Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 6, 2026

COGNITIVE SENTINEL: DYNAMIC DEFENSE AGAINST MALICIOUS FOG NODES IN EVOLVING FOG -2- FOG COLLABORATIVE MODEL

[1]Rimsha Ehsan, [2]Imran Rashid, [3]Danish Manzoor
[1]engr1997re@gmail.com, [2]irashid@mcs.edu.pk, [3]danishmanzoor161@gmail.com

DOI:https://doi.org/10.5281/zenodo.20615253

Keywords
Fog nodes, offloading, reputation, iFogSim, COMITMENT mode

Article History
Received: 18 May, 2026
Accepted: 07 June, 2026
Published: 09 June, 2026

Corresponding Author: *

Abstract
IoT applications with stern time constraints often demand very-low latency, and meeting the Quality of Service (QoS) requirements proves challenging with conventional cloud computing. To mitigate this challenge, Cisco introduced Fog Computing in 2015. However, the ever-evolving nature of the fog computing environment introduces several security challenges. Compounding the issue, fog nodes are often deployed by various developers with varying security guidelines. Collaboration amongst the fog nodes, especially in data offloading scenarios, presents security concerns that are currently unexplored. The existing work on security in fog computing is limited, and conventional cryptography strategies are ill-suited for detecting networks having malicious nodes. Consequently, the reputation of IoT services is threatened by presence of malicious fog nodes and this compromises user's privacy. This research paper advocates for a trust-based model, aiming to identify the maximum trustworthy node to offload tasks while separating any malicious fog nodes within the network. By doing so, the proposed method enhances the security of network and elevates overall Quality of Service (QoS).

1. Introduction

The development of the IoT allows emergence of multiple million micro devices emitting tons of information and requiring huge data processing power. However, a data set of such a size is already a challenge in and of itself; the cloud computing services try to address these problems by looking at QoS, efficiency and latency. Time-sensitive apps and those which require a certain QoS level and even a low latency, will be prevented from getting the most satisfactory performance through the traditional cloud computing approach. In 2015, Cisco declared fog computing to be a suitable futuristic model to embrace the difficulties encountered. This cloud extension provides the same functionalities as a cloud, but better-quality services which bring support for issues like latency, mobility and so on. Fog computing as an edge computing platform is applicable to a wide range of things: IoT, 5G networks, VR, just to name a few. The heart of a fog computing system is composed by the fog node. On the flip side, a fog node is resourceful on one side but limited on the other side when it comes to high computation tasks like smart data transportation, image/ data processing, and VR. To overcome these issues needs great data manipulation, and fog nodes within a fog layer need to share their knowledge, providing user with enhanced data supervision.

Fog computing contains two main ways for devices to interact: (a) Centralized mode: a common controller, like a traffic surveillance gadget, focuses communication throughout the entire fog network. (b) Distributed mode: additionally, the fog devices team up individually, without a single governing organ. While these offer advantages, this collaboration can increase security and privacy risks. Contrarily, fog computing is comparatively better in terms of reliability than cloud computing because the data is handled locally by fog nodes at the end-user end, making the remote connection less reliant on the availability of network connectivity. Exchange of fog nodes for offloading data brings along security dangers where the offloaded data can become a victim by way of malicious fog nodes.

The unique characteristics of the fog environment are found in the deployment with different vendors who work in accordance with varying registries that are tailor made to different security policies, which causes security and integrity issues. Moreover, cryptographic solutions do not possess adequate resources to handle the threats that are predominantly internal and, thus, they include topology attacks that fall under anonymity. Most of the network traffic is untraced and it is difficult to find a fake device corresponding to a recognized user ID used to run the system within the network [1].

In fog computing, despite the criticality of node security, minimal research has been undertaken in this domain. Security lapses are significant as even secure data becomes vulnerable when nodes are compromised during the task offloading process. Certain applications demand heightened security for sensitive data, posing challenges for fog nodes to provide requisite security. There aren't many strategies now in use to build confidence between fog nodes and overcome offloading concerns; most trust-based techniques only handle internal attacks and don't address cold-start issues.

1.2 Problem Statement

Developing fog computing to address latency issues for delay-sensitive applications, encounters challenges when overloaded fog nodes struggle to handle incoming requests. Applications that are sensitive to delays suffer from this constraint. Overloaded fog nodes transfer tasks to other fog nodes in the fog domain to address this. But not every fog domain node can be relied upon. Confidential data may be compromised or stolen if outsourced data is offloaded to a rogue fog node. The lack of extensive research on fog node security, despite its pivotal role in securing data, remains a significant concern.

Existing trust-based schemes fall short, lacking efficiency and neglecting cold-start problems. In fog computing, node trust does not utilize such methods mainly for task offloading. The development of an effective method for recognizing trustworthy fog nodes and detecting malicious fog nodes in fog environment is crucial.

