater degree of credibility, the detection of hidden data attack makes sense. The fog-based hierarchical trust evaluation mechanism ensures the credibility of information from WSNs.

The rest of this paper is organized as follows: Section 2 gives related work. Three types of definitions for hidden data attacks are presented in Section 3. Section 4 is some algorithms and design details of FDS. Section 5 is experiment results and analyses. The final section is conclusion.

2 | RELATED WORK

2.1 | Trust evaluation mechanism

Internal attacks are serious security issues in Sensor-Cloud, which steal data or break network functions by legal identifications. The trust evaluation mechanism is an efficient way to solve internal attacks, which usually consists of the direct trust, the recommendation trust...
and the synthesis trust. The direct trust is based on some monitoring evidences from communication, energy and data transmission; the recommendation trust collects and calculates recommendation values from adjacent sensors or other sensors on the trust chain; the synthesis trust is a comprehensive consideration of the recommendation trust and the direct trust. For the trust evaluation mechanism in WSNs, the most fundamental requirements are high resource efficiency and dependability. However, the high overhead is the main problem for existing trust evaluation mechanisms to satisfy these requirements. So, some studies are proposed to solve the high overhead problem.

To reduce resource consumption in WSNs, some trust evaluation mechanisms adopt following means: collecting conclusive evidences rather than vast evaluation evidences (however, this system cannot deal with more complex attacks), moving the computing and storage tasks to more powerful sensors (however, cluster head should too many data forwarding, communication and processing tasks), and reducing computational complexity (however, they are not sensitive to sudden attacks due to lack of the feedback mechanism). To detect attacks against the recommendation trust (bad-mouthing attack, on-off attack, and so on), some methods are based on excluding/reducing recommendation value weight rules, such as familiarity, outlier, and deviation.

There are also two problems that need to be considered. The one is “a simple trust evaluation mechanism can only detect and prevent some basic internal attacks.” The other is “a large and complex trust evaluation mechanism can detect and prevent more internal attacks but at the cost of network lifetime and performance.” Thus, it is better to move the complex part of trust evaluation mechanism out of the underlying network.

### 2.2 Outliers detection

To determine whether a sensor is malicious, the long-term state of sensor in the data level should be considered, since there are some uncertainties, such as sensor fault, communication error, and so on. To record the long-term state of sensors, it is necessary to determine whether sensors produce outliers during detection periods.

In the current studies, there are two types of attacks in the data level: the data modification attack and the false data injection attack, whose points of attack are the data fusion or the data transmission. Many methods are proposed to detect these attacks, such as encryption, verification, and trust behavior observation. Prabha and Latha adopted an elapsed time at node (ETN) to find the data modification attack, which measures sensor’s elapsed time from the data reception to the data forwarding. Sultana et al. proposed a novel lightweight secure scheme for detecting provenance forgery, which uses the in-packet Bloom filters to encode provenance and does data verification and reconstruction at the base station. Aman et al. proposed a light-weight protocol for data provenance, which uses physical unclonable functions (PUFs) to provide physical security and uniquely identify the devices. However, these means do not solve the hidden data attack since these data are from malicious sensors with legal identifications.

In terms of sensor monitoring, sensor monitoring mode, sensor monitoring target attribute, etc, there are some data correlations among sensor data. For the general study of data correlation, there are two types: the temporal correlation and the spatial correlation. These data correlations are more used to recover missing sensor data, compress/aggregate sensor data. Yuan et al. proposed a data density correlation degree to measure the data correlation between one sensor and its neighboring sensors, which is used to do the data aggregation in the complex environment. Bijarbooneh et al. proposed a multiphase adaptive sensing algorithm, which only selects a small number of typical sensors in the network to reconstruct the missing sensor data in the cloud. Data correlations can be used to find the outlier in every detection period, and Fog Computing can be competent for the data analysis and the sensor state storage.

### 2.3 Fog Computing

The term of “Fog Computing” is formally introduced by Cisco System in 2011, which is a highly virtualized platform that provides computing, storage, and networking services between devices and traditional Cloud Computing Data Center, typically, but not exclusively locates at the edge of network. Fog Computing has many characteristics: a) low latency and location awareness; b) mobility; c) predominant role of wireless access; d) strong presence of streaming and real time applications; e) and so on. Based on these advantages, Fog Computing is used in many fields, such as Internet of Things (IoT), smart traffic light systems, smart grid, wireless sensor and actuator networks, Software Defined Networks (SDN) and so on.

