Article

TETES: Trust Based Efficient Task Execution Scheme for Fog Enabled Smart Cities

Ahmad Naseem Alvi [1], Bakhtiar Ali [1], Mohamed Saad Saleh [2,*], Mohammed Alkhathami [2], Deafallah Alsadie [3] and Bushra Alghamdi [2]

[1] Department of Electrical and Computer Engineering, COMSATS University Islamabad, Islamabad 45550, Pakistan; naseem_alvi@comsats.edu.pk (A.N.A.); bakhtiar_ali@comsats.edu.pk (B.A.)

[2] Information Systems Department, College of Computer and Information Sciences, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh 11432, Saudi Arabia; maalkhathami@imamu.edu.sa (M.A.); 444008873@sm.imamu.edu.sa (B.A.)

[3] Information Systems Department, Umm Al-Qura University, Makkah 21955, Saudi Arabia; dbsadie@uqu.edu.sa

* Correspondence: msmsaleh@imamu.edu.sa

Abstract: Quality lifestyle leads to increasing trends in smart cities by offering modern communication and information technologies. Smart cities offer multiple applications with smart management of resources such as smart agriculture, Intelligent transportation systems, waste management and energy management. These applications are based on IoTs that are composed of sensor networks with limited processing and computing capabilities and are connected with different types of networks. Due to limited computational capability, IoT sensor nodes require more time to compute different tasks and are required to offload some tasks to remotely placed cloud servers for task execution. Fog nodes are preferred over the cloud as they are placed in close access to IoT nodes distributed in different networks. Different types of networks make it more vulnerable to malicious attacks. Malicious nodes offload complex and high computing tasks to fog nodes to compromise their performance and create delays in the computing tasks of legitimate nodes. In addition, fog nodes even after removing the malicious nodes are unable to process all the legitimate tasks within a specific time frame. In this work, a Trust-based Efficient Task Execution Scheme (TETES) is proposed for fog node that scrutinizes the offloaded tasks sent by the malicious nodes and efficiently execute most of the trusted tasks within a stipulated time cycle. The simulated results show that TETES execute more offloaded tasks as compared to well-known First Come First Serve (FCFS), Longest Task First (LTF), and Shortest Task First (STF) algorithms.

Keywords: trust management; fog computing; task offloading; smart cities

check for updates

Citation: Alvi, A.N.; Ali, B.; Saleh, M.S.; Alkhathami, M.; Alsadie, D.; Alghamdi, B. TETES: Trust Based Efficient Task Execution Scheme for Fog Enabled Smart Cities. *Appl. Sci.* **2023**, *13*, 12799. https://doi.org/10.3390/app132312799

Academic Editors: Siti Hafizah Ab Hamid and Raja Wasim Ahmad

Received: 14 October 2023
Revised: 22 November 2023
Accepted: 24 November 2023
Published: 29 November 2023

1. Introduction

The concept of smart cities has been emerging rapidly in the last few years as it provides a quality lifestyle by offering modern communication and information technologies. Smart cities provide home surveillance, intelligent transportation systems, smart industrial infrastructure, health care systems, agricultural farming, energy management, waste management and resource management [1,2]. These applications demand secure data communication and reliable delivery of data to provide secure and successful communication.

Internet of Things (IoT) applications in smart cities comprise of number of sensors that are required to forward their data to other nodes. With the evolution of 5G and 6G technologies, these nodes are required to adapt themselves to communicate with such base stations and also with their neighbouring nodes in transferring the data by using heterogeneous communication [3–5]. This heterogeneous communication makes it vulnerable and malicious nodes may attack it and disturb the communication channel. This becomes more serious when data comes from a health care system where unreliable data might be life-threatening because treatment and suggestions from the specialist physician are based on this patient's data [6].

Smart cities comprise various applications that possess multiple networks with different service providers. In this scenario, trust with different service providers is a fundamental challenge and is required to be implemented by slicing a network over different infrastructures by considering the different service providers and different types of users [7–9]. Furthermore, virtual machines are susceptible to different types of security attacks so should be un-trusted during the communication operation, otherwise, it may disturb the network performance resulting in the QoS of the network being seriously compromised [10,11]. There is also a possibility that the virtual networks are deployed by a third party and there are chances that it may become infected during network operation resulting in communication disturbances.

The dramatic increase in the IoT heterogeneous network requires trust management in communicating their data to other devices is such a challenging job. IoT devices mostly comprise of large number of tiny sensors with limited computational and resource capabilities [12,13]. Though limited resources reduce the cost of these sensors, at the same time they are more vulnerable to other networks and are under attack. This makes trust management an even bigger challenging job.

Multiple smart city application such as healthcare and industrial applications are based on IoT networks, that are based on large number of wireless sensor nodes. These wireless sensor nodes generate vast amounts of raw data. Transmitting this huge amount of raw data to the central location makes it inefficient and it is required to be filtered and analyzed locally and only the relevant information needs to be forwarded to the concerned node. This may require sensor nodes to run complex analytic algorithms. Wireless sensors have limited processing capacity and are unable to execute such complex algorithms and need to offload their tasks to cloud servers. However, approaching cloud servers with multi-hop distances causes communication delays and the quality of services is compromised. To overcome these delays fog computing nodes are deployed in the near vicinity of these IoT nodes. Fog nodes consist of single or multiple data centres located at the edge of user networks without routing over the Internet backbone [14–16]. A fog node-based smart city IoT network is shown in Figure 1. Fog nodes with higher processing capacity than the IoT sensor nodes, execute these offloaded tasks in relatively much less time.

The presence of malicious nodes compromises the performance of fog nodes by intentionally offloading the unwanted complex tasks to fog nodes. Fog nodes utilize their resources in executing these tasks by considering them as legitimate tasks, resulting in increased task execution time for legitimate nodes. Differentiating between the legitimate and malicious offloaded tasks before their execution improves the performance of the fog nodes in executing legitimate tasks. Collecting trust values of all the nodes present in a network requires more time and overheads are also increased. To overcome this delay and control overheads, the network is divided into a number of clusters.

Assessing legitimate node before the start of communication makes it more secure and increases the data reliability with improved quality of work [17,18]. In order to assess data legitimacy, centralized authentication scheme is used to identify the malicious nodes data. However, it is not preferred in IoT based smart city environment due to following limitations.

- Centralized scheme faces challenges in dynamic and evolving systems because it struggles to adapt quickly.
- Decision making at the central level creates bottle neck resulting in a compromised communication environment.
- Centralized authentication does not encourage the secure and cooperative environment.
- There are increased number of inside threats in centralized systems where authentication decisions are primarily based on credentials.