1.3 Gap and Objective

From a responsibility standpoint, unlike other dispersed networks, such as wireless sensor networks (WSN), which are managed by a single provider, fog networks are composed of multiple providers, each of which has its own set of security standards for its fog nodes [2]. Since that

the fog devices function at the edge of networks, the cloud-based security strategy might not be appropriate for fog computing environments, facing unique threats absent in well-managed clouds [3]. While trust approaches are common in other distributed networks, their direct applicability to fog networks remains uncertain.

1.4 The Cold-Start Problem

Whenever a node is introduced in the network without prior interactions or trust value during offloading, a problem occurs known as cold-start problem. Figure 1.1 illustrates a fog domain network, the overloaded node f0 transfers the data to other nodes that are nearby in the domain. When a third type of neighboring node, with no past interaction history, is selected for offloading, calculating the node's trust value becomes challenging—a scenario known as the cold-start problem.

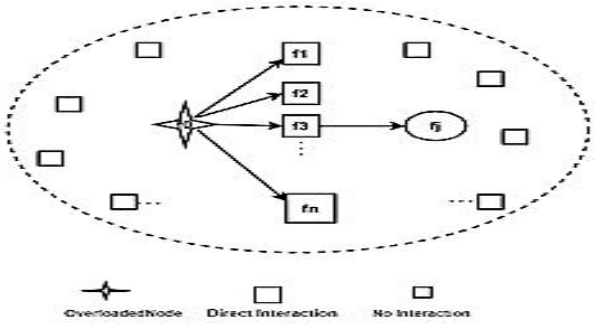


Figure 1.1: Cold-start problem

Current trust-based methods determine trust based on the assumption that every device in the fog network is known. However, none of these approaches addresses the handling of any additional node whose trust value is unknown, joins the network. Cold-start problem remains largely unexplored in fog computing, with limited research in other domains [5]. Existing approaches defining trust mechanisms for cloud-based networks, vehicle-based networks, and wireless or sensor-based networks lack direct applicability to fog based IoT networks.

2. Literature Review

2.1 Related Research

Paper published in 2019 displayed a faith in the fog-based model, suggesting application of a General Trust Model. In spite of time, money and other resources being invested to create the model, it was still in the theoretical phase during the period of the publication itself. The research paper equally acknowledges the above-mentioned limitations and points out the need of empirical testing and practical implementation to prove the successful role of the General Trust Model in fog computing.

The first paper that was published in 2020 also dealt with the problems of end-to-end latency by introducing the Context Aware Trust Evaluation Model (CATEM). Yet, it mentions failures while discussing the designed system. The paper points out that the system addresses Quality of Service (QoS), but it has a single point of failure. This might not only create the impression that the architecture is vulnerable but may as well lead to uncertainty about how reliable the system is.

In [4], the author proposed to partially offload fog computing that consists of the instances where a fog node can be relieved from the execution of the demanding task on another fog node augmenting its effect and life. It has many disadvantages such as increased energy consumption and additional communication overhead. Furthermore, data and fog node security lapses are possible during the offloading process.

The author in [2], recommends a Two-way Trust Management system (TTM) which is logic based and tailored for fog computing. This system benefits both service requesters and providers, allowing verification of a provider's capability to deliver secure and reliable services, and vice versa. The primary focus is on identifying malicious fog servers and clients, leading to the isolation of detected bad nodes. The trust-based model determines fog node trust values based on social

information and quality of service (QoS) and is distributed as well as event-based. However, it has notable drawbacks, being specialized for traffic application requests and lacking a solution for the cold start problem.

In 2022, a publication aimed to ensure the reliability and efficacy of Social Internet of Things (SIoT) introduced Context-Dependent Trust management method (ConTrust). The paper revealed a limitation: ConTrust had not been implemented on real physical nodes. Despite this, the study recognized latency considerations and the importance of node security.

The 2021 study proposed the Fog Computing Architecture for Load Balancing (FOCALB) as the prototype and utilized iFogSim and Eclipse as tools for implementing the system. While optimizing load balancing, the paper acknowledged a limitation: the lack of explicit consideration for the security and privacy of fog nodes.

In 2022, the authors of the paper focused on designing the fog computing-based trust and reputation system with the help of a recommender system working on Collaborative Filtering approach. The research underlined a weakness which had to do with the trust and reputation systems reinforcement of the existing (biasing) rather than bringing fairness and accuracy. QoS, latency, and node security were discussed, but was no concentrated effort on the "cold start problem", directing attention to the possible areas for future studies.