Fog Computing extends Cloud Computing paradigm to the edge of network, which is a new breed of application and service. There is a fruitful interplay between the cloud and the fog, particularly when it comes to data management and analytics. Comparing to Cloud Computing providing resource distributed in the core network, Fog Computing provides elastic resource and services at the edge of network. In other words, Fog Computing facilitates the management and programming of computing, networking, and storage services between datacenter and end devices. It also supports user mobility, resource and interface heterogeneity, and distributed data analytics that address the low latency requirements of widely distributed applications.

### 3 PROBLEM DESCRIPTION

Hidden data attacks can be detected through analyzing some data correlations, which exist in the sensor arrangement mode and the target’s characteristic. In this paper, we propose three types of data correlation scenes, which exist in both dynamic and static networks. The dynamic
network or static network is distinguished by mobile or fixed sensors. Following three scenes can be found in many fields, such as target location, target tracking, industrial environmental monitoring, and so on.

### 3.1 The redundant data scene

In order to obtain more accurate and credible monitoring values, many sensor networks deploy lots of sensors in the same area for the same parameter. There are more than three sensors that can get target information as shown in Figure 2A. In the redundant data scene, monitoring values from these redundant sensors show certain redundancy correlation that is little difference among these values. In the static network, the partition and location of redundant sensors are based on network topology and sensor identification (ID). In the dynamic network, the partition and location of redundant sensors are decided by underlying fog devices that have the larger communication radius and the direction judgment function. Underlying fog devices or cluster heads can also divide the monitoring area into different blocks, which is convenient to collect and calculate network topology information. In dynamic networks, the topology is relative.

### 3.2 The data curve characteristic scene

In the second scene, monitoring values from sensors show some kinds of physics curve characteristics in the same or different area, such as linear model of sound on time, attenuation model of radiation on distance, change model of temperature on humidity and so on. In the second scene as shown in Figure 2B, sensor data from these sensors have gradualness or regularity. In the static network, we can get sensors' position information from network parameters. These position values are used to calculate the expected monitoring values based on the certain characteristic curve. Then, these expected values are compared with the real values to determine which sensors are abnormal. In the dynamic network, the position information among sensors are relative, which are recorded by sensors and transmitted to fog layer or collected by underlying fog devices. As a simple example, in the mobile target monitoring, the position of target can be got through location algorithms and we can select many groups of sensors to confirm target position. When the position of target is determined, the expected value of every sensor in this monitoring area can be calculated. Then, the comparison between one sensor's expected monitoring value and its real monitoring value is carried on.

### 3.3 The data validation scene

For the third scene, the target moves in a prescribed line or has an unchanged characteristic. In other words, the target does some repetitive motions or has the same state in the duration of existence. As shown in Figure 2C, there are two fixed moving lines for targets. If some sensors don't give monitoring values or give wrong monitoring values when the target passes, these sensors can be seen as abnormal sensors. This scene can be designed as a detection mechanism in some certain fields. Trusted targets regularly appear in the monitoring area and the detection system can find malicious sensors through the comparison between sensor's monitoring value and the known monitoring value. Compared to the static network, the dynamic network has more problems to solve, such as sensor position, sensor data timeliness, transient topological relationships, and so on. The smaller division of monitoring area can be a good solution. In divisions, sensor information and network topology are easy to collect and affirm by underlying fog devices or cluster heads.

### 4 DESIGN OF DETECTION SYSTEM

There is a trend that more and more data analysis and information extraction tasks are done in upper level, even if it brings a slight delay issue. With the improvement of technology in the network or hardware, the delay issue can be solved, such as adding some mobile sensors. Before
the design of hidden data attack detection system, it is necessary to ensure the underlying network credible. In WSNs, the trust evaluation mechanism is mature. On the basis of previous study, we designed a fog-based hierarchical trust evaluation mechanism for the underlying network, which is divided into three layers as shown in Figure 3. In the WSNs layer, the sensor monitors other sensors’ behavior evidences and performs the direct trust calculation. Moreover, the sensor also uploads its trust list and topology list to the upper level. For this design, the fog layer consists of two part: the underlying fog devices and the fog platform. Main functions of underlying fog devices consist of collecting sensor trust and topology information, doing primary decision, and passing messages from fog platform. Underlying fog devices can store sensor trust list, network state information, and topology, which are used to make some preliminary decisions and sent to fog platform. Fog platform can perform some data processing tasks and conduct network trust state analyses. The goal of this design is transferring computing and storage expenditure of trust evaluation mechanism to upper level, such as Fog Computing. Within the tolerable range of delay, the hierarchical trust evaluation mechanism can resist more attacks at the less cost of network resource consumption.