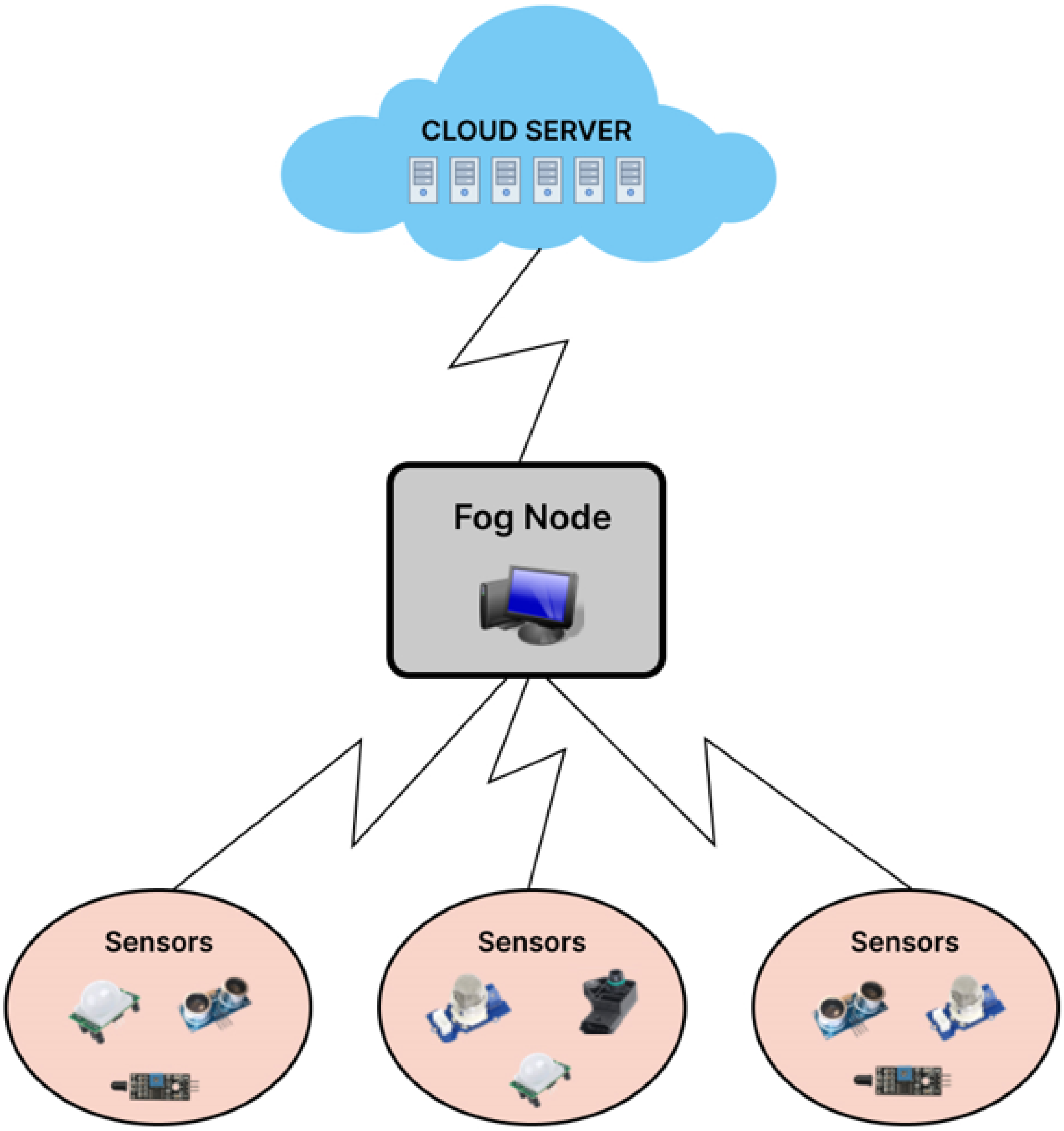

Figure 1. Fog enabled IoT network in smart cities.

Classic centralized authentication scheme only identifies the malicious nodes that are not part of the member. However, there are many attacks from the malicious nodes which are also members of the network. IoT based smart cities applications demand adaptive and dynamic environment [19]. In such scenario, a trust based scheme is preferred over classic centralized authentication scheme. Malicious nodes being a member of the wireless network, offload complex tasks to compromise the execution efficiency of the fog node, because it consumes its processing resources in executing these malicious tasks. To improve the efficiency of fog node in task execution, it needs to find the tasks offloaded by a trusted node or not. A trust management system is designed to differentiate between legitimate and malicious tasks and improve the quality of the network by identifying malicious nodes.

In this work, a Trust-based Efficient Task Execution Scheme (TETES) is proposed for fog nodes to effectively discard the tasks offloaded by fog malicious nodes and proposes an algorithm to efficiently execute the maximum number of legitimate tasks. The salient features of TETES are mentioned below.

1. Distribution of IoT networks attached with a fog node in small clustering units in order to minimize the delay and control overheads.
2. Trust management scheme to help fog node in evaluating the trust value of each IoT node in a cluster to differentiate between legitimate nodes and malicious nodes.
3. An efficient 0/1 knapsack-based task offloading scheme for fog nodes to execute a maximum number of offloaded tasks within a specified time.

The rest of the paper is organized as: The previous research work relating to trust management in different prospects is discussed in Section 2. Sections 3 and 4 describe the system model and our proposed scheme respectively. Comparative analysis with extensive simulations and results are discussed in Sections 5 and 6 concludes our manuscript.

2. Related Work

Smart cities require different varieties of networks to support multidimensional applications widely supported by IoT setups. Trustful communication between different varieties of applications is a dire need to avoid malicious nodes. That's why, it is in highly researched area these days and there is a lot of research on trustful management in different areas of the communication field.

In [20], authors emphasized the tradeoff between benefits and risks in trustful communication and developed a multi-aspect and adaptive trust-based situation-aware access control framework and named it "MATS". The framework is based on semantic web technologies and game theory by considering dynamic network situations by carefully balancing efficiency and simplicity. The authors validate the performance of their framework with experimental results.

In [21], authors proposed a trust mechanism for Unmanned Aerial Vehicles (UAVs) and IoTs in vehicular network scenarios. The proposed framework emphasizes two ways: Firstly a deadline deadline-aware data is collected in collaboration with mobile vehicles and UAVs, and secondly an active and verifiable trust approach is used by considering the security and privacy of the system. Finally, a trajectory optimization algorithm is proposed to collect maximum baseline data to decide on trustful agreements. The authors claimed that their proposed framework improved the data accuracy by 33.34% on average with 35% reduced cost and 58.32% reduced delay.

In [22], authors addressed the trustful data needs in online dating networks and emphasized harmful attacks such as creating fake accounts. The authors proposed a trust-aware framework to detect malicious users by developing a user trust model and applied a data balancing technique to find out the malicious nodes continuously send data and disturb the system. The authors evaluated the performance of their proposed scheme through extensive experiments and claimed that their proposed model improves the data precision up to 59.16% more form the other baseline algorithms.

In [23], authors focused on big data problems and proposed a MapReduce-based framework in big data processing tasks by initially quantifying and sensitising the data and trust values. Depending upon this received data, the authors formulated the MapReduce scheduling problem with the bipartite matching problem by considering the maximum weights. The problem became the NP-hard problem and is tackled by sharing the slot nodes within a computing node by allocating the same level of trust share to each slicing network user. After tackling this NP-hard problem, an efficient heuristic-based algorithm is proposed. Authors claimed that their proposed framework achieved 94.7% of the optimal solution obtained through exhaustive search in a big data environment.

In [24], V. Varadharajan et al. addressed the fundamental issues faced in online property-based trust attestation due to virtual network involvement. The authors proposed a model to eliminate the virtual network problems faced during boot and run time. The authors defined simple rules in order to identify legitimate property data users. The authors claimed that their proposed model helps in developing trustworthy applications. The authors implemented a prototype of a trust management platform by using an opensource MANO platform and claimed that their proposed platform slightly reduced the performance of the network over a virtual network function in a dynamic environment.