Table 1.1: *Research papers comparison*

Paper	Year	Objective	Mechanism	Limitations
[5]	2019	Preserves user trust in fog computing model	General Trust Model	Proposal only, no implementation
[6]	2020	Reduction in end-to-end latency	Context aware trust evaluation model	Issue of SPF (Single point of failure)
[4]	2019	Fog Node based energy consumption to minimize delay in task processing	Partial offloading (Energy and delay efficient)	The security of node and data not considered
[2]	2020	Efficient distribution of load and selection of trusted service provider	Two-way trust management model	Issue of further overhead communication
[7]	2022	To check the reliance and effectiveness of SIoT for job selection and allocation	Context dependent trust management technique (ConTrust)	Not implemented on real physical nodes
[8]	2021	Fog Computing Architecture of Load Balancing (FOCALB)	iFogSim and Eclipse	Security and privacy of fog node
[9]	2022	Fog computing-based trust and reputation system	Recommender System based on Collaborative filtering	The issue of biasing

3. Methodology

3.1 System Overview

Several fog nodes that are dispersed over several physical locations and are in direct communication with one another make up typical fog layers. Fog nodes are owned and operated by providers, who can have several nodes under their ownership. There is no central controller to facilitate offloading and resource sharing. In fog networks, trust is decentralized, and there is no single authority to determine which nodes are trustworthy. In the fog network, every node keeps track of its neighbors' trusted list and regularly assesses their trust rating. Three components make up our reputation base trust model.

Definitions

The following parts define our trust model in formal terms.

Fog Network = {FN1, FN2, FN3...FNn} refers to a network with n fog nodes. R= {R1, R2, R3...Rn} is a collection of reputations. When a fog node in a fog network becomes overloaded f over, it must transmit its data to a different node within its domain. The receiver node, f_{rec}, represents the best fog node that is presently available to receive offloading data.

Each fog node in our model updates the Trust table (TT). As shown in Table 3.1, the TT contains a list of nearby fog nodes (F_n), as well as information on their providers, trust levels, and so on. Our system distinguishes between two categories of service requests: heavy requests (HR) and light requests (LR). Our methodology assesses trust between fog nodes using two metrics: quality of protection (QoP) and quality of service (QoS).

Table 3.1: *Trust and Reputation table for fog nodes*

Trust and Reputation Table

Nodes	Reputation	Trust Values
FN1	R1	T1
FN2	R2	T2
FN3	R3	T3
-	-	-
FNn	Rn	Tn

3.2 Reputation Based Trust Model (RBT)

To identify the optimal fog node for collaboration and data offloading, the suggested reputation-based trust model assists the fog nodes. When nodes work together, there are three scenarios.

i) Direct communication with the node.
ii) Indirect communication with the node.
iii) No communication.

In our method for determining the fog node's trust score, the modules and interaction timeline are represented by the number 1. Everything begins when a newly connected node tries to offload data to another overburdened node on the network. The overloading node can choose to unload data on the current node, which is one of their two possibilities. The nodes that have interacted with overloaded nodes in the past are the ones that are now in use. Overloading node unload data on the new node is the second option. Our technique comprises of following modules:

- Assessment based on trust
- Overall trust calculation
- Recommendation based on trust

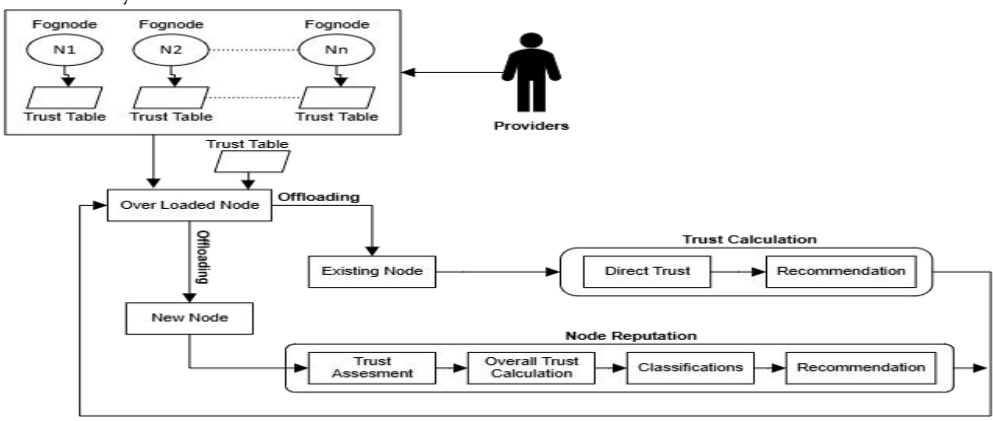


Figure3.1: Elements of the proposed trust model

3.2.1 Assessment based on trust

The first part investigates the communication nature that occurs amongst the receiving node and overloaded nodes. Next, it assesses the credibility of the node depending on the communication type. A triangular trust computation is implemented.