The owner of Fog Platform can be government or company with credibility, and detection information are transparent to data consumers. That is to say, the service consumer can get origin data and analysis results when they want to verify some exceptions. Fog computing is the extension of Cloud Computing, so security mechanisms that are in Cloud Computing can be used in Fog Computing, such as access control, audit, authentication and some complex trust evaluation mechanisms. So, the credibility of Fog Computing can be guaranteed.

### 4.1 Trust evaluation mechanism in the underlying network

For the trust evaluation mechanism, there are two important parts: the periodic detection and the exception handling. The periodic detection aims at maintaining trust state of sensors, and the exception handling quickly responds to unusual events, such as large change of trust value, abnormal state of sensor’s behaviors, and so on. Three layers of trust evaluation mechanism are shown in Figure 3.

#### 4.1.1 Direct trust layer

In WSNs, many evidences can be collected, such as routing success rate, packet success rate, forwarding delay, data correctness, and so on. Here, we just list three typical types of evidences: the packet success rate, the routing failure rate, and the forwarding delay.

- The trust calculation of packet (Trust\textsubscript{packet}) is shown as formula (1), which represents the trust value of general evidence. When the difference value between old packet trust value and new packet success rate (Packet\textsubscript{success}/Packet\textsubscript{total}) is less than Threshold, the packet success rate is set as Trust\textsubscript{new}\textsubscript{packet}, or else, the old packet trust value is considered in the packet trust calculation, which is done to reduce the impact of temporary environmental change. Trust\textsubscript{old}\textsubscript{packet} has less weight, which can better reflect the current trusted state of sensor

\[
\text{Trust}_{\text{new}}_{\text{Packet}} = \begin{cases} 
\frac{\text{Packet}_\text{success}}{\text{Packet}_\text{total}} & \text{if } |\text{Trust}_{\text{old}}_{\text{Packet}} - \frac{\text{Packet}_\text{success}}{\text{Packet}_\text{total}}| \leq \text{Threshold} \\
\frac{w}{\text{Packet}_\text{total}} \times |\text{Trust}_{\text{old}}_{\text{Packet}} - \frac{\text{Packet}_\text{success}}{\text{Packet}_\text{total}}| + (1 - w) \times \text{Trust}_{\text{old}}_{\text{Packet}} & \text{else}
\end{cases}
\]

where Packet\textsubscript{success} is the number of packet received, and Packet\textsubscript{total} is the number of packet sent. The smaller Threshold is, the more sensitive to abnormal situation (also, more resource consumption and misjudgment rate) the trust evaluation mechanism is. The setting of Threshold and w is according to expected convergence of packet trust value.

- The trust of routing failure rate (Trust\textsubscript{routing}) is on behalf of one type of evidences that can be used to monitor network load and provide decision-making evidences for data transmission path. If the routing failure rate of one sensor is higher than threshold value in the detection
period, it would be identified as an exception and sent to underlying fog devices. Under this circumstance, the direct trust value of this sensor is lowered (as shown as formula (2)) until fog platform replies judgment information about whether this sensor is malicious or broken-down. The recovery of trust value is determined by these judgment information and network load condition.

- The trust value of forwarding delay ($\text{Trust}_{\text{forwarding}}$) can be set as 0 or 1, which represents whether some serious abnormalities happen, such as the decision/judgment from fog platform, the forwarding delay, and the larger difference between the new and old direct trust value.

The direct trust consists of three parts as shown as formula (2): $\text{Trust}_{\text{packet}}$, $\text{Trust}_{\text{routing}}$, and $\text{Trust}_{\text{forwarding}}$

$$\text{Trust}_{\text{direct}} = (\text{Trust}_{\text{packet}} - \text{Trust}_{\text{routing}}) \times \text{Trust}_{\text{forwarding}}. \quad (2)$$

4.1.2 Preliminary decision layer

When exception emerges, the information are first sent to underlying fog devices. To reduce unnecessary energy consumption, the preliminary decision layer is only triggered by some exceptions, such as the forwarding delay exception, the difference value exception, the network state exception. In the preliminary decision layer, the integrative trust is calculated as shown as formula (3), which is based on sensors’ trust lists and network topology. Every sensor has a trust list in its memory, which is used to record adjacent sensor’s trust value and decide the next transmission sensor.

$$\text{Trust}_{\text{integrative}}^{m,j} = \sum_{i=1}^{n} \text{weight}_{m,i,j} \times \text{Trust}_{i,j}. \quad (3)$$

where $i$ is the serial number of trusted adjacent sensor of abnormal sensor $j$ and $n$ is the number of selected sensors. $m$ is one underlying fog device. $\text{Trust}_{i,j}$ is the trust value of sensor $i$ to sensor $j$. The $\text{weight}_{m,i,j}$ is calculated based on communication frequency as shown as formula (4).

