# MODIFIED FOG-BASED TRUST METHOD OF DATA MONITORING FOR MULTI-SENSOR CONFIGURATION SYSTEMS

*Roman Diachok, PhD Student; Lviv Polytechnic National University,*
*Halyna Klym, Dr.Sc., Prof.; Lviv Polytechnic National University, Ivan Franko National University of Lviv, Ukraine;*
*e-mail: halyna.i.klym@lpnu.ua*

**Abstract.** A modified Fog-based trust method to prevent third-party interference in establishing trust relationships between sensors and cloud service providers in multi-sensor systems is considered. Trust in behavior between nodes is established at the level of wireless sensor networks; in the nodes and objects data at the Fog layer. With more detailed data analysis of the latter, it becomes possible to monitor the trust status of the entire network, detect data attacks and recover from misjudged nodes. Fog layer can be built as a reliable third party. Experiment results show that the proposed trust mechanism is inherent in advantage due to reducing energy consumption and ensuring the trust state of Edge nodes and whole the network as well as detecting hidden attacks on data and recovering nodes.

**Key words:** Multi-sensor platform; artificial Intelligence; cyber-physical system; machine learning.

## 1. Introduction

Recent advances in electronic and wireless communications have revolutionized the IoT with the development of miniaturized, multi-sensor systems that can harness and manage data collection and sharing. These advantages have made it possible to design small, cost-effective, and low-power multifunctional sensor platforms capable of monitoring and transmitting information in such sectors as automotive, healthcare, industry, etc. [1].

The IoT approach is combined with machine learning based on innovative algorithms to obtain valuable information useful for intelligent cyber-physical systems (CPS) that contain physical and cyber parts connected through a network [2-3]. The physical part consists of sensors and actuators that collect the data and perform tasks based on the collected information. The collected data is sent over the network to the cyber part, where is stored and processed. Advanced machine learning algorithms are implemented to produce information for the physical part. CPS is presented in a variety of industries including manufacturing, logistics, oil/gas, transportation, energy/services, mining, metallurgy, and aviation [4-5]. Industrial CPS can build autonomous self-service machines and improve inventory management through machine learning. It is the basis of Industrial IoT, which can collect transaction data and send it over the network to cloud servers, where it is analyzed and stored. The main tasks of IoT in this case are focused on storing and manipulating data of IoT devices equipped with sensors that are characterized by low computing power and small memory.

## 2. Disadvantages

Due to the rapid growth and diversity of IoT-connected devices, the traditional centralized network architecture must meet new service requirements, and challenges, and effectively identify and provide large amounts of data concerning security, integrity, and privacy. It is important to develop methods for cleaning data in such networks [6]. To solve the mentioned problems and achieve a better quality of service (QoS) and quality of experience (QoE) by performing data storage and processing operations physically near the data source in a distributed infrastructure, Fog/Edge computing is applied [7-8].

## 3. Goal of the Work

The goal of this work is the modification of the trust method based on Fog computation to prevent third-party interference when trust relationships are established in a network with a multi-sensor configuration, as well as to detect hidden attacks on data and recover nodes of incorrect evaluation.

## 4. Industrial Cyber-Physical System

A typical scheme of an industrial CPS for efficiency and productivity of industrial processes [9] is shown in Fig. 1. It is based on the combination of IoT, and machine learning. Typical IoT applications are based on wireless networks of sensors that collect and transmit data to a measurement storage center.

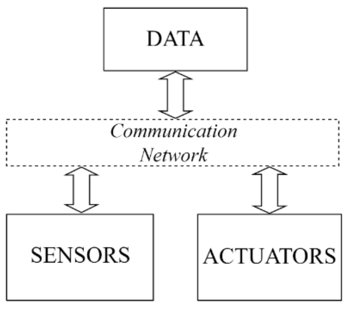


*Fig. 1. Typical structure of Industrial CPS*

IoT applications involve different types of data, including emergency response, real-time video surveillance, computer vision, and autonomous driving, the whole of which have quality requirements for the processing of received data that can change over time, such as latency, and bandwidth. IoT must be able to adapt to these variations and provide each device with the services it needs. The network information shared may be confidential, require protection, and provide limited and controlled access to the data.

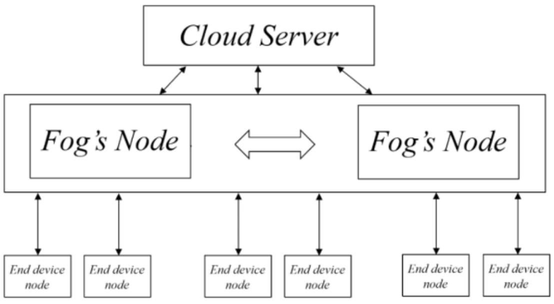


*Fig. 2. Fog computing architecture*

Most IoT devices are vulnerable because they have limited security capabilities and can be relatively easily compromised by accessing stored information or sending incorrect data to the cloud. The security of the IoT system is important and should be guaranteed both at the network level (transmission channel) and at the level of the storage system (cloud server). Therefore, Edge and Fog computing now can offload the cloud. Fog computing can be considered an extension of the cloud computing paradigm [9], presented in the aligned service structure, as shown in Fig. 2. It allows more local monitoring of data obtained from multi-sensor CPS systems in real-time and their modification IoT applications, while the cloud provides global optimization and other advanced services.