In [25], M. Ebrahimi et al. focused on an IoT-based healthcare system and exploited the evidence theory to a decentralized service-oriented trust management model in it. Malicious attacks on healthcare systems are avoided by measuring the evidence distance by rewarding the healthcare service providers and punishing the malicious users. The authors proposed a two-fold system model by providing trust in healthcare services and trust in recommendations. The authors claimed that their proposed system is robust and efficient due to dynamic parameters and provides security against bad mouthing, good mouthing and on-off attacks.

Besides, the above work, profit and cost analysis of fog computing networks is critical [26,27]. Multiple factors related to cost can be considered while offloading the tasks and managing the resources of fog nodes.

3. System Model

Diverse applications of a smart city are based on IoT nodes. These nodes are designed to meet the application requirements for necessary automation. Due to the limited processing and computing capacity of these nodes, they are unable to perform most of their tasks and have to offload these tasks to other nodes. The fog nodes are placed in different areas and the IoT nodes are supposed to offload their tasks to the fog node placed in their close vicinity. All fog nodes are backwardly connected with the cloud server. A complete system model is shown in Figure 2.

IoT nodes of different applications are subdivided into small clusters and multiple clusters are connected with one fog node. Nodes in all these clusters are supposed to offload their tasks to their attached fog node directly or through intermediate nodes. There are N number of legitimate nodes and M number of malicious nodes in each of the C clusters. The maximum time interval required in transferring the offloaded task data of any of the nodes placed in the connected cluster to the fog node is T_{max}.

After each T_{max}, the fog node computes the total number of offloaded tasks received within this time duration. If a node b offloads T_b tasks from C_a clusters to fog node and there are J number of nodes in each of the K numbers of clusters, the total number of offloaded tasks (σ_{OL}) that are required to be executed by fog node are calculated as:

$$\sigma_{OL} = \sum_{a=1}^{K} \sum_{b=1}^{J} C_a \times T_b \tag{1}$$

There are m malicious nodes present in each of the K number of clusters and m_a malicious node m_a offload m_t tasks then a total number of tasks offloaded by legitimate nodes (σ_{LN}) is calculated as:

$$\sigma_{LN} = \sum_{a=1}^{K} \sum_{b=1}^{J} (C_a \times T_b) - m_a \tag{2}$$

All the offloaded tasks are different in nature and require different execution times. If the average task execution time of a task is T_d, then the total execution time ζ_{OL} for all offloaded tasks with a uniform processing capacity of fog node is calculated as:

$$\zeta_{OL} = \sum_{a=1}^{K} \sum_{b=1}^{J} (C_a \times T_b) \times T_d \tag{3}$$

The task execution time of offloaded tasks by all legitimate nodes ζ_{LN} is calculated as:

$$\zeta_{LN} = \sum_{a=1}^{K} \sum_{b=1}^{J} [(C_a \times T_b) - m_a] \times T_d \tag{4}$$

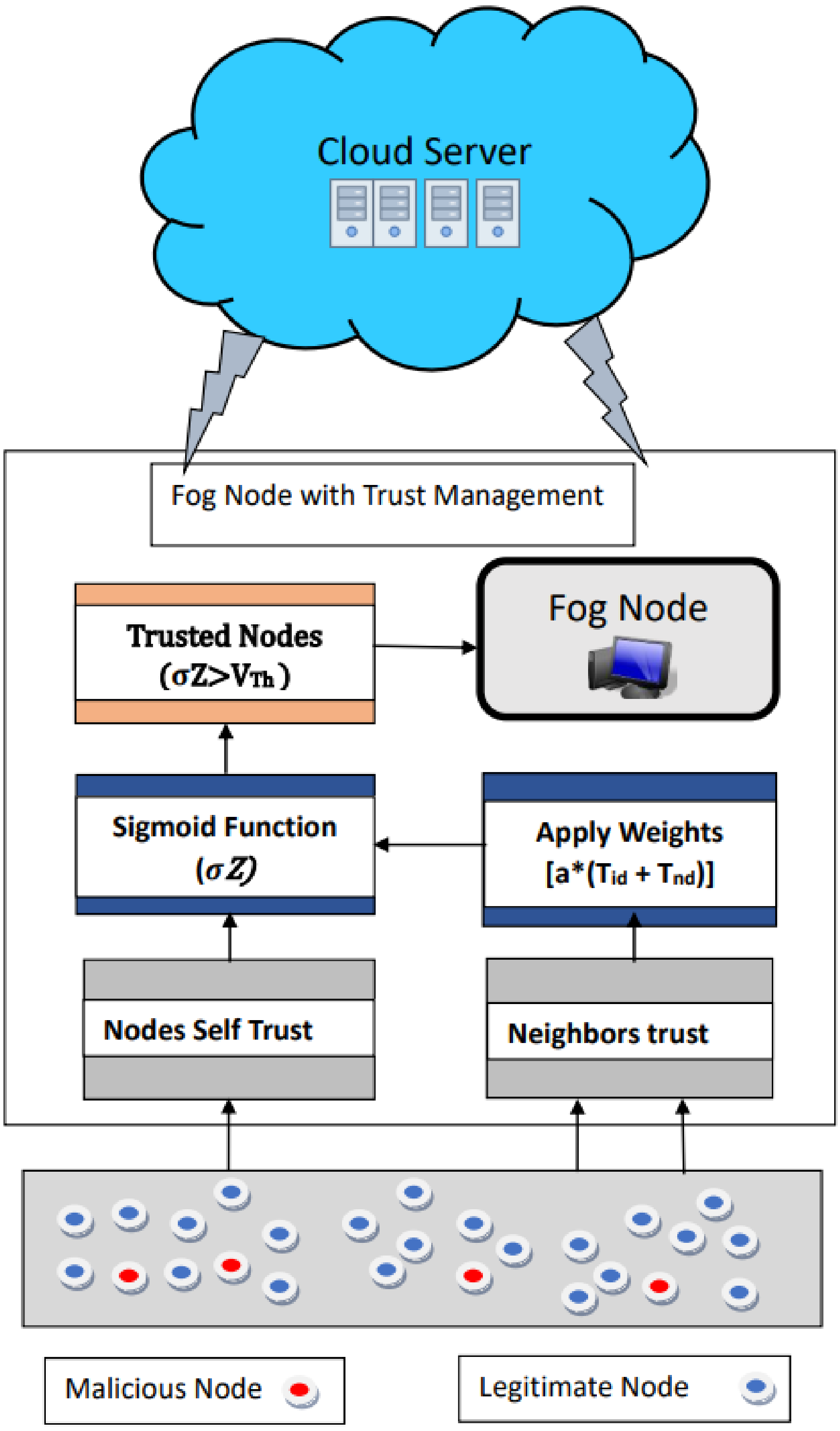
Cloud Server
Fog Node with Trust Management
Trusted Nodes ($\sigma Z > V_{Th}$)
Fog Node
Sigmoid Function (σZ)
Apply Weights [$a*(T_{id} + T_{nd})$]
Nodes Self Trust
Neighbors trust
Malicious Node
Legitimate Node

Figure 2. System Model.

4. Proposed Scheme

In this work, a trust management scheme for fog computing nodes is proposed to efficiently execute the offloaded tasks received from different smart city applications. Malicious nodes are supposed to upload their tasks to compromise the performance of the fog node by utilizing its resources to execute the tasks offloaded by a malicious node. This may cause a delay in the execution of offloaded tasks of legitimate nodes. In this work, a Trust-based Efficient Task Execution Scheme (TETES) for fog-enabled smart cities is proposed. The salient features of TETES are:

- IoT nodes in diverse areas are divided into multiple small clusters.
- trust level calculation of all nodes in a cluster based on different parameters through direct and indirect trust calculations.
- an efficient task execution algorithm for fog computing nodes.