$$\text{weight}_{m,i,j} = \frac{\text{communication}_{i,j}}{\sum_{i=1}^{n} \text{communication}_{x,j}}, \quad (4)$$

The preliminary layer can suitably change data transmission path and then recovery trust value of sensor based on local sensors’ trust value changes. Moreover, it can find some internal attacks aiming at the integrative trust calculation. The measure is comparing every sensor’s trust list with other related sensors’ trust lists, since underlying fog devices store the trust list of every sensor and network state information.

4.1.3 Data analysis layer

In the preliminary layer, judgments are preliminary and simple, and decisions are also temporary. The data analysis, processing, and decision are main tasks of the data analysis layer. The fog platform gets global network information and sensor data, which are used for data analyses. For sensor’s trust list transmission from underlying fog devices to the fog platform, it can be realized through compressed changing values of trust lists.

Based on the trust list, sensor data, and the network state, we can do the network global trust calculation, the sensor failure supervision, the misjudgment sensor recovery, the hidden data attack detection and so on. For the global trust calculation, it can be integrally considered from sensors’ trust lists and sensors’ data analyses. In one area, we also can analyze the network congestion by observing whether some sensors’ trust values slowly lower and network state exceptions frequently occur. For the sensor failure supervision, it can be realized through analyzing sensor’s long trust state, signal intensity and underlying fog devices’ observation. The hidden data attack detection will be introduced below in detail.

4.2 Detection of hidden data attacks

Detection algorithms for three types of hidden data attacks are given in the following. We do some data analyses to ferret out outliers in every detection period, and then, we affirm whether these sensors are malicious through analyzing sensors’ historical or future trust state. The setting of detection period is decided by the network manager since internal attacks are not always occur. The shorter the detection period is, the more the resource consumption is. However, it will not add burden to WSNs since the fog platform is out of WSNs.

4.2.1 The detection method for the first scene

For this detection method, the trend similarity analysis (TSA) is the basic calculation cell, and the malicious sensor extraction (MNE) is the main body. TSA is triggered by period detection, which only randomly detects a small number of sensors. MNE is triggered by exceptions from the underlying network and TSA, and it also executes when it’s free time exceeds the threshold value. For the redundant data scene, it is important to get a more accurate sensor value. In general, the data compression can be done in WSN. However, when the detection period comes, the
entire sensor data should be uploaded to the upper level rather than a compressed sensor data. Of course, the data compression sensor (sink sensor) can find central value and compress sensor data according to the degree of deviation from the center value.

### TSA

In TSA, we set an array $Array_{trend}$ for every redundant sensors, which records the sensor data change trend and some relative parameters. $Array_{trend}$ adopts FIFO(First In First Out) structure, which only stores sensor information for certain detection periods. There are five main components in $Array_{trend}$, which are the identification (ID) of redundant sensor, the monitoring value at one detection period, the monitoring value variation of two adjacent detection periods, the state value (peak, trough or intermediate state) and the error-tolerant number that avoids temporary fault or error data influencing peak/trough record.

In formula (5), $Trend_{grade}$ represents two monitoring values’ change rate between two adjacent detection periods. When the value of Error-tolerant is set as 2, if the value of $Trend_{grade}$ is negative in one detection period and is positive in the next two detection periods, we set the state value as $-1$ (Trough) in $Array_{trend}$. If the value of $Trend_{grade}$ is positive in one detection period and is negative in the next two detection periods, we set the state value as $1$ (Peak) in $Array_{trend}$. Otherwise, the state value is set as 0 (intermediate state). When the state value is 1 or $-1$, if the next value of $Trend_{grade}$ is still 0, the state value would be the same with previous state. $Y_1$ and $Y_2$ are two adjacent monitoring values from one sensor, and $X_1$ and $X_2$ are two corresponding adjacent detection periods. The variation of monitoring values is $Y_2 - Y_1$. For all redundant sensors, their $Array_{trend}$ are updated when monitoring values are collected. When the detection time arrives, the contrast analysis begins. Firstly, every sensor’s peaks/troughs are contrasted at the corresponding time. Then, monitoring values at corresponding peaks/troughs are contrasted. If some peaks/troughs occur in different time or monitoring values at the same peaks/troughs have a large gap, MNE would be triggered

$$Trend_{grade} = \frac{Y_2 - Y_1}{X_2 - X_1}. \quad (5)$$

### MNE

The outlier is extracted in every detection period of MNE, and we set an array $Array_{monitoring}$ to record some intermediate results in every detection period as shown in Table 1. The Value0 column stores the monitoring value of every redundant sensor at one detection period. The Value1 column stores the total deviation distance of initialization value between one sensor and all other redundant sensors, the formula is shown as (6). The Value2 column indicates the position of sensor in ordered Value1 sequence, and we select the first value in ordered Value1 sequence as base value to find outliers. The Value3 column stores different levels, such as normal (0), tolerableness (1), and outlier (2).