In general, the Fog architecture consists of three layers: 1. End-device layer. It includes IoT end-user devices that manage data generation. Their main task is to perceive surrounding objects and events and transmit data to upper levels for storage and processing. 2. Fog level, the middle level, which has a significant number of Fog nodes (further FNs) - servers, routers, and switches, which are located at the Edge level of the network and distributed geographically. FNs are connected to the end-device layer via wireless technologies (Wi-Fi, 4G, Bluetooth). They analyze and store the received data and send only valuable data to a cloud server for further data processing. 3. The cloud layer, is inherent in capabilities for computational analysis and permanent storage of big data. For optimal efficiency, only a few computing and storage operations are performed by the cloud layer.

## 5. Multi-leveled architecture

There we consider a platform that is a multi-level architecture for analyzing data coming from intelligent IoT-based devices. It consists of a cloud computing layer, Fog systems, and sensors that operate together. It is known that Fog computing is a virtualization technology that offers storage and computing between end devices and the cloud layer [10]. The schematic diagram of this architecture is shown in Fig. 3.

The first physical layer consists of an IoT device set and several different sensors which collect data and send it to Fog/Edge gateways. Once the data arrives at the Fog/Edge gateways, it needs to be filtered and pre-processed for further processing. This process removes 30–70% of incorrect data from further analysis. Due to this the load on the data transfer channel and the increase in the processing speed of data analysis can be reduced. This layer acts as a server. The data volume is delivered to the Fog/Edge servers and then distributed among the different Fog/Edge devices according to the data computation requirements to reduce real-time latency. Work on unloading requests should be carried out using the proposed efficient algorithm. Fog computing makes it easy to pre-process data before it even enters the cloud, minimizing communication time and reducing the need to store massive amounts of data through filtering. The described approach is closely related to the Fog computing architecture. Fog/Edge devices work offline. For the set of computing tasks within the proposed architecture, data analytics and an offloading node were introduced. If tasks exceed resources from a given Fog/Edge server, the last offloads to another Fog/Edge server on-premises or in the cloud.

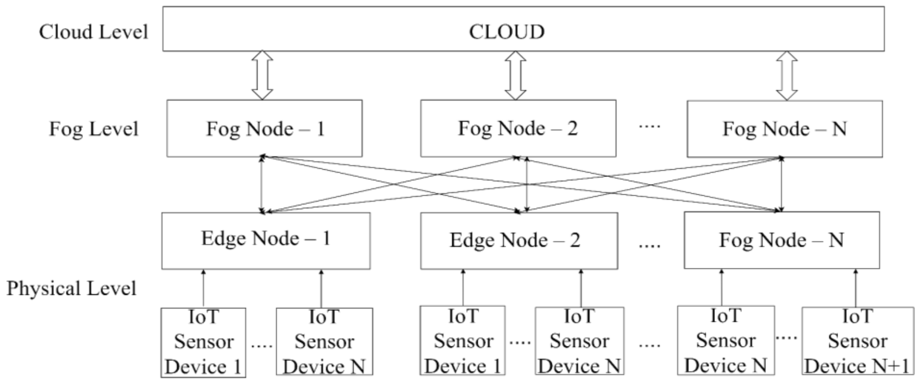


*Fig.3. FN Architecture*

An FN, which provides data analysis methods and capabilities for IoT devices has to be able to communicate and collaborate with the cloud layer and devices at the Fog layer. It provides capabilities for Fog data to and from the cloud, as well as a gateway. The latter allows end devices not directly connected to the Internet to access cloud services. Although the “gateway” provides a specific networking-oriented function, it is also a concern to a group of end devices that manage and process data on behalf of their clustering

Fig. 4 is shown an algorithm for analytical data processing using the FN with an exponential increase in the volume and size of data [10]. Streaming data is analyzed locally at the FN, while data of the last is collected and transmitted to the cloud for offline analytics and further processing. Data analytics nodes deployed on FNs are updated periodically taking into account policies adopted and communicated by cloud analytics.

The raw data is pre-processed, filtered, and cleaned in the FN before being uploaded to the cloud nodes, the amount and size of the uploaded data are smaller than the data generated by IoT devices. In addition, analytics on the FN is performed in real-time, while analytics in the cloud is performed offline. An FN has limited computing power and storage capacity compared to the cloud side but cloud-side processing and management require higher latency. An FN offers a high level of fault tolerance, as tasks can be handed over to other nearby FNs in the event of a failure. With the advent of a resource-based IoT that enables high-speed real-time applications, the best approach seems to be to move analytics to the data source and make possible real-time processing. In the future, an FN can host many different hardware components, such as a multi-core processor, and a high-detail graphics processor, compared to a cluster of similar nodes in the cloud.

Recent research indicates that Fog/Edge computing technology provides an opportunity to overcome the hardware limitations of the end-user device. Limitations can be provisioning by offloading computationally intensive tasks to powerful Fog/Edge servers for further processing. Execution on Fog/Edge servers meets task requirements and delivers results to end devices. The Fog computing paradigm brings both computing and network resources closer to the user. An FN deploys a multi-FN offloading network architecture. The offloading scheme was proposed keeping in mind the selection of FNs according to task scheduling metrics, and then offloading tasks to FNs that require the minimum task delay. Whenever a compute task is created on a terminal node. The number of FNs is selected according to the performance requirements and characteristics of those nearest FNs. Instead of computing a task locally, it is split into multiple subtasks and offloaded to these selected FNs for computation. After that, the calculation results are sent back to the terminal node.