- Direct

- Indirect
- Reputation-based

3.2.2 Reputation Based Trust (RBT)

Every fog node in our notion has a provider, and a single provider may have several nodes. The experience satisfaction score is recorded by every node in the fog layer for every contact with the partner node using QoS and QoP. There are two types of interaction: direct and indirect. However, if a fresh node adjoins the network and has never interacted with anybody before, then the node trust of the existing nodes is used to assess the reputation-based trust. Two instances are present.
A. New node using the current neighbor nodes.
B. A new neighbor node with a new node.

$$\lambda_{(i,j)} = \lambda_{k(i,j)} + \lambda_{prev} \quad (3.1)$$

Case-I: New node with existing neighbor node

When there is no previous experience of offloading tasks between the node f_L node and the f_{rec} node, in the fog-2-fog cooperation paradigm the f_{rec} node becomes a new node for the f_{over}. Based on reputation base trust, the f_{over} node determines the new node's trust score. The basis for assessing reputation-based trust is the confidence of current provider nodes. Based on QoS and QoP, our model states that the experience satisfaction (ES) score of the neighboring nodes following each interaction is preserved by each fog node. Moreover, the trust tables, which are included in fog nodes, also hold data on nearby nodes, their suppliers, and the trust scores of those nodes. Let us say that fog node fa wishes to transfer data to fog node fb. Overloaded fog nodes will check the trust table's list of nearby nodes and remove those that share a provider with the newly added node. Assume that the overloaded node keeps trust scores for these nodes since it has direct experience with them. In such case, these nodes calculate total node trust alongside level L0 nodes, resulting in level zero L0 nodes. When level L0 nodes are overloaded with requests from other providers' surrounding nodes, they take over and assist with the trust computation. These types of nodes are known as level one (L1) nodes. The f_L node uses formulae 3.2 for the total node trust score calculation based on the new node's current level zero L_0 nodes. To calculate the final trust $\lambda_{(i,j)}$, the history experience λ_{prev} in addition to the recent interaction $\lambda_{k(i,j)}$ are combined. Where, i stands for the i^{th} fog nodes, and j for the j^{th} fog nodes. In our model based on Level of Trust (LoT) we divide the recommender into two categories: highly trusted and less trusted, based on the x.

Table 3.2: *Recommender types*

Recommender types	
LoT	Type
0-0.4	Malicious
0.5-1	Trusted

Equation 3.2 is used to determine the new node trust based on neighboring nodes.

$$\lambda_r = 1/n \sum_{i=1}^{n} f_i \quad (3.2)$$

Where n denotes the total number of received proposals.
Equation (3.3) calculates the reputation of the node

$$\gamma(i,j) = \sum_{k \in Nj} \lambda(i,j) * [I(k,j) / \sum_{k` \in Nj} I(k`,j)] \quad (3.3)$$

When the threshold value for reputation-based trust is not met, recommendations are accepted or rejected based on user conscience. This implies that if the recommendation is accepted, the users agree to offload data to a less trusted node; if not, the data is not offloaded. Precisely as described in equation 3.3, recommendations are accepted and rejected.

$$R1 = \begin{cases} 1, & \text{if consciences} = \text{Yes} \\ 0, & \text{otherwise} \end{cases} \quad (3.4)$$

When **R** has a value of 1, suggesting acceptance and 0 suggesting rejection.
The total node trust is given by equation (3.5)

$$\mu(i,j) = \lambda(i,j) + \gamma(i.j) \quad (3.5)$$

The following outlines the steps in Algorithm 1 for reputation-based trust calculations:

Step1: Check node type

Verify the type of node; if the node is overloaded, review its history of interactions with the chosen node.

Step2: Check reputation

Verify that the new node reputation is on the list of current nodes.

Step3: Trusted existing nodes

Retrieve the list of current nodes and their trust score from the same source, then compute the total trust of all nodes.

Step4: Assess credibility

Provided node reliability is verified based on the trust level of all active nodes.

Step5: Take Decision

When R has a value of 1, suggesting acceptance and 0 suggesting rejection.

Algorithm 1: Trust Calculation

Input: *Fog Node$_a$* (f_a); *Fog Node$_b$* (f_b)

Parameters: *Fog List* (*FL*); *Fog Provider* (*P*); *Service Type* (*Hr*, *Lr*),

R← *Reputation*

Outcome: Trust score of a new node reputation initialization;

Method 1: Get the list of the nodes by

F_L=list (f_n) ← *gets Nodes list* (*Out: (Service type, provider)*)

Node$_e$ ← *existing node list*

Node$_n$ ← *New node*

for $Fn \square FL$ **do**

  if *Node$_n$* □ *Node$_e$* **then**

    Case1 **if** *Direct interaction exists* **then**

$\lambda\ (i, j) = \lambda_{k\ (i, j)} + \lambda$ prev

end

Case2 **if** *No direct interaction exists* **then if** *fn trusted* **then**

$\lambda_r = 1/n \sum^{n} fi$

end

end

Case3 Calculate node reputation

$$\gamma\ (i,j) = \sum_{k \in Nj} \lambda\ (i, j) * [\ I(k,j) / \sum_{k` \in Nj} I(k`,j)]$$