4.1. Clustering Setup

IoT-based sensor nodes are widely placed in a diverse range of smart city applications. Fog nodes are placed in different locations of smart cities for fast processing. These diversely placed wireless networks are divided into n number of small clusters and a fog node must be a part of one of these clustering network. Each of the sensor node must join one of the clustering networks.

A node N_i joins a cluster in such a way that the nodes which are in close vicinity of fog nodes are placed in one cluster. If N_i finds fog nodes F_a and F_b in its close vicinity, then it will prefer to become a member of fog node F_a, provided its hop count is less than the hop count of F_b node. In case, both the fog nodes are accessible with the same hop count, then priority will be given to the one that has less number of member nodes.

4.2. Trust Management

The presence of malicious nodes in a system degrades the QoS of the network by offloading the fictitious tasks to the fog node to reserve its computing resources. In this section, a mechanism to find out the trusted offloaded tasks is discussed. TETES proposed an algorithm to evaluate the probability of trusted nodes through the information collected by all nodes in a clustering network. TETES helps fog nodes to determine the trust value of fog nodes after regular time intervals and identifies the trust value of all the offloaded tasks nodes.

The trust level of a node is evaluated by its neighbours when they exchange their control and data packets with their neighbouring nodes. If requested data packets are received correctly against the number of requests generated. If a node N_a send K requests to its neighbouring node N_b and it receives J response files from N_b node. If J_{Req} are correctly received requested files and J_{corr} are corrupted files received, then trust level of node N_a for node N_b (T_b^a) is calculated as:

$$T_b^a = \frac{J_{Req} - J_{corr}}{K} \tag{5}$$

The probability of finding a legitimate node is evaluated on the following three parameter values.

1. Trust status of the trust finding node about its neighbouring nodes.
2. Trust value computed by its neighbours who are in direct access to the trust finding node.
3. Trust information is collected by all other nodes in a cluster except its neighbouring nodes.

All these trust finding parameters are described below.

4.2.1. Trust Status of Trust Finding Node

Each node is required to forward its trust-finding report of all its directly connected neighbours to the fog node of its cluster. The trusted value of trust finding node for all its directly connected neighbors is based on the total number of neighbors and the number of trusted neighbors around it.

If trust finding nodes T^{tf} has m number of directly connected neighbours and the trust value of each of its neighbouring nodes, that is calculated through Equation (5) is mentioned with VT_i^{tf} then trust value of VT^{tf} is calculated as:

$$T^{tf} = \frac{\sum_{i=1}^{m} T_i^{tf}}{m} \tag{6}$$

A legitimate node will send the true trust value of its neighbouring nodes, however, a malicious node may send a factitious trust value. So it can not be calculated from only this parameter value that it is a trusted node. To validate the trust value generated by this node, we need to get information from other nodes of the clustering network.

4.2.2. Neighboring Nodes Trust Value

All nodes who are in direct connection to the trust-finding nodes are required to send their trust values about the trust-finding node to the fog node. This helps in evaluating the trust value of each of its neighbouring nodes.

If there are k number of nodes that are placed in direct connectivity of the trust finding node, then the trust value of all k neighbours for trust finding node T_{nd}^{tf} is calculated as:

$$T_{nd}^{tf} = \frac{\sum_{j=1}^{k} T_k^{tf}}{k} \tag{7}$$

4.2.3. Indirect Trust Value

All those nodes in a cluster, that are not in the direct connection with the T^{tf} nodes are required to send their trust value for T^{tf}. These nodes exchange their information with T^{tf} through the intermediate node/s and calculate their trust value individually as mentioned in (5). Indirect trust value of all non-neighboring nodes for T^{tf} is calculated as:

$$T_{id}^{tf} = \frac{\sum_{j=1}^{k} T_k^{tf} \times H}{k} \tag{8}$$

here, H is the trust value of intermediate nodes that is between 0 and 1. This results in indirect trust values always less or equal to the trust values calculated by the directly connected nodes. The larger the number of intermediate nodes to the trust finding nodes, the smaller will be this trust value due to the conditional probability.

4.2.4. Trust Probability

Trust probability of all nodes present in a cluster are calculated by assigning weighted metric on trust values calculated for above mentioned three parameters. The highest weight is applied to the accumulated trust values collected from the directly and indirectly connected neighbouring nodes for the trust-finding node. Weights assigned to the trust values of the node itself T^{tf} is the lowest as it may have false information due to malicious node and it will be collective trust value of T_{nd}^{tf} and T_{id}^{tf}.

The trust probability is calculated through a sigmoid function (σZ) for each trust-finding node in a cluster as:

$$\sigma Z = \frac{1}{1 + e^{-[a(T_{nd}^{tf} + T_{id}^{tf}) + T^{tf}(T_{nd}^{tf} + T_{id}^{tf})]}} \tag{9}$$

The nodes with trust values less than a threshold limit are considered malicious nodes. A complete trust-finding mechanism is shown in Figure 3.

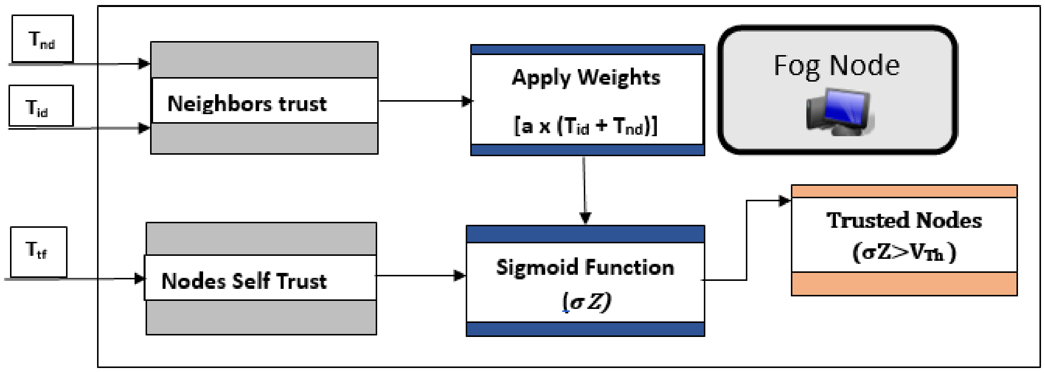

Figure 3. Trust Management Procedure for Legitimate Nodes.

Fog nodes after determining the trust value of each of its attached clustering nodes, compare it with a threshold value. In case, the trust value of a node is less than the predetermined threshold value then it is considered as a malicious node, otherwise, it will be considered as a trusted node. A complete procedure is mentioned in the Algorithm 1.