The “Deviation” of every redundant sensor is calculated through formula (6). $Data_0$ is the monitoring value of sensor that needs to get Deviation. $Data_{i}$ is the one monitoring value of redundant sensors and $n$ is the number of redundant sensors. To reduce the computing resource consumption in formula (6), we divide the interval between the minimum to the maximum into many equal parts. Then, we count the number of sensors in every part and determine the base interval based on several consecutive parts who contain more sensors. Finally, if sensor’s value is out of the base interval, it is considered outlier. If the number of normal sensors is larger than malicious sensors, the minimum in Value1 is more likely from normal sensors. Sensor that has larger Deviation is considered as abnormal sensor and assigned corresponding “level” value of Value3. The Value3 is set to reduce misjudgment rate, since minor threshold value causes too many misjudgments and larger threshold value misses chance

$$Deviation_i = \sum_{k=1}^{n} |Data_0 - Data_{i}|. \quad (6)$$

Since there may be some temporary network or sensor malfunctions, we need to do outlier detection more than once. In formula (7), $A$ is the initial matrix of monitoring values for all redundant sensors. The number of sensors is $n$ and the number of detection periods is $m$ in $A$. During the outlier detection of matrix $A$, monitoring values are converted to corresponding ‘level’ of Value3. An example of the converted matrix is shown as $A'$. There are some means to decide whether one sensor is malicious. Means-1: we check whether sensor’s previous records are consecutive outliers, if this case occurs, we think they are malicious. Means-2: we calculate the proportion of sensor’s outliers in certain number of detection periods, if the proportion value is larger than threshold value, it is a malicious sensor. Means-3: if sensor has outlier in the previous one or two detection periods, we will focus on the detection result for next several periods and judge whether it is in a continuous abnormal state.

$$A = \begin{bmatrix}
ad_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\
ad_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,m} \\
a_{3,1} & a_{3,2} & a_{3,3} & \cdots & a_{3,m} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,m}
\end{bmatrix} \rightarrow A' = \begin{bmatrix}
0 & 1 & 0 & \cdots & 0 \\
2 & 2 & 1 & \cdots & 2 \\
0 & 0 & 1 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & 1 & 0 & \cdots & 0
\end{bmatrix}. \quad (7)$$

### Table 1

<table>
<thead>
<tr>
<th>Name</th>
<th>Value0</th>
<th>Value1</th>
<th>Value2</th>
<th>Value3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node ID</td>
<td>Initialization Deviation Ordinal Level</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

ZHANG ET AL.
4.2.2 The detection method for the second scene

In the second scene, the fog platform should have some knowledge about targets and their characteristics, such as the speed of sound in air, the signal attenuation at different distances, and the diffusion velocity of solute in different solvents, and so on. In addition, the position of sensor in the monitoring area can be obtained by underlying fog devices. If there is the static network, underlying fog devices can directly detect and record the whole network topology and sensor position. For the dynamic network, underlying fog devices should establish a relative network topology and calculate position of every sensor by the orientation and distance perception. The expected monitoring value of every sensor can be calculated by sensor's relative position in the monitoring area. Then, abnormal sensors can be detected through the comparison between expected monitoring values and real monitoring values. At the end, malicious sensors can be determined through analyzing sensors' overall state during certain detection periods. The detection algorithm for the second scene is shown as Algorithm I, whose name is hidden data attack detection based on parameter curve feature (HDAPCF).