Calculate reputation based trust

$$\mu\ (i, j) = \lambda\ (i, j) + \gamma(i.j)$$

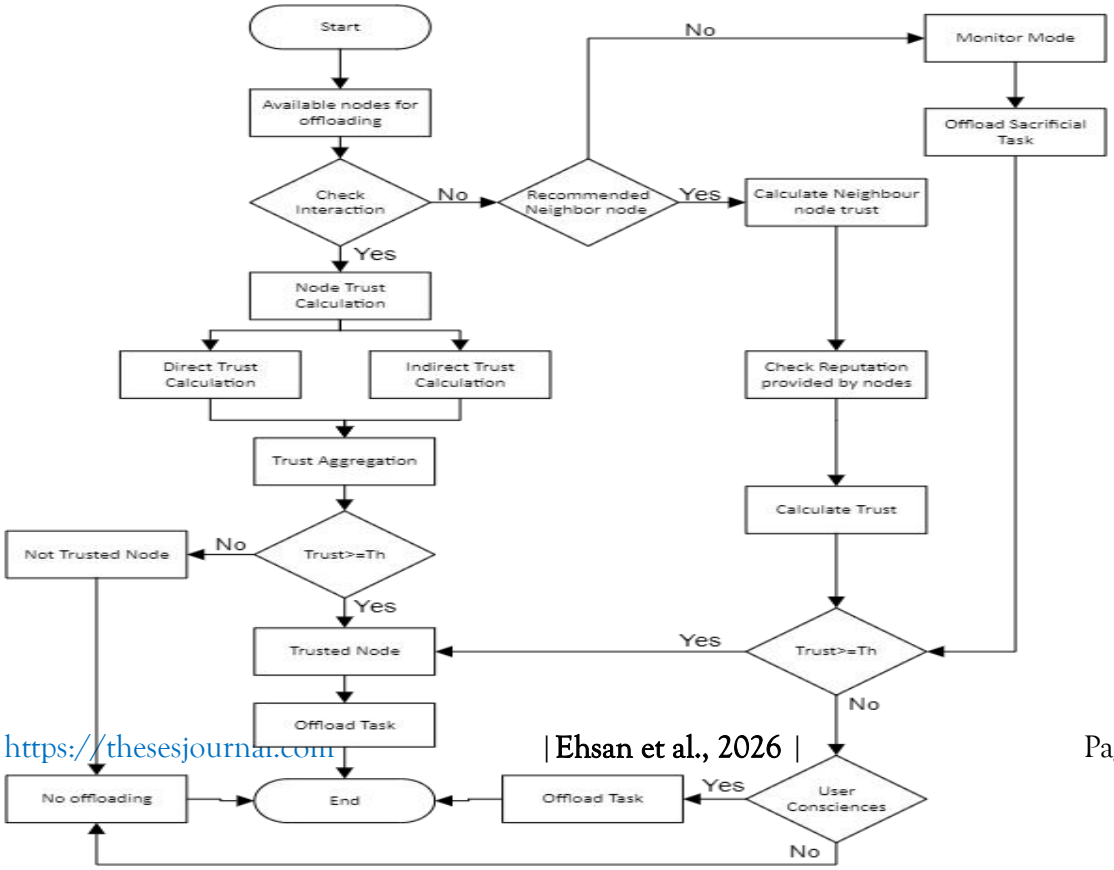
end

```
Case 4 if Reputation based trust < Th then

    if User consciences = yes then
    |  set Flag (R) = 1;
    end

    end

    else
    |  λk= 0.5;
    |
    end

    return λk

end
```

3.5 Flow Diagram

Figure 3.2 depicts the general flow pattern of our methodology. It begins when a node in the network experiences overload and is unable to process requests by the deadline. In order to determine whether overloaded nodes have already interacted with the available node, first check the available node in the absence of any prior interaction experiences. Following the trust computation, verify the trust score. If it is above the threshold, the node may be trusted to offload data to them. If the trust score is below the threshold, ask the user to input before offloading any data; if not, place the fog node in the monitor mode.

Fig.3.2 Flow diagram of proposed model



4. Results

4.1 Overview of the Simulator

Cloud data centers, IoT devices, and fog nodes comprise of IoT-based fog infrastructures. Creating a real-world fog scenario for research purposes is highly expensive. As a result, employing simulators to recreate fog conditions is both practicable and affordable. However, just a few simulators, such as FogNetSim++, Edgecloudsim, and iFogSim, are accessible in this regard. All of these simulators allow you to model and simulate the conditions of fog nodes. iFogSim is the best choice among these toolkits because it supports both entities and services. Most researchers use iFogSim to simulate a variety of fog situations [10] since the scholarly community has given it a lot of support [11]. It is a framework that makes it possible to model and simulate fog and Internet of Things scenarios. It can keep an eye on a number of performance indicators [12]. In order to mimic any application situation in a fog environment, distributed dataflow and Sense-Process-Actuate application models are utilized. Evaluation of power consumption, end-to-end latency, QoS satisfaction and network congestion is made easier by it. For application environments that include IoT, fog, and cloud, this simulator is ideal. This simulator connects to the fog nodes for the first time.