Algorithm 1: Trust Evaluating Algorithm for Fog Node

```
1  **Trust calculating policy**
2  **Input:**
3  Number of nodes $N$
4  Threshold Value $V_{th}$
5  Self-trust value of nodes $T_1^{tf}, T_2^{tf}, T_3^{tf}, \ldots, T_N^{tf}$
6  Direct trust of neighboring nodes for node $x = T_{ndx}^{tf}$
7  Indirect trust of nodes for node $x = T_{idx}^{tf}$
8  **for** $i = 1$ *to* $N$ **do**
9      Calculate $\sigma Z_i$ for all nodes
10     **if** $\sigma Z_i \leq V_{th}$ **then**
11         Node is Malicious
12     **end**
13     **else**
14         Node is legitimate
15     **end**
16     $i++$
17 **end**
```

4.3. Task Execution Procedure

Fog nodes after receiving all the offloaded tasks from its associated IoT nodes, need to execute these tasks. Fog nodes with the help of trust management system, has a knowledge about legitimate and malicious nodes. Malicious nodes intentionally offload the complex nature tasks to consume the processing capacity of fog nodes. This may cause delay in executing the tasks offloaded by the legitimate nodes and their tasks may not be executed within their defined timeline. TETES proposes algorithms for fog nodes to efficiently execute the offloaded tasks to minimize the delay in such a way that the majority of the trusted offloaded tasks will be executed within their deadline. The salient features of TETES in executing offloaded tasks are:

- excludes the offloaded tasks that are offloaded by malicious nodes
- an optimal task execution algorithm for the fog node to efficiently scrutinize the offloaded tasks, so that maximum tasks may be executed before their deadline.

The offloaded tasks of all those nodes that do not meet the required trust level will be discarded by the fog node by applying Algorithm 1. If a number of offloaded tasks can be executed within their defined deadlines after excluding malicious nodes' tasks, then all the tasks will be executed by the fog node by applying a first come first serve basis. However, if all the offloaded tasks can not be executed within their deadlines, then a maximum number of tasks are required to be executed within their defined deadline. To meet this requirement, an optimal execution of task offloading scheme is proposed that is based on the 0/1 knapsack algorithm.

The fog node has limited processing capacity and within a stipulated time cycle only a limited amount of offloaded tasks can be executed. At the same time, the offloaded tasks are different in size with varying deadlines as well and require different execution times. If the processing capacity of the fog node is mentioned by F_{proc}, and T_{et} is the execution time required by each offloaded task with priority T_p, then this task execution problem can be mapped with 0/1 knapsack problem as mentioned in the Table 1.

Table 1. Knapsack Mapping Parameters.

	Task Execution Problem	**Knapsack Problem**
F_{proc}	Processing capacity of fog node	Carrying capacity of the knapsack
T_{et}	Task execution time	Weight of item
T_p	Task priority	Value of item

Priority of each offloaded task of node i (P_i) is calculated through the maximum processing cycle of fog node (P_c), trust value of task offloading node (V_i^t), and the remaining time to reach its deadline V_i^t as:

$$P_i = V_i^t \times (P_c - V_i^t) \tag{10}$$

The higher the priority, the higher will be the value of the offloaded task.

Suppose there are T trusted nodes that have offloaded their tasks to fog node with F_{proc} processing capacity and T_{et}^i is the task execution time of node i with P_i calculated priority. The proposed algorithm scrutinizes S number of offloaded tasks by fulfilling the following two constraints:

- The accumulated task execution time of offloaded tasks should be within the task processing limit of the fog node and is expressed as:

$$\sum_{i=1}^{T} \leq F_{proc}$$

- The priority values of all the scrutinized tasks must be maximum in all respects.

$$Max \sum_{a=1}^{S} P_a$$

the knapsack problem is solved by implementing a knapsack table to scrutinize the offloaded tasks. A complete algorithm for implementing a knapsack table along with scrutiny of offloaded tasks for their execution are shown in Algorithm 2.

Algorithm 2: Optimal Task Scrutiny Criteria

```
1  $j \leftarrow$ Current task size
2  $P \leftarrow$ Fog node's processing capacity
3  $i \leftarrow$ Offloaded task ID
4  $T \leftarrow$ Max. no. of trusted tasks
5  $D_{max} \leftarrow$ Max. task execution time
   $X[i, j] \leftarrow$ Cell value of table withi$^{th}$task and j processing
6  $j_i \leftarrow$ task size requested by $i^{th}$ task
7  If $D_{max} \leq T$
8  Execute all tasks by applying STF
9  Else
10 Apply knapsack algorithm
11 filling of knapsack table:
12 for $j = 0$ to $P$ do
13     $X[0, j] = 0$
14     // Initialize 1st row to 0's
15 end
16 for $i = 1$ to $T$ do
17     $X[i, 0] = 0$
18     // Initialize 1st column to 0's
19 end
20 for $i = 1$ to $T$ do
21     for $j = 0$ to $P$ do
22         If $j_i \leq j$ If $j_i + X[i-1, j-j_i] > X[i-1, j]$
23         $X[i, j] = j_i + X[i-1, j-j_i]$
24         Else
25         $X[i, j] = X[i-1, j]$
26         EndIf
27         Else
28         $X[i, j] = X[i-1, j]$
29         EndIf
30     end
31     Initialize i and w:
32     $T \leftarrow i$
33     $P \leftarrow m$
34 end
35 optimal task selection:
36 while $i > 1$ and $j > 1$ do
37     If $B[i, j] > B[i-1, j]$
38     $i_{\text{th}}$ content is included in optimal solution
39     $i = i - 1$
40     $j = j - j_i$
41     Else
42     $i = i - 1$
43     EndIf
44 end
```

5. Results and Analysis

The performance of our proposed scheme TETES is evaluated by calculating the offloaded tasks executed by a fog node in different scenarios. To evaluate its performance, a simulation environment is created by deploying number of IoT based sensor nodes in diverse areas and they are supposed to communicate with each other. To determine the performance of the "TETES" in determining the trust value of each of the networking node, some malicious nodes are added in the simulation environment. These IoT nodes are subdivided into different groups to make a clustering network. Each cluster comprises both legitimate and malicious nodes. These nodes are supposed to offload their tasks to the fog node that is placed in close vicinity and any far-end node can access the fog node through a maximum of 2 intermediate nodes. The data size of each offloaded task ranges from 50 kB to 100 kB and the time required to execute these tasks ranges from 1 processing cycle to 4 processing cycles. A complete list of simulation parameters is shown in Table 2.

The results are compared with First Come First Serve (FCFS), Longest Task First (LTF), and Shortest Task First (STF) algorithms for varying numbers of offloaded tasks and varying fog node task execution capacity.

Table 2. Simulation Parameters.

Parameter	Value
Fog Computing Node coverage area	150 m
Distance between fog mining node and task requesting nodes	50 to 150
Number of intermediate nodes between fog and IoT nodes	0 to 2
Number of simultaneous offloaded tasks for each fog node	10 to 20
Data size of each offloaded task (kB)	50 to 100
Processing capacity of fog node (Processing Cycles)	10
Task deadline (Processing Cycles)	10
Task Processing time of offloaded tasks (Processing cycles)	1 to 2
Number of clusters connected with a fog node	2 to 3
Number of Malicious nodes	2 to 4
Number of legitimate nodes	10 to 18

5.1. Task Execution of Legitimate Nodes

In this section, several executed offloaded tasks of legitimate nodes along with their percentages are calculated for varying numbers of incoming offloaded task requests as well as for varying processing capacity of fog nodes.

Results in Figures 4 and 5 show the number of offloaded tasks that have been offloaded by legitimate nodes for varying processing capacity of fog nodes and varying numbers of offloaded tasks respectively. In Figure 4, results are obtained for a fixed number of offloaded tasks by all its associated nodes with varying processing capacity of fog node. The results show that the number of executed tasks increases with the increase in the processing capacity of fog nodes. It is evident from the results that the proposed TETES executes more offloaded tasks of legitimate nodes as compared to the other three algorithms for all different capacities of fog nodes. The tasks executed by STF are more than FCFS and LTF, because it executes smaller tasks first and more tasks will be executed. LTF executes the least number of tasks because it executes heavy tasks first and when processing capacity is low it executes less number of offloaded tasks. However, TETES by applying the knapsack algorithm optimally executes a maximum number of offloaded tasks within the processing capacity of the fog node.