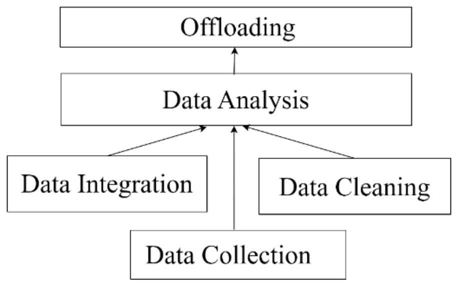


*Fig. 4. Data Analytics on the Fog level before sending to the cloud level*

The nearest FNs with the most powerful computing capabilities were selected to achieve the minimum task delay and the best performance [11]. The cycle of acquisition, processing, and activation in applications is performed with the help of sensors that transmit the received data to the cloud side, where it is processed, and executive mechanisms are notified of the need for action. The cloud layer was primarily responsible for complex, resource-intensive tasks and rule updates for fog-level detection. The data collection stage is the main aspect of these solutions, which establishes communication protocols between IoT software platform components, global data analysis, and overall resource monitoring. Cloud provides an approved FN model that was a user attribute. Depending on the IoT application, in case of limited access to power, the Fog core can run on battery power and must be energy efficient, while the cloud is supported by a constant power supply.

## 6. Design of modified trust method

Many behavioral characteristics can be applied to evaluate the true state of nodes during the communication process between them. However, some features that resulted in complex-system implementation due to software and hardware limitations should be considered, such as energy consumption, network load, and others. For the modified trust method, we choose packet loss rate, route failure rate, and forwarding delay as parameters to evaluate the trust state of a node.

$Truth_{packet}$ loss rate refers to the ratio of the number of data packets lost by the receiver to the total number of data packets in the communication cycle. This is a piece of evidence that can indicate the state of a node or whether a node is compromised [12]. The route error rate is the ratio of the number of routing packets rejected by the receiver to the total number of routing packets sent by the sender during a certain time interval. This is evidence that can indicate the state of the network. $Delay_{forwarding}$ refers to the time interval between receiving data and forwarding data when the relay node transmits the data. This is evidence that can indicate the node is compromised or has a serious fault. The source node can apply this evidence to establish direct trust relationships on shared nodes. Moreover, the observed value of a node's behavior may fluctuate with the change in environment and network load. So, the value of the $Truth_{history}$ trust history is added to the direct trust calculation to reduce the rate of misjudgments of common nodes and the unnecessary consumption of network resources. Direct trust is calculated according to equation (1) [12]:

$$Truth_{direct} = (w_1 Truth_{packet} + w_2 Truth_{history}) \times \times Delay_{forwarding}\ , \qquad (1)$$

where $Delay_{forwarding}$ is an important indication of a serious security problem if the relay node has changed the data. When the time interval exceeds the threshold, $Delay_{forwarding}$ is set to 0, otherwise, it is set to 1. If a $Delay_{forwarding}$ exception occurs, the $Truth_{direct}$ value is 0. Otherwise, the $Truth_{direct}$ value is determined by the $Truth_{packet}$ and $Truth_{history}$ based on a weighted algorithm. For weighted values $w_1 + w_2 = 1$. To reduce the energy consumption of a node during data transmission, the trust discovery period between nodes is maximized within an acceptable range. In this case, the trust value may become too old to currently reflect the current trust state of the node. So, the weight of $Truth_{history}$ can be reduced using equation (2):

$$w_2 = real_1 \times Period_{network} \times \times exp(-real_2 \times Period_{network}), \qquad (2)$$

where $Period_{network}$ is the period from the last update to the present time; $real_1$ and $real_2$ are two real numbers that are set during initialization.

We take the derivative of a function to check its trend of change. Then find the region of the decline of this function. Finally, an appropriate descent region is selected to adjust the $Period_{network}$ weight:

$$(w_2) = (real_1 \times Period_{network} \times \times exp(-\ real_2 \times Period_{network})) = =real_1 \times (1-\ real_2 \times Period_{network}) \times \times exp(-\ real_2 \times Period_{network})\ (w_2) = 0,$$
$$sthen\ Period_{network} = (\ 1/real_2\ ) \qquad (3)$$

The dependence was decreasing if $Period_{network}$ exceeds ($1/real_2$), it decreases sharply in the 1st part and smoothly decreases in the 2nd part. To achieve an almost perfect result, the value of $real_2$ can be set to [0.7,1], and the value of $real_1$ is set according to $real_2$, which can set different weights of coefficients for $truth_{\ history}$ according to the different periods [11]. In this layer, the source node requests recommendation values from its trusted neighbors when it finds some neighbor exceptions. Meanwhile, the source node also sends these exceptions to the fog layer to analyze the trust status of each node in that region. If these abnormal nodes are determined to be malicious, the Fog layer notifies the cluster to isolate the malicious nodes.