The cloud in our scenario does not allow data offload on fog nodes and has a hierarchical design [13]. Thus, the iFogSim simulator has also been used to mimic the suggested method. HP laptop with an Intel Core i5 CPU and 8GB of RAM is used for the simulation. After the process of offloading, we compute trust. Consequently, iFogSim is a useful simulator for proposing resource management strategies in a fog computing context [14].

iFogSim's design is built on a layer that handles a certain function. Physical, logical, and managerial components make up its three basic parts. All of these elements work together to support resource and quality of service monitoring in fog and IoT-based fog environments [15]. We add more classes to implement trust and change a few iFogSim classes to apply our method. Below are the specifics of the suggested classes:

- Sensors
- Fog Devices
- Tuples
- Offloads and trust

4.1.2 Simulation Setup

Because of its advanced research functions, the simulated result evaluation for fog networks has gained global acceptance. It was also our preference to assess the suggested algorithm in a simulated environment using iFogSim, an addition to the iCloudSim toolbox [10].

Table 4.1: *Simulation Settings*

Simulation parameters	Value
Simulator	iFogSim
Operating system	Windows 10
Number of fog nodes	15
Network topology	Mesh

Table 4.2 displays the features of the simulated system. To apply our iFogSim simulator technique, we make changes to the pre-built classes, including Module Edge ward policy and Module Mapping.

Table 4.2: Configuration of Nodes

NAME	RAM	MIPS	LEVEL	UPLINK BW	DOWN BW
Fog Nodes	4000	2800	1	1000	1000
Sensor	1000	2500	2	1000	1000

Fig. 4.2 depicts the physical simulation network structure of our methodology.

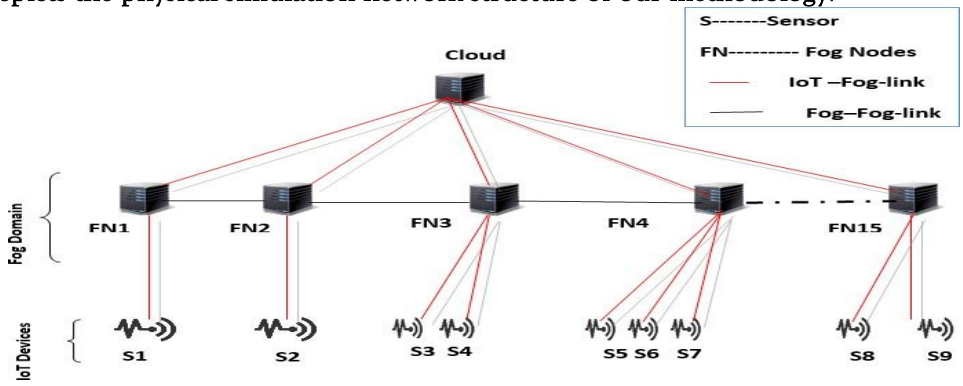


Figure 4.2: Physical Topology of Simulation Table

Discussion

This section discusses the particulars of implementation with different setups, such as adjusting the load on fog nodes and the sensing interval. The evaluation of node recommendation accuracy for offloading, the detection of illicit cooperation requests, and the impact of offloading with and without trust were the first topics covered. Furthermore, we compare our technique to that of "COMITMENT: A Fog Computing Trust Management Approach." [1].