Algorithm 1 HDAPCF algorithm

Require: target characteristic curve - Characteristic\textsubscript{curve}, real monitoring values - Value\textsubscript{real}, the number of sensors in monitoring area - n, perceptual information of every sensor - Sensor

Ensure: the set of abnormal sensors - Array\textsubscript{abnormal}

Table\textsubscript{connection} = f\textsubscript{1} (Sensor\textsubscript{1}, Sensor\textsubscript{2}, ..., Sensor\textsubscript{n}) / get the connection information of sensors

Table\textsubscript{distance} = f\textsubscript{2} (Sensor\textsubscript{1}, Sensor\textsubscript{2}, ..., Sensor\textsubscript{n}) / get the distance information between sensors and underlying fog devices

Topology\textsubscript{relative} = f\textsubscript{3} (Table\textsubscript{connection}, Table\textsubscript{distance}) / calculate the relative network topology in the monitoring area

while (Rate < Threshold\textsubscript{1}) do //Rate is the ratio that the number of target position in the same position range occupy the total number

for j = 1 : Threshold\textsubscript{2} do

    Position\textsubscript{j} = f\textsubscript{4} (Value\textsubscript{1}, Value\textsubscript{2}, Value\textsubscript{3}) //Determine target position

    if Position\textsubscript{j} in Array\textsubscript{position}(x) then //If the target position range already exists, the number of times is added one

        Array\textsubscript{position}(x, 2) = Array\textsubscript{position}(x, 2) + 1

    else

        Array\textsubscript{position}(i + 1, 1) = Position\textsubscript{j}

        Array\textsubscript{position}(i + 1, 2) = 1

    end if

end for

for k = 1 : 1 do //Calculate the ratio for every target position range

    Rate\textsubscript{k} = Array\textsubscript{position}(k, 2) / max(Threshold\textsubscript{2})

    select maximum in Rate\textsubscript{k} as Rate

end for

end while

for i = 1 : n do

    Value\textsubscript{expected} = Characteristic\textsubscript{curve} (Position\textsubscript{i}, Position\textsubscript{target})

    if |Value\textsubscript{real} - Value\textsubscript{expected}| > Threshold\textsubscript{3} then

        set i into Array\textsubscript{abnormal}

    end if

end for

HDAPCF executes when the detection period arrives. In the monitoring area, the connection state among sensors, Table\textsubscript{connection}, are recorded through sensors’ route lists or trust lists. The distances from underlying fog devices to sensors, Table\textsubscript{distance}, are calculated by underlying fog devices based on Received Signal Strength Indication (RSSI). The relative network topology in the monitoring area, Topology\textsubscript{relative}, can be got based on Table\textsubscript{connection} and Table\textsubscript{distance}, which is used to locate target position range. There are many sensors in the monitoring area, but we have no knowledge of which sensors are malicious. So, we select three sensors to locate the position range of target. Firstly, two sensors locate the possible position range of target. Then, the third sensor decides the position range of target. There are two means for selecting sensors, which are the interconnection sensors and the non-interconnection sensors. In these groups, groups of non-interconnection sensors tend to occupy a larger proportion. If the target position is always located in one range and the ratio that the number of target position in this range occupy the total number is larger than Threshold\textsubscript{1}, we set the center of this position range as target’s position. Threshold\textsubscript{2} is the number of groups for one time of calculation. Every target position range and the number of times of appearance are recorded in Array\textsubscript{position}. The target’s final position is related to maximum of Rate. After determining the final position of target, we calculate expected monitoring value of every sensor according to Characteristic\textsubscript{curve}. If the difference value between expected monitoring value and real monitoring value is larger than Threshold\textsubscript{3}, the sensor is determined abnormal and put in Array\textsubscript{abnormal}. 
The determination means for malicious sensors are similar to the first scene. If there is a larger proportion of malicious sensors, there would be some invalid combinations or be many valid target position ranges. Under the circumstances, the validation of sensor should be carried on by the network manager or the fog platform.

4.2.3 The detection method for the third scene

For the third scene, we also need to get the relative network topology. The fixed attributes of targets are known by the fog platform. When targets appear in the monitoring area, sensors send their monitoring values to the fog platform. The fog platform compares fixed attributes and real monitoring values from sensors. If the difference between two types of values exceeds threshold, these sensors are considered abnormal. When the detection period comes, the fog platform will scan the sensor data stream from WSN and find sensors that have detected the same target. Then, according to target’s state and network topology, all sensors that can detect target are found. There are two cases that can explain this scene. One is that the target moves on a fixed line, and the other is that the target with fixed attribute move freely in the monitoring area. For the first case, the fog platform can know which sensors are in the fixed line. For the second scene, the fog layer can foresee which sensors can monitor the target by Topologyrelative. Algorithm II is proposed below, which is called hidden data attack detection based on fixed attribute target (HDAFAT). There are two types of sensors that are put into Arrayabnormal. One is that sensors don’t send monitoring values to the fog platform even if the target has moved into their monitoring scope. The other is that sensors send monitoring values to the fog platform but the difference value between monitoring value Monitorreali and fixed attribute Attributefixedi is beyond Threshold. Abnormal sensors are stored into Arrayabnormal; then, we check whether these abnormal sensors are malicious. The determination means for malicious sensors are similar to the first scene.