Exceptions in a sensor network are divided into 3 categories: route error rate exception, forwarding delay exception, and difference value exception. Routing error is a normal phenomenon in sensor networks, but it is considered an exception when the route error rate reaches a threshold value in a certain period. While the forwarding delay exceeds the threshold, a forwarding delay exception occurs. A difference value exception is when the difference value between the new confidence value and the historical confidence value is outside the reasonable range. The equation for calculating trust in recommendations had the form (4):

$$Trust_{recommendation} = = {}_{i \in set(neighbor)} w_{i(i,j)} \times Trust_{(j,k)}\ , \qquad (4)$$

where *set(neighbor)* is a set of trusted nodes of the source node, $Truth_{(j,k)}$ is the trust value of node $j$ to node $k$. However, the source node has different trust values for different neighboring nodes. In this case, there should be some mechanism to properly mitigate the impact of low-performing nodes. Then the trust table was sorted of the source node from small to large by trust values, and then calculate the weighted value of each neighboring node using the arithmetic progression of equation (5):

$$w_{i\ i,j} = \frac{i}{{}_1^n i} = 2 \times \frac{i}{n(n+1)}\ , \qquad (5)$$

where $i$ is the location value of the nodes in the ordered trust table, $n$ is the *set(neighbor)* node number. $Truth_{recommendation}$ gives the source node an advisory opinion and the source node's final decision about $Truth_{direct}$ and $Truth_{recommendation}$, as shown in (6). The weighted value of $Truth_{direct}$ is greater than $Truth_{recommendation}$, and $w_3 + w_4 = 1$

$$Truth_{synthesis} = w_3 \times Truth_{direct} + w_4 \times \times Truth_{recommendation} \qquad (6)$$

The most reliable source node has the largest weighted value, which is $\frac{2}{n+1}$. The value of the difference between the weighted value of two adjacent nodes in the trust table is $\frac{2}{n(n+1)}$. When n is in [2, 3…, n], the corresponding largest weighted value is in [ $\frac{2}{2+1}$ , $\frac{2}{3+1}$ …,$\frac{2}{n+1}$ ]. The larger n, the smaller the weighted value of each node.

It is known that there are 3 types of data analysis in the Fog layer. The 1st type recovers faulty nodes and detects attacks on hidden data based on trust tables, historical sensor data, and network topology. The 2nd type checks for malicious sensor network nodes based on trust tables, recommendation tables, historical sensor data, and network topology. The 3rd type concerns edge node trust, which is based on trust tables and sensor data correlations.

Whole sensor nodes send the change values of the trust table together with the sensor data to the fog layer within a certain period. The source node sends the recommendation table together with the sensor data to the Fog layer after completing the recommendation trust calculation. The Fog layer periodically analyzes the global trust state of each node and determines whether there are misjudged nodes and hidden data attacks. In addition, we can predict a certain state of the network, in particular, network load, residual energy of nodes, etc.

These nodes are harder to find because they behave normally when communicating with other nodes. Certain data correlation phenomena are observed within the same area or cluster. For example, sensor data from multiple nodes in the same geographic location are similar, and sensor data from multiple nodes moving together have a trajectory correlation. The Fog layer can simultaneously process sensor data from multiple nodes and analyze the presence of incorrect nodes with some indicators of data correlation phenomenon, such as change trend, and similar trajectory. We perform a multipath operation to analyze sensor data from different nodes. Here we mainly consider nodes that implement the same function.

The structure of the process [12] is shown in Fig. 5. Basically, nodes that implement the same function in the same geographical location are considered, and described by the system (7):

$$Array = \begin{matrix} Count_{crest} \cup degree, \frac{X2_i - X1_i}{Y2_i - Y1_i} > 0 \\ Count_{trough} \cup degree, \frac{X2_i - X1_i}{Y2_i - Y1_i} > 0 \end{matrix} \quad (7)$$

An array is applied to store the vertex, west and degree. $Count_{crest}$ indicates the crest of the sensor data curves, which is written as 1. $Count_{trough}$ indicates the trough of the sensor data curves, which is written as -1. The degree is the value of the difference between two adjacent sensor data. Peak/trough detections are continuous negative/positive values, and continuous peak/trough continues when the sensor data change value is zero after the peak/trough is recorded. At each point in time, the Array writes the state value (1, -1, 0) and the degree value to the Array. Fog/Edge nodes have less communication with other nodes relative to internal nodes. In the sensor network, we set a short period for Fog/Edge nodes. In addition, the Fog layer scans and analyzes the true state of the Fog/Edge nodes in a short period.

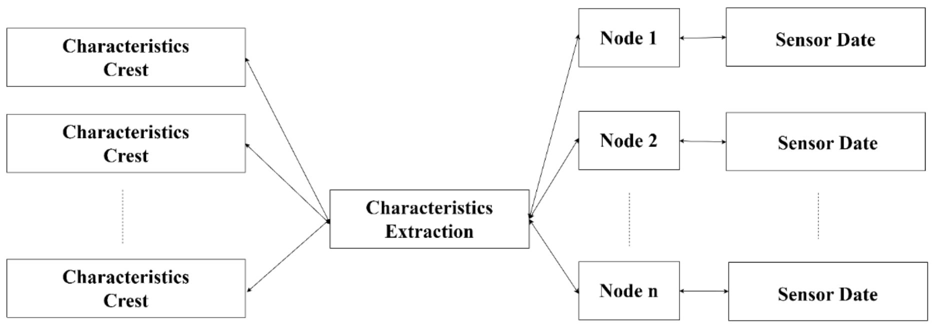


*Fig.5. Structure of sensor data analysis*

The trust relationship between CSP and SSP is divided into two parts. The first is the trust relationship between CSP and SSP, and the second is the trust relationship between SSP and CSP. Cloud service providers (CSPs), expect data from sensor service providers (SSPs) to meet certain requirements such as timeliness, integrity, and accuracy. However, the service user may not demand service providers to meet requirements. It means that service providers only have to meet the service user's specific requirements. Therefore, there should be some recommendation mechanisms to find CSPs that offer good services in specific aspects. Fog computing can handle these problems satisfactorily. A third party based on Fog computing can provide the reliability of a three-part SSP as shown in equation (8):