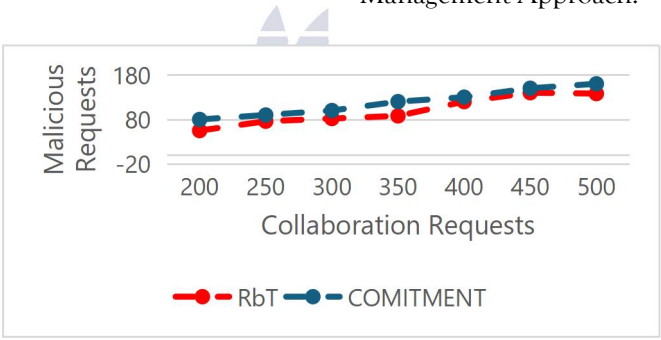


Figure 5.1: Malicious collaboration requests based on the LoT score

5.1 Malicious collaboration requests detection

We utilized the Level of Trust (LoT) score to identify requests for secure cooperation that are malicious. Higher LoT values indicate greater reliability, whereas lower LoT scores indicate less reliability. The trust score of every node is initially set to 0.5, or a neutral score. Following that, the trust level based on secure or malicious requests is reviewed by the LoT score generated by LoT function. Finding more trustworthy nodes and choosing the best offloading option are aided by it. For the two methodologies studied, the outcomes of malicious node cooperation requests are displayed in Figs5.1 and 5.2, respectively, for malicious and secure collaboration requests. In our example, Figure 5.1 shows that the number of malicious cooperation requests is lower. In our case, there are about 80 bogus collaboration requests. On average, [1] includes 120 harmful collaboration requests. It shows that the recommended strategy generates 33% fewer detrimental collaboration requests. This is because the trust in our proposed solution is based on the current node trust score. The Nodes are kept in monitor mode, resulting in much fewer hazardous cooperation requests compared to [1]. The proposed model has an average of 142 secure collaboration requests, whereas [1] only has 106, resulting in a 25% difference. This is due to the fact that, under the suggested architecture, only the reliable provider node—and not the malicious node with a lower trust score—may participate.

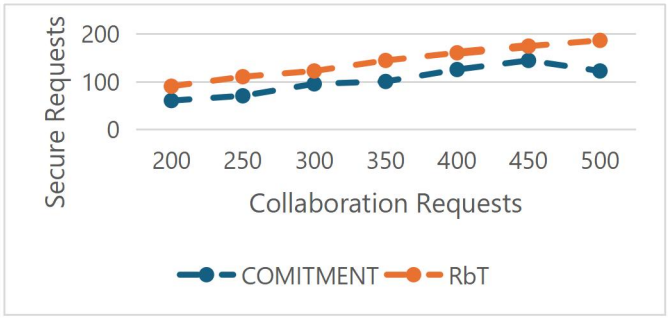


Figure 5.2: Secure collaboration requests based on the LoT score

5.2 Node recommendation for offloading

We discovered that less trusted nodes were only selected for offloading in this experiment if there was no trust check after running the simulation multiple times. In the proposed model, there is no option of selecting less trustworthy nodes since nodes are picked only on the basis of their confidence. It is prohibited to choose a node for offloading with a lower trust value (0.5 in our case). The goal of this experiment is to determine how successfully the proposed strategy selects the most trustworthy node for the offloading. We have assumed that the node's level of trust influences whether the QoS is satisfied. Figure 5.3 shows the study's suggested nodes. Figure 5.3 shows that the proposed model only selected reliable nodes, which explains the lack of any discernible pattern in the node selection process. The comparison approach simultaneously selected nodes at random for offloading. It shows how, without considering their reliability, it selected the first node in the first iteration, the first node in the second iteration, and so on.

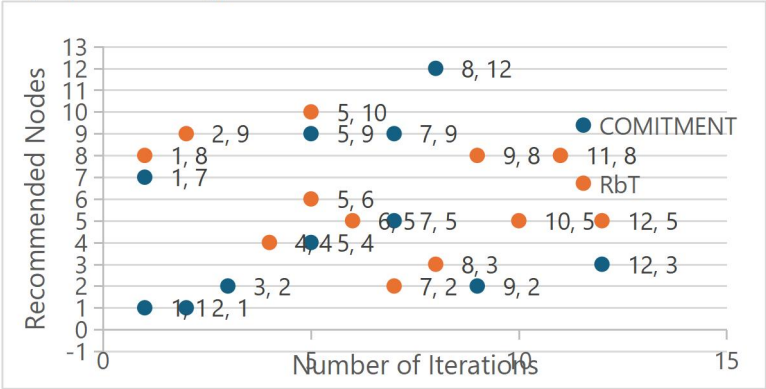


Figure5.3: Recommended nodes for offloading

5.3 Latency

In addition, the latency factors of the two previously mentioned models are compared. We took different tuple sizes into consideration. We added more characteristics to heterogeneous fog nodes during the simulation. The fog device's maximum MIPS, the tuple MIPS, and the service arrival rate per second are all fixed. Figure 5.4 shows the average reaction time to all received requests, with the y-axis representing the latency of both models in processing the requests and the number of interactions represented by x-axis. The proposed model handled the same number of requests in 0.042 milliseconds on average, whereas [1] took 0.055 milliseconds. Because the queue size or busyness of a fog node is taken into account when calculating trust, the suggested trusted node will process the request more quickly than other nodes while offering a level of trust and more secure offloading to the trusted nodes than when no such mechanism is used. This results in 23.6% less latency in the proposed model. The available resources at the node where the data will be offloaded play a significant role in the trust calculation of the recommended trust mechanism, which is especially tailored for time-sensitive applications.