Algorithm 2 HDAFAT algorithm

Require: fixed attribute of target - Attributefixedi, real monitoring values - Valuereali, the number of sensors in monitoring area - n, perceptual information of every sensor - Sensor
Ensure: set of abnormal sensors - Arrayabnormal

$Table_{connection} = f_2 (Sensor_1, Sensor_2, Sensor_3, ... , Sensor_n)$ /get the connection information of sensors

$Table_{distance} = f_2 (Sensor_1, Sensor_2, Sensor_3, ... , Sensor_n)$ /get the distance information between sensors and underlying fog devices

$Topology_{relative} = f_3 (Table_{connection}, Table_{distance})$ /calculate the relative network topology in the monitoring area

for $i = 1 : n$

if $|Monitor^real_{i} - Attribute_{fixed_{i}}| > Threshold$, then

set $i$ into $Array_{abnormal}$

end if

end for

4.2.4 Malicious/misjudgment sensor processing

Since all sensors had registered in WSNs with identifications, these malicious sensors would be delisted from sensors’ route lists. Moreover, the register of new sensor must be audited by the fog platform or network manager.

If there are misjudgment sensors, the recovery of their legal identifications must be through confirm of the fog platform or network manager.

5 PERFORMANCE EVALUATION

In this section, we carry out some simulation experiments for three types of hidden data attacks based on MATLAB R2016B.

In experiments, there are three clusters corresponding to three types of hidden data attacks. For the first scene, there are six types of monitoring values and every type of monitoring value is monitored by twenty sensors. For the second scene, there are three types of monitoring values and every type of monitoring value is monitored by twenty sensors. For the third scene, there are two experiments and every experiment consists of twenty sensors, which are the fixed line and the fixed attribute.

5.1 The first scene

There are two phases in the first scene experiment: abnormal sensor detection and malicious sensor determination.

As shown in Figure 4A, it is the phase of abnormal sensor detection, which indicates the ratio of correct judgment under different settings of both malicious sensor number and the base interval. In this phase, every type of monitoring value has a fault tolerance interval length, LengthInterval. In this experiment, we test six types of base interval, Intervalbase, and set the maximum distance from Intervalbase is six LengthInterval. The larger the distance from real monitoring value to Intervalbase is, the higher the ratio of correct judgment for malicious sensors is. The reason is that the
missing judgments of malicious sensor become more with the distance becoming larger. When the ratio of malicious sensors is higher, the ratio of correct judgment for malicious sensors becomes lower. The reason is that more malicious sensors influence the judgment of the base interval, and normal sensors are judged to be abnormal. When the ratio of malicious sensors is exceed 50%, we think that network is severely paralyzed. Of course, the threshold value that is used to distinguish malicious sensors has a certain influence on the misjudgment ratio of normal sensors as shown in Figure 4B. With the threshold value increasing, the ratio of misjudgment for normal sensors is lower. The reason is that the sensitivity of abnormal sensor detection becomes larger with the threshold decreasing, which can lead to misjudgment on normal sensors.

After the detection of abnormal sensors, the determination of malicious sensors is executed. In the experiment, we save every sensor's state information for twenty detection periods. We set 1 when sensor is detected as abnormal sensor at one detection period, otherwise set 0. Firstly, we check sensor's previous three records to ascertain whether it is in continued abnormal state. If this case occurs, we think it is malicious sensor. Secondly, we calculate the proportion of sensor's abnormal state. If the proportion value is larger than 20%, it is a malicious sensor. Thirdly, if sensor is abnormal in the previous one or two detection periods, we will continue to focus on its state for next few periods and judge whether it is in continued abnormal state.

5.2 The second scene

There are also two phases in the second scene experiment: abnormal sensor detection and malicious sensor determination.

For the second scene, the ratio of malicious sensors and the number of groups influence the detection of abnormal sensors. In the abnormal sensor detection, we set the number of basic groups respectively as 3, 4, 5, 6, 7, and 8 for one time of execution. As shown in Figure 5A, when the number of basic groups increase, the misjudgment ratio of normal sensors tends to reduce. The reason is that the influence of malicious nodes reduces with the increasing of participating nodes' number when the ratio of malicious nodes is lower. However, more number of basic groups would consume more calculation resource. With the ratio of malicious sensors increasing, there will be more misjudgments on the target position, which cause the misjudgment of normal sensor. When the number of basic groups is fixed, we can find that the ratio of malicious sensors would cause the ratio of correct judgment of malicious sensors lower as shown in Figure 5B.