$$Truth_{SSPs} = w_5 Truth_{service} + \\ + w_6 Truth_{sensor\ network} + w_7 Truth_{CSPs}, \quad (8)$$

where $Truth_{service}$ is the truth value for service parameters. Before a service transaction, the SSP and CSP agree on service parameter standards. The Fog layer then monitors these service parameters during a real-time transaction and compares these service parameters with standard values. If the value of the monitored service parameter is in the acceptable range, the entry for that parameter is 1, otherwise 0. Finally, the $Truth_{service}$ is calculated from the various weighted values of the service parameters. $Truth_{sensor}$ network is the truth value for the sensor network and is appropriate to the exception information records of the sensor network. If a sensor network has more exceptions in a transaction, it was assigned a lower value. $Truth_{CSP}$ is a type of truth value that is calculated according to the information record of other CSPs in the Fog layer.

There are 2 steps for deciding on the choice of CSP: $R_{general}$ and $R_{similar}$. Some CSPs are short-listed candidates whose service records contain requested service parameters. $R_{general}$ assigns selected CSPs to different sets, which are classified by the number of redundant parameters. Then, the truth value of each SSP is separately computed in different sets. Finally, some anomalous CSPs are excluded from the candidate list according to the rule of changing the confidence value between different sets. $R_{similar}$ is an optimal selection strategy based on the principles available for service selection. $Truth_{CSP}$ is calculated using these selected CSPs. Here $w_5$, $w_6$, and $w_7$ are 3 weighted values that are set during initialization according to different requirements, and $w_5 + w_6 + w_7 = 1$. The value of $Truth_{service}$ $Truth_{sensornetwork}$ and $Truth_{CSP}$ is between 0 and 1.

SSPs expect services provided by CSPs to meet certain criteria such as reliability, security, convenience, controllability, and stability. These indicators are important for SSP to establish veracity in CSP. Fog level can monitor these indicators in real-time. The true relationship between SSP and CSP contains two components, as shown in formula (9):

$$Truth_{SSPs} = w_8 Truth_{service1} + w_9 Truth_{SSPs}\,, \qquad (9)$$

where $Truth_{service1}$ is the truth value of CSP service parameters similar to $Truth_{service}$. $Truth_{SSP}$ is computed with some selected SSPs whose selection process is similar to $Trutht_{CSP}$. There are several databases in the Fog layer for storing time-based maintenance records. $w_8$ and $w_9$ are two weight values that are set during initialization according to different requirements, and $w_8 + w_9 = 1$. The $Truth_{service}$ and $Truth_{SSP}$ values are in the range of 0 to 1.

## 7. Results of Issue and Discussion

Research of the proposed modified method of monitoring the truth of network data during resource allocation at the level of Fog in multi-sensor systems was carried out using the MATLAB R2016b application package. Eight cluster structures with more than 300 nodes randomly deployed at the level of WSNs are presented. Each cluster is divided into 4 layers, where the outer layer has more nodes than the inner layer. In each clustering structure, cluster cores can receive sensor data packets from eight nodes simultaneously. The maximum delay time from the WSN to the fog layer is set as 10 communication cycles. The described parameters are listed in Table 1.

**Table 1.** Simulation parameters

| Parameters | Values |
|---|---|
| Network protocol | The Ladder Diffusion Algorithm |
| The number of Clusters | 8 |
| The number of Cluster Heads | 46 |
| The number of Cluster Nodes | 350 |
| The number of levels | 4 |
| The maximum delay | 10 |

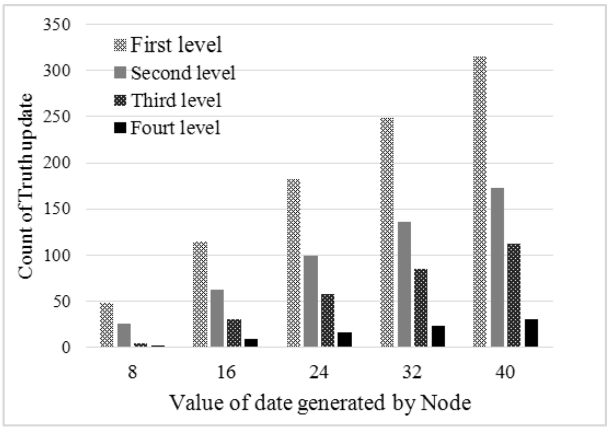
As mentioned above, there are 2 types of trust mechanisms: periodic and non-periodic updates. For aperiodic updating, nodes update the trust state of their neighboring nodes when abnormal behavior is detected. Here there are some shortcomings in aperiodic updating, for example, not enough attention is paid to edge nodes, and no updated trust states of nodes. The results of research for four levels of the aperiodic update are shown in fig. 6, a. Aperiodic update cannot detect malicious nodes in time. For periodic updating (Fig. 6b), nodes update the trust values of their neighboring nodes after the end of the period. There are also some drawbacks to periodic updating, such as much memory and computing resources, reducing network performance, etc. The proposed design is reliance on periodic updating (Fig. 6c). We have established that the truth update cycle at the outer layer is the same as the periodic update, which can be found in Fig. 6(b) and fig. 6(c). We can extend the ground-truth update cycle internally by using Fog computations, which can avoid additional resource costs for periodic detection, as shown in Fig. 6(c).