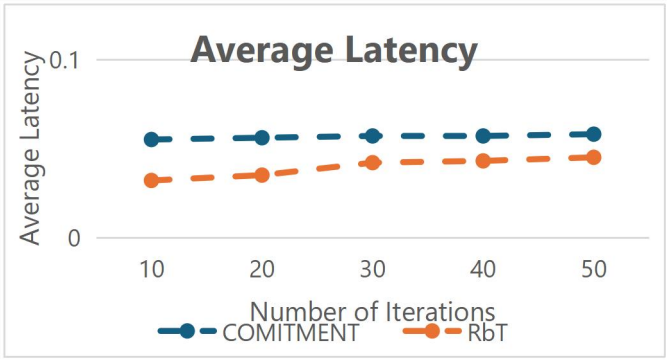


Figure 5.4: Average latency against two algorithms (COMITMENT and RbT)

Conclusion

Fog computing stands as a vibrant field of research, significantly contributing to addressing quality of service (QoS) and latency challenges. In the contemporary landscape, where applications depend on both time and location so achieving minimal response time is paramount. This is especially crucial for IoT services, which demand minimal service delays. Given the resource and computational limitations of fog nodes compared to the cloud [5], this study introduces an efficient trust model for offloading among fog nodes. The proposed reputation-based trust management solution is based on extensive experiment and simulations of situations and hence our caliber stands higher than the current approaches with regards to detecting malicious nodes and response time. The data provides evidence in favor of the effectiveness of trust via reputation-based techniques which performs better than COMITMENT model.

REFERENCES

[1] M. Al-khafajiy, T. Baker, M. Asim, Z. Guo, R. Ranjan, A. Longo, D. Puthal, and M. Taylor, "Comitment: A fog computing trust management approach," Journal of Parallel and Distributed Computing, vol. 137, pp. 1–16, 2020.

[2] E. Alemneh, S.-M. Senouci, P. Brunet, and T. Tegegne, "A two-way trust management system for fog computing," Future Generation Computer Systems, vol. 106, pp. 206–220, 2020.

[3] I. Stojmenovic and S. Wen, "The fog computing paradigm: Scenarios and security issues," in 2014 federated conference on computer science and information systems, pp. 1–8, IEEE, 2014.

[4] A. Bozorgchenani, D. Tarchi, and G. E. Corazza, "An energy and delay -efficient partial offloading technique for fog computing architectures," in GLOBECOM 2017 - 2017 IEEE Global Communications Conference, pp. 1–6, IEEE, 2017.

[5] H. A. Khattak, M. Imran, A. Abbas, and S. U. Khan, "Maintaining fog trust through continuous assessment," in World Congress on Services, pp. 129–137, Springer, 2019. [6] Y. Hussain, H. Zhiqiu, M. A. Akbar, A. Alsanad, A. A.-A. Alsanad, A. Nawaz, I. A.

[6] Khan, and Z. U. Khan, "Context-aware trust and reputation model for fog-based IoT," IEEE Access, vol. 8, pp. 31622–31632, 2020.

[7] R. Latif, "ConTrust: A Novel Context-Dependent Trust Management Model in Social Internet of Things," in IEEE Access, vol. 10, pp. 46526-46537, 2022.

[8] M. Kaur and R. Aron, "FOCALB Fog Computing Architecture of Load Balancing for Scientific Workflow Applications", Springer, 2021.

[9] D. Shehada, A. Gawanmeh, Chan Yeob Yeun, M. Jamal Zemerly, "Fog-based distributed trust and reputation management system for internet of things", Journal of King Saud University – Computer and Information Sciences,2021. 4950

[10] K. S. Awaisi, A. Abbas, S. U. Khan, R. Mahmud, and R. Buyya, "Simulating fog computing applications using ifogsim toolkit," .

[11] M. I. Bala and M. A. Chishti, "Offloading in cloud and fog hybrid infrastructure using ifogsim," in 2020 10th International Conference on Cloud Computing, Data

Science & Engineering (Confluence), pp. 421–426, IEEE, 2020.

[12] R. Mahmud and R. Buyya, "Modelling and simulation of fog and edge computing environments using ifogsim toolkit," Fog and edge computing: Principles and paradigms, pp. 1–35, 2019.

[13] M. M. E. Mahmoud et al., An Energy Efficient-aware Fog-enabled Cloud of Things Model for Healthcare. PhD thesis, Sudan University of Science & Technology, 2018.

[14] B. Jamil, M. Shojafar, I. Ahmed, A. Ullah, K. Munir, and H. Ijaz, "A job scheduling algorithm for delay and performance optimization in fog computing," Concurrency and Computation: Practice and Experience, vol. 32, no. 7, p. e5581, 2020.

[15] D. P. Abreu, K. Velasquez, M. Curado, and E. Monteiro, "A comparative analysis of simulators for the cloud to fog continuum," Simulation Modelling Practice and Theory, vol. 101, p. 102029, 2020.

[16] H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, and R. Buyya, "ifogsim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments," Software: Practice and Experience, vol. 47, no. 9, pp. 1275–1296, 2017.