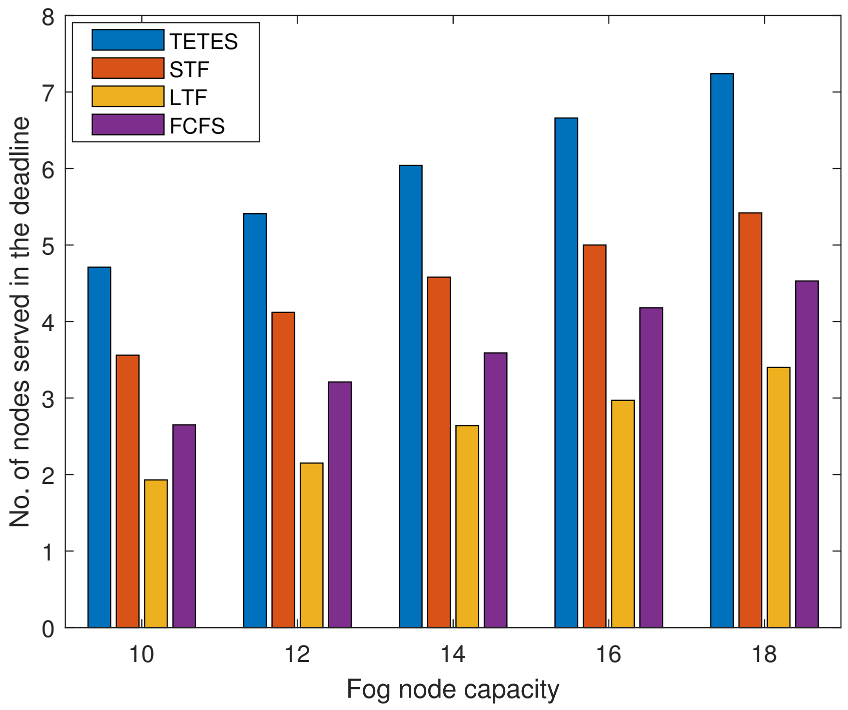

Figure 4. Number of Nodes Served for Varying Capacity of Fog Node.

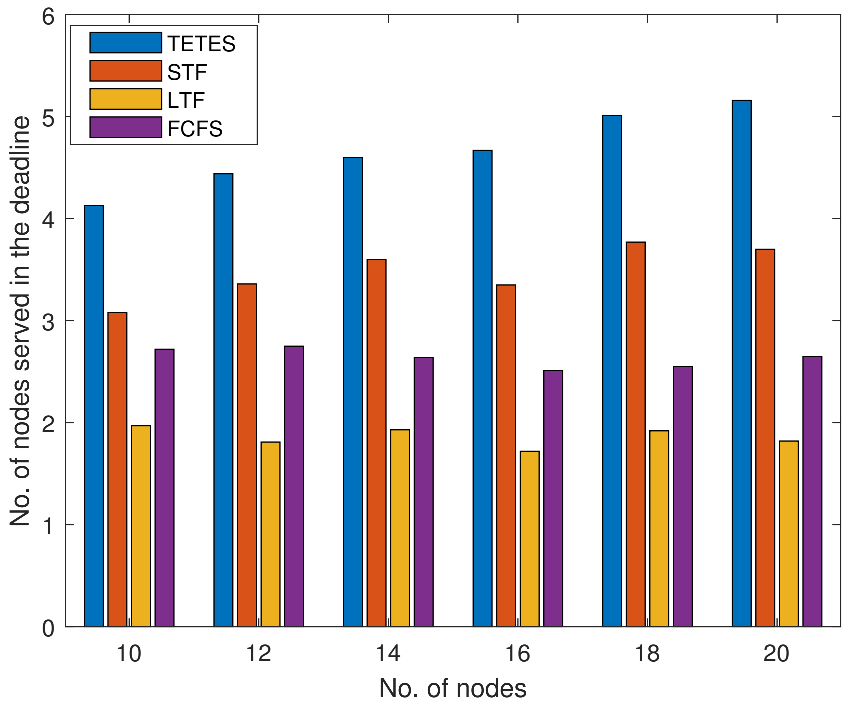

Figure 5. Number of Nodes Served for Varying Number of Requested Tasks.

Figure 5 shows the number of executed tasks of legitimate nodes for varying numbers of offloaded tasks with a fixed processing capacity of fog node. The results show that TETES executes more tasks with the increase in the number of offloaded tasks as compared to the other three schemes. Because TETES optimally scrutinizes the offloaded tasks that can be executed within the processing cycle of the fog node. The same increasing trend is observed in STF, as with an increased number of tasks, it executes smaller tasks within the processing capacity. However, the number of tasks executed in FCFS and LTF are almost the same for all increasing numbers of offloaded tasks.

Results in Figures 6 and 7 show the percentage of executed tasks offloaded by trusted nodes for varying processing capacity of fog node and varying number of offloaded tasks respectively. The results shown in Figure 6 verify that TETES executes more offloaded tasks as compared to the other three schemes for 15 offloaded tasks. TETES executes more than 32% of the total offloaded tasks when the processing capacity of the fog node is 10 and increases to 48% when the processing capacity of the fog node increases to 18. However, STF, LTF, and FCFS execute 24%, 12.6%, and 18% offloaded tasks, when the processing capacity is 10 and executes 35%, 22%, and 30% offloaded tasks respectively when the processing capacity of the fog node increases to 18 Processing cycles.

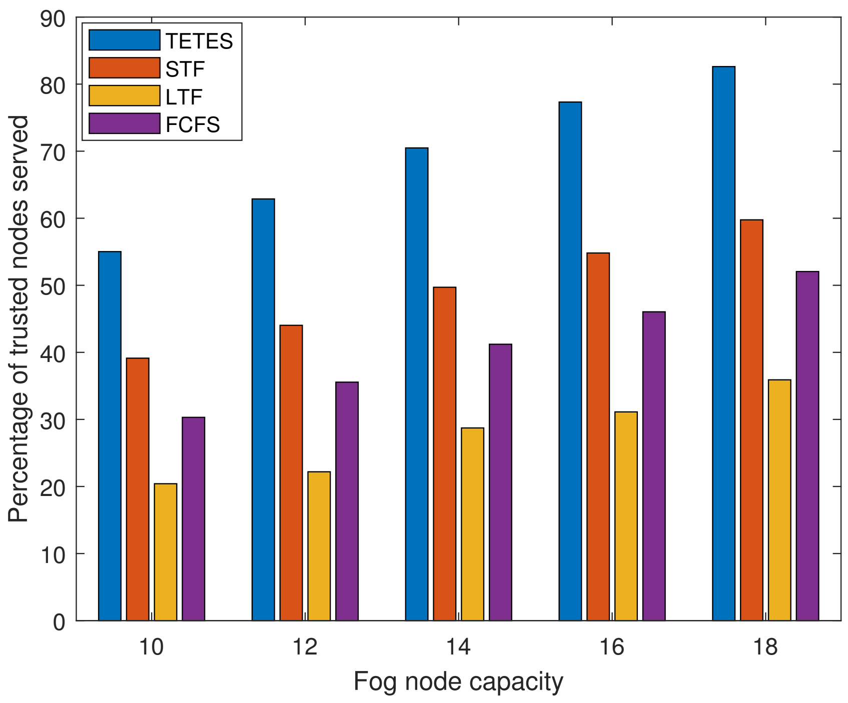

Figure 6. Percentage of Executed Trusted Tasks for Varying Processing Capacity of Fog Node.