Three experimental results are presented, considering the number of truth updates at each level. The load on the network increases based on the number of nodes that generate data. Fig. 6(d) shows the total number of truth updates in a shorter test time. From these experimental results, we can obtain the following information: 1) for the non-periodic update, the amount of truth update time increases gradually with more random nodes selected to transmit data; 2) a steady state is maintained for the periodic update, but there is a slight reduction from 8 to 40 on the x-axis, the reason being that the direct truth update reduces the number of periodic updates; 3) for a specific design, we can get more advantage when the update cycle is extended. When the network is congested, the network bandwidth decreases and the truth update time increases due to frequent routing failures. Compared with the periodic update, proposed in this work design can save network energy and maintain network performance by reducing the number of periodic updates.

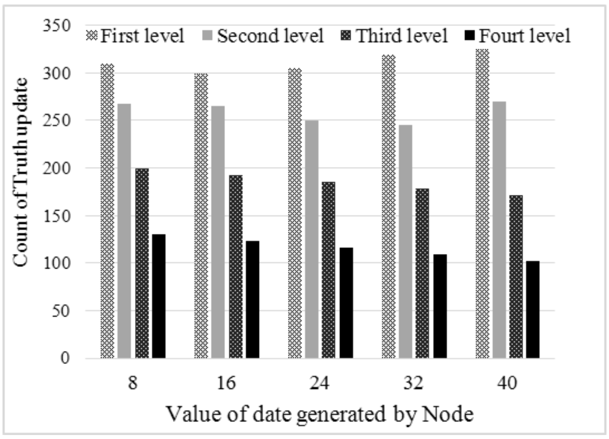
There is no need for frequent updates since internal attacks occur at a certain point in time and frequent updates take more transmission and computing resources. We compare the detection rate of malicious nodes between our design and a periodic update.

Defective nodes can be detected relying on 2 parameters: at the WSN level and the Fog level. Since the delay time at the Fog layer is longer than that at the WSN layer, the detection of malicious nodes at the Fog layer is introduced as an auxiliary detection. The speed of detection of the periodic update is shown in Fig. 7(a), and Fig. 7(b) shows the detection rate of our design. In the experiment, we separately placed corrupted nodes at different levels of the network during initialization, which spend more time than during the operation of the mechanism.

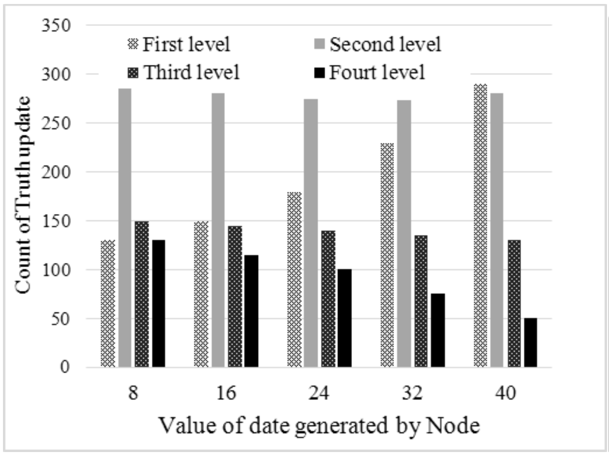
The experimental results show that the detection rate of malicious nodes increases, except for the outer layer, because the true state of the nodes is updated more often when the network load is growing. In Fig.7(c), we randomly placed the corrupt nodes in 4 levels, indicating a more intuitive downward trend. Despite some latency issues with speed detection, we can take advantage of Fog computing to get the full true state of the network through data analysis, such as hidden data attack detection and anomaly detection.



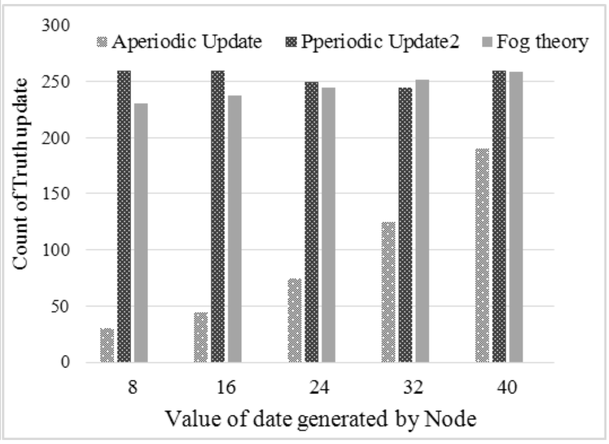
Aperiodic Update (a)



Periodic Update (b)



Modified Design (c)