After the detection of abnormal sensors, the determination of malicious sensors is executed. The determination means of malicious sensors are similar to the first scene.

5.3 The third scene

For the fixed line experiment, we establish a table that records some information for determining abnormal sensors or failure sensors as shown in Table 2. There are three targets in the same time slot. When targets move in fixed lines, sensors in these lines will send their monitoring values (target, time) to sink sensor or underlying fog devices. Of course, when the detection period comes, the fog platform can get these data and store them into $Array_{real}$. The fog platform knows the information of lines, and store targets' state information (target) and time that targets may arrive at sensors into $Array_{expected}$. To detect abnormal sensors, we contrast $Array_{expected}$ and $Array_{real}$ in the same time slot and record all different values between two arrays. As an example, the sensor whose number is 7 is an abnormal sensor or fault sensor in Table 2. The determination of malicious sensors is similar to the first scene.
FIGURE 5 The second type of scene. A, Misjudgment ratio of normal sensors; B, Correct judgment ratio of malicious sensors

**TABLE 2** Information for abnormal sensors determination

<table>
<thead>
<tr>
<th>Name</th>
<th>Line</th>
<th>Array\textsubscript{expected}</th>
<th>Array\textsubscript{real}</th>
<th>Time Slot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target1</td>
<td>Line1</td>
<td>1.2,7,12,15,19</td>
<td>1.2,12,15,19</td>
<td>Slot1</td>
</tr>
<tr>
<td>Target2</td>
<td>Line2</td>
<td>13,14,10,6,7,4</td>
<td>13,14,10,6,4</td>
<td>Slot1</td>
</tr>
<tr>
<td>Target3</td>
<td>Line3</td>
<td>5,10,11,8</td>
<td>5,10,11,8</td>
<td>Slot1</td>
</tr>
</tbody>
</table>

**TABLE 3** Information for abnormal sensors determination

<table>
<thead>
<tr>
<th>Name</th>
<th>Array\textsubscript{expected}</th>
<th>Array\textsubscript{real}</th>
<th>Time Slot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target1</td>
<td>1.2,3,4,5,6,7,8</td>
<td>1.2,3,4,5,6,7,8</td>
<td>Slot1</td>
</tr>
<tr>
<td>Target2</td>
<td>9,10,11,12,13,14,15,16</td>
<td>9,11,12,13,14,15,16</td>
<td>Slot2</td>
</tr>
<tr>
<td>Target3</td>
<td>17,18,19,20</td>
<td>17,18,19,20</td>
<td>Slot3</td>
</tr>
</tbody>
</table>

For the fixed attribute experiment, we also establish a table that records some information for determining abnormal sensors or failure sensors as shown in Table 3. There are also three targets in the different time slot, and it will consume some network resource when underlying fog devices get these target information. When targets appear in the underlying network, sensors will send monitoring values to sink sensor or underlying fog devices. The fog platform knows these fixed attributes. When one target’s appears, underlying fog devices locate target and find sensors around it. The fixed attribute of target and sensors’ IDs are stored into Array\textsubscript{expected}. To detect abnormal sensors, we contrast Array\textsubscript{expected} and Array\textsubscript{real} in the same time slot and record all different values between two arrays. As an example, the sensor whose number is 10 is an abnormal or fault sensor. The determination of malicious sensors is similar to the first scene.

6 | CONCLUSION

The trust evaluation mechanism is an important means to ensure Sensor-Cloud security. However, there is one type of hidden data attacks, which is rarely considered in existing studies. For the detection of hidden data attacks, it needs to carry on more data analyses, which are not suitable for realization in WSNs. We design a FDS to detect hidden data attacks in three scenes based on fog computing and trust evaluation mechanism. Moreover, these schemes can be combined to satisfy complicated situations. We do some experiments for three scenes, respectively, which validates that hidden data attacks can be found in some extent.

ACKNOWLEDGMENTS

This work was supported in part by the grants from the National Natural Science Foundation of China (NSFC) under grants 61872154, 61772148, and 61672441, the Natural Science Foundation of Fujian Province of China under grant 2018J01092, the Fujian Provincial Outstanding Youth Scientific Research Personnel Training Program, the National China 973 Project No. 2015CB352401, the Chinese National Research Fund (NSFC) Key Project Nos. 61532013 and 61872239, the FDCT/0007/2018/A1, DCT-MoST Joint-project No. (025/2015/AMJ), the University of
REFERENCES