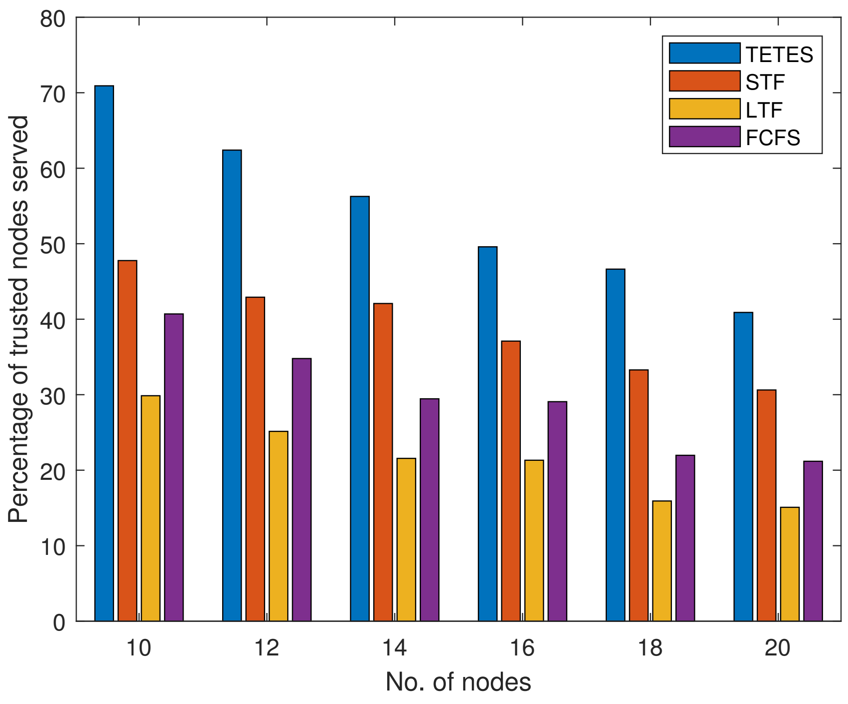

Figure 7. Percentage of Executed Trusted Tasks for Varying Number of Requested Tasks.

The results shown in Figure 7 verify that the tasks executed by TETES are more than the other three schemes and it executes 41% of tasks when a number of offloaded tasks is 10 and the fog node has fixed processing capacity. However, the percentage of executed tasks in TETES decreases to 26% with the increase in the number of offloaded tasks to 20, however, it is more than the other three schemes. This is due to the fixed processing capacity of the fog node as it can not execute more tasks. On the other hand, STF, LTF, and FCFS execute 30%, 19%, and 27% of offloaded tasks when the number of offloaded tasks is 10 and decreases to 19%, 9.5%, and 14% respectively when a number of offloaded tasks rise to 20.

5.2. *Executed Data of Trusted Nodes*

Performance of TETES is analyzed by computing the total amount of data for all those offloaded tasks that are executed successfully for varying capacity of fog node and for varying number of offloaded tasks as shown in Figures 8 and 9 respectively. The performance of TETES is analyzed in terms of executed data and is compared with STF, LTF, and FCFS.

Results shown in Figure 8 verify that the data executed by TETES is more than the other three schemes. The results further show that with the increase in processing capacity of the fog node, the amount of data executed by all algorithms increases. However, offloaded task data, that are executed by TETES is significantly large as compared to the other three schemes. This is due to the fact, that TETES not only eliminate the malicious nodes tasks but also efficiently executes the offloaded tasks by applying the 0/1 knapsack algorithm.

Figure 9 shows the executed amount of data against varying numbers of offloaded task requests received by the fog node. The results show that the amount of data executed by the proposed TETES is more than the other three schemes. The results further show that with the increase in the number of offloaded tasks, TETES executes more data. However, the amount of data executed by the other three schemes remains almost the same with the increased number of nodes and does not show an increasing trend as seen for TETES. This is due to the optimal selection of offloaded tasks from the offloaded tasks' range within the execution capacity of the fog node.

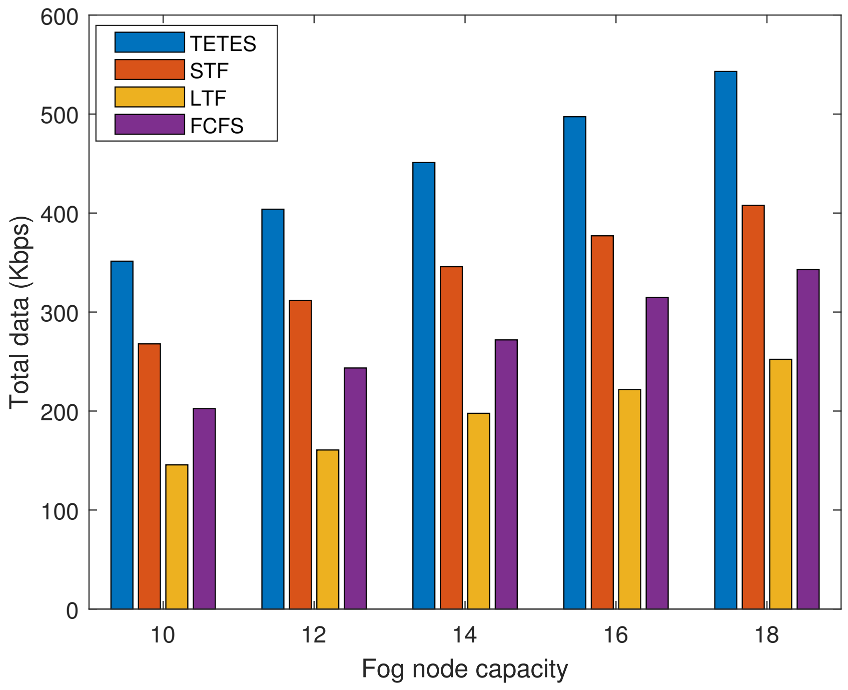

Figure 8. Total Amount of Processed Data for Varying Capacity of Fog Nodes.

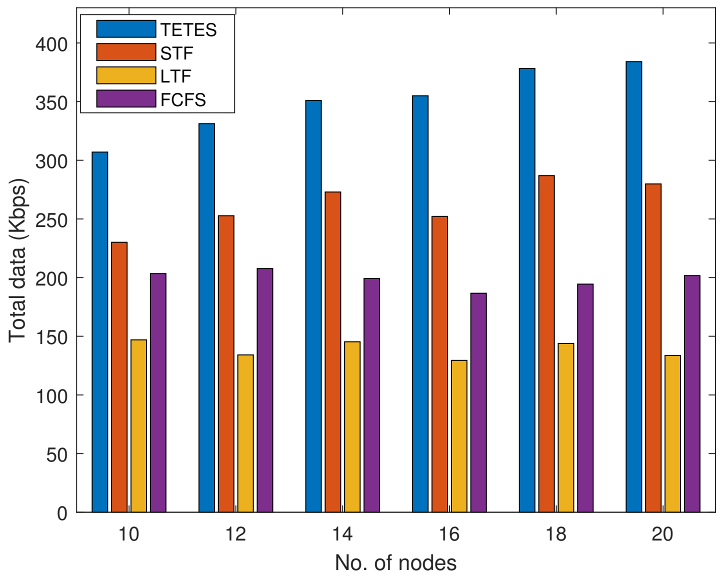

Figure 9. Total Amount of Processing Data for Varying Number of Requested Tasks.