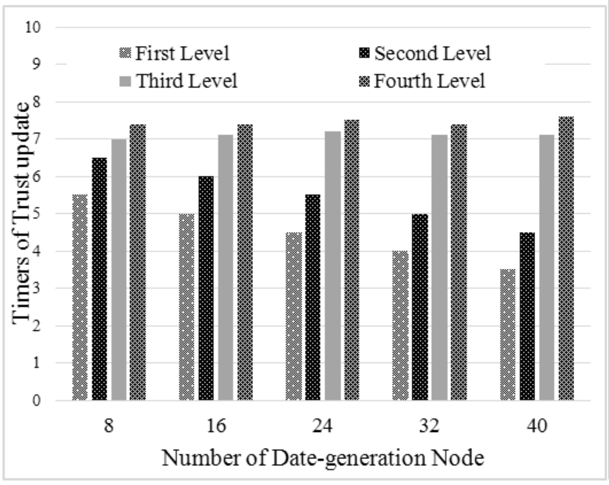


After Data Processing (d)

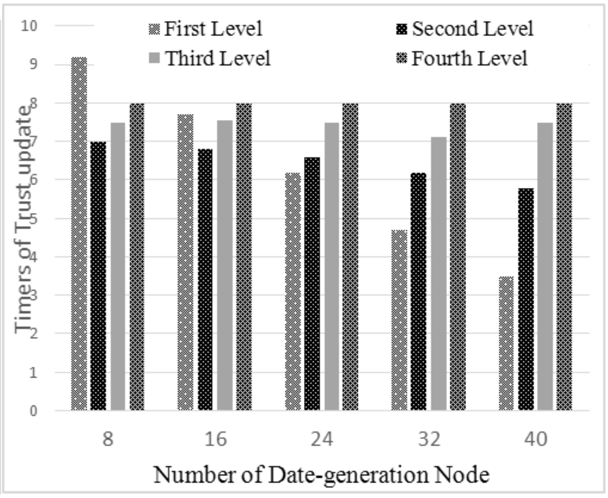
*Fig.6. Comparison of three schemes of truth updates at different levels*

**Table 2.** The information record of SSPs in the Fog level

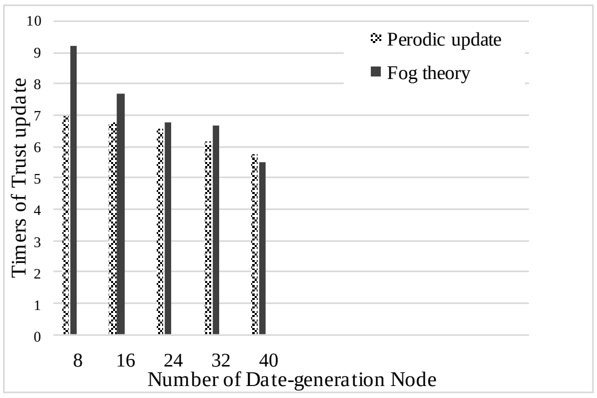
| CSP | Service requirement | Interaction Record | Recommendation Record | Number of accepted | Check-in time |
|---|---|---|---|---|---|
| CSP1 | Integrality precision | SSP1 (60)<br>SSP2 (90)<br>SSP3 (96) | CSP2 CSP4 | 96/97 | 15 |
| CSP2 | Integrality precision | SSP2 (90)<br>SSP3 (94)<br>SSP4 (93) | CSP1 CSP4 | 60/80 | 9 |
| CSP3 | No tampering integrality precision | SSP1 (86) | CSP1 CSP2 CSP4 | 26/29 | 7 |
| CSP4 | No tampering integrality timeliness precision | SSP1 (74)<br>SSP2 (72)<br>SSP3 (85) | CSP3 | 72/73 | 2 |



Periodic Update (a)



Modified Design (b)



The result after Data Processing (c)

*Fig.7. Comparison of two schemes of truth updates at different levels*

For $Truth_{service}$, this can be obtained by comparing real-time values and standard values. For the $Truth_{sensor}$ network, it can be calculated based on some exception records that the Fog layer detects and records. Before calculating the $Truth_{SSP}$, there are some scoring mechanisms, for example, the score is reduced by one level when the service parameters are one less than the required service parameters. Table 2 lists some important parameters for the example. A service request stores a record of the service parameters of the CSP that requested those parameters in a previous transaction. The interaction record stores the SSPs and their truth values that provided the CSP data. A referral record is maintained by CSPs who have received services from a CSP. The number of accepted requests corresponds to the number of accepted recommendations. The registration time preserves the CSP existence time at the fog level.

A service request stores CSP service parameter records. The fuzzy layer selects service records that are within an acceptable time range, as shown in Table 2, where integrity and timeliness depend on different sizes of service requirement sets, such as set 1 (CSP1, CSP2), set 2: (CSP3), set 3: (CSP4). SSP1 is without obstacles, integrity), SSP2 - integrity, timeliness, SSP3 - without obstacles, integrity, accuracy, SSP4 is integrity, timeliness. When calculating $R_{general}$, the mean confidence value of each SSP is calculated in different sets, for example [SSP1(60), SSP2(90), SSP3(96)] in set 1, [SSP1(86), SSP3(94), SSP4 (82) ] in set 2 and [SSP1(74), SSP2(72), SSP3(85)] in set 3. The true value of an individual SSP in the smaller set must be greater than or equal to the value in the larger set. So, with this rule, we can find some abnormal scores, for example, CSP1 may be a wrong choice. After removing the non-standard recommender, optimal options such as familiarity, popularity, and risk are considered. Referral records, SSP location, and reputation affect referrals. Popularity refers to the number of referrals received by a single CSP. The risk focuses on whether there were any losses if a new recommender is chosen.