5.3. Mean Trust Value

The mean trust value of offloaded tasks is calculated by taking an average of the trust value of all the executed offloaded tasks. If a number of executed tasks is m and the trust value calculated by the fog node for i_{th} offloaded tasks is T_i, then the mean trust value (MTV) for these executed tasks is calculated as:

$$MTV = \frac{\sum_{i-1}^{m} T_i}{m} \tag{11}$$

Results shown in Figures 10 and 11 calculate the mean value of executed offloaded tasks by fog nodes for varying fog node capacity and for varying numbers of offloaded tasks respectively. The results show that the mean trust value of executed offloaded tasks by TETES is slightly higher than the other three schemes because it discards the tasks offloaded by malicious nodes with trust values less than the threshold. MTV calculated by the other three schemes in all these results is slightly lower than TETES because MTV is calculated for only those offloaded tasks that are successfully executed and does not vary much with more or less number of executed tasks.

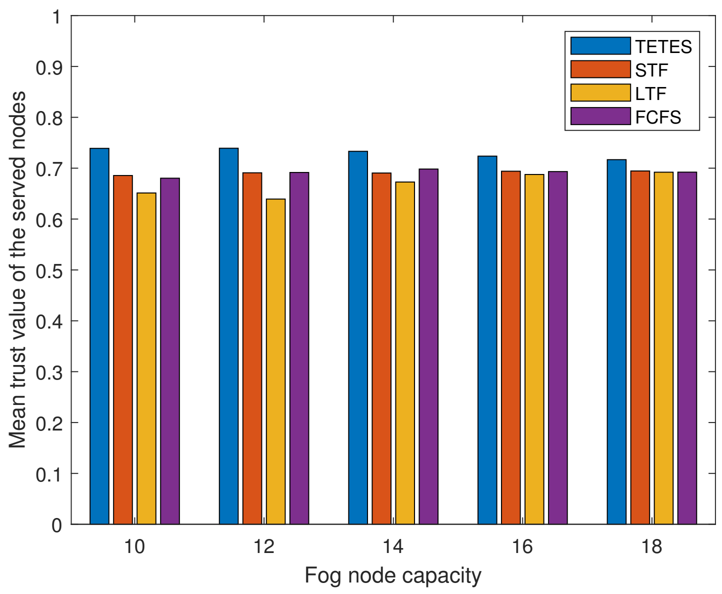

Figure 10. Mean Trust Value of Executed Tasks for Varying Capacity of Fog Nodes.

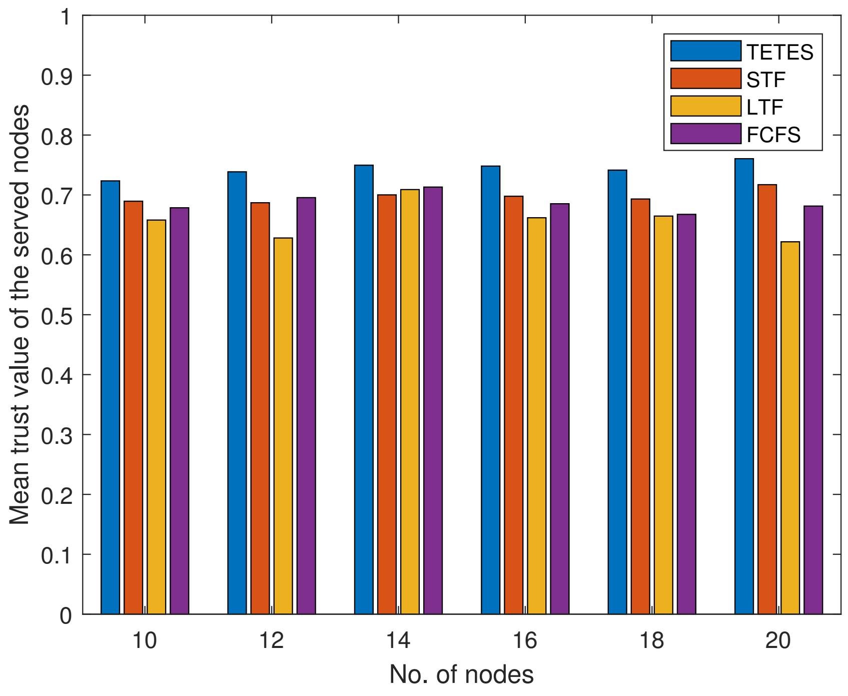

Figure 11. Mean Trust Value of Executed Tasks for Varying Number of Requested Tasks.

5.4. Discussion

As can be seen from the results, the proposed scheme improves the security and reliability of the IoT network by providing a trust based task computation mechanism. While the proposed technique offers improved trust level and higher rate of task execution, it requires sharing of control messages to compute trust in the network. Thus, enhanced reliability of task offloading is achieved at the expense of additional message overhead.

6. Conclusions

Malicious nodes disturb the quality of service of IoT-based wireless networks in smart city applications. Limited IoT nodes' processing capacity requires offloading of tasks to high computing nodes such as fog nodes. Malicious nodes present in the network try to disturb the performance of fog nodes by offloading complex natural tasks that require more computing power and resources. This compromises the execution performance of the fog node causing delay in task execution of legitimate nodes. TETES is a trust-based efficient task execution scheme for fog nodes, that discards the offloaded tasks executed by fog nodes and efficiently executes the tasks offloaded by legitimate nodes. The performance of TETES is compared with well-known STF, LTF, and FCFS algorithms in calculating number of tasks offloaded by trusted nodes, the executed data of offloaded tasks and the mean trust values for varying processing capacity of fog nodes and varying number of offloaded tasks. The results show that TETES executes 33%, 125%, and 60% more tasks as compared to STF, LTF, and FCFS respectively for varying processing capacity of fog nodes and 40%, 189%, and 93% tasks for increasing number of offloaded tasks as compared to STF, LTF, and FCFS respectively. It is evident from the results that the mean trust value of executed tasks in TETES is higher than any other scheme in all varying results. In future, we will develop a distributed and cooperative trust management scheme for task offloading in IoT networks.

Author Contributions: Conceptualization, A.N.A., B.A. (Bakhtiar Ali), M.S.S., M.A., D.A. and B.A. (Bushra Alghamdi); Writing—original draft, A.N.A., B.A. (Bakhtiar Ali), M.S.S. and M.A.; Writing—review & editing, D.A. and B.A. (Bushra Alghamdi). All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported and funded by the Deanship of Scientific Research at Imam Mohammad Ibn Saud Islamic University (IMSIU) (grant number IMSIU- RG23112).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Haseeb, K.; Saba, T.; Rehman, A.; Ahmed, Z.; Song, H.H.; Wang, H.H. Trust Management With Fault-Tolerant Supervised Routing for Smart Cities Using Internet of Things. *IEEE Internet Things J.* **2022**, *9*, 22608–22617. [CrossRef]
2. Bornholdt, H.; Röbert, K.; Kisters, P. Accessing Smart City Services in Untrustworthy Environments via Decentralized Privacy-Preserving Overlay Networks. In Proceedings of the 2021 IEEE International Conference