## 8. Conclusions

The Fog-based trust method has been modified to compensate for existing shortcomings and solve consumer problems such as data overload and insufficient network resources for real-time processing and implementation in systems with a multi-sensor configuration.

It is shown that trust in behavior between nodes is established at the level of wireless sensor networks. The trust in the data of nodes and objects is installed in the Fog layer.

The consideration shows that we can monitor the trust status of the entire multi-sensor network, detect data attacks, and recover incorrectly evaluated nodes in the case of increased network load, in a situation when data needs to be parsed in real-time. Compared with the periodic update, our design can save network energy and maintain network performance by reducing the number of periodic updates. The Fog layer can be constructed as a reliable third party. Experimental results show that the modified trust method has certain advantages in ensuring the state of trust of boundary nodes and the network, restoring nodes with an incorrect assessment.

## 9. Gratitude

The authors express their gratitude to the Ministry of Education and Science of Ukraine for the support (Project for young scientists No 0122U000807, 2022).

## 10. Conflict of Interest

The authors state that there are no financial or other potential conflicts regarding this work.

## References

[1] D. Xu, M. Li, W. He, Sh. Li, Internet of things in industries: A survey [J]. IEEE Transactions on industrial informatics 10(4), 2233-2243 (2014). DOI:10.1109/TII.2014.2300753. https://ieeexplore.ieee.org/abstract/document/6714496

[2] Q. Wang, X. Zhu, Y. Ni, L. Gu, H. Zhu, Blockchain for the IoT and industrial IoT: A review [J]. Internet of Things 10, 100081 (2020). DOI:10.1016/j.iot.2019.100081. https://www.sciencedirect.com/science/article/abs/pii/S254266051930085X

[3] L. Tseng, L. Wong, S. Otoum, M. Aloqaily, J. B. Othman, Blockchain for managing heterogeneous internet of things: A perspective architecture. IEEE network 34(1), 16-23 (2020). DOI: 10.1109/MNET.001.1900103. https://ieeexplore.ieee.org/abstract/document/8977441

[4] R. Basir, S. Qaisar, M. Ali, M. Aldwairi, M. I. Ashraf, A. Mahmood, M. Gidlund, Fog computing enabling industrial internet of things: State-of-the-art and research challenges [J]. Sensors 19(21), 4807 (2019). DOI: 10.3390/s19214807. https://www.mdpi.com/ 1424-8220/19/21/4807

[5] P. O'Donovan, K. Bruton, C. Gallagher, D. O'Sullivan A Fog computing industrial cyber-physical system for embedded low-latency machine learning Industry 4.0 applications [J]. Manufacturing letters 15, 139-142 (2018). DOI: 10.1016/j.mfglet.2018.01.005. https://www.sciencedirect.com/science/article/abs/pii/S2213846318300087

[6] R. Diachok, H. Klym, Data cleaning method in wireless sensor-based on intelligence technology [J]. Measuring Equipment and Metrology 83(2), 5-10 (2022). DOI: 10.23939/istcmtm2022.02.005. https://science.lpnu.ua/uk/istcmtm/vsi-vypusky/vypusk-83-no2-2022/data-cleaning-method-wireless-sensor-based-intelligence

[7] P. Hu, S. Dhelim, H. Ning, T. Qiu, Survey on fog computing: architecture, key technologies, applications and open issues [J]. Journal of network and computer applications 98, 27-42 (2017). DOI: 10.1016/j.jnca.2017.09.002. https://www.sciencedirect.com/science/article/abs/pii/S1084804517302953

[8] I. Al Ridhawi, M. Aloqaily, A. Boukerche, Comparing fog solutions for energy efficiency in wireless networks: Challenges and opportunities [J]. IEEE Wireless Communications 26(6), 80-86 (2019). DOI: 10.1109/MWC.001.1900077. https://ieeexplore.ieee.org/abstract/document/8938188

[9] O. Bouachir, M. Aloqaily, L. Tseng, A. Boukerche, Blockchain and fog computing for cyber-physical systems: The case of smart industry [J]. Computer, 53(9), 36-45 (2020). DOI: 10.1109/MC.2020.2996212. https://ieeexplore.ieee.org/document/9187468

[10] M. Muneeb, K.M. Ko, Y.H. Park, A Fog computing architecture with multi-layer for computing-intensive IoT applications [J]. Applied Sciences 11(24), 11585 (2021). DOI: 10.3390/app112411585. https://www.mdpi.com/2076-3417/11/24/11585

[11] A. Naouri, H. Wu, N.A. Nouri, S. Dhelim, H. Ning, A novel framework for mobile-edge computing by optimizing task offloading [J]. IEEE Internet of Things Journal 8(16), 13065-13076 (2021). DOI: 10.1109/JIOT.2021.3064225. https://ieeexplore.ieee.org/abstract/document/9372288

[12] T. Wang, G. Zhang, M. Z. A., Bhuiyan, A. Liu, W. Jia, M. Xie, A novel trust mechanism based on fog computing in sensor–cloud system [J]. Future Generation Computer Systems 109, 573-582 (2020). DOI: 10.1016/j.future.2018.05.049. https://www. sciencedirect. com/science/article/abs/pii/S0167739X17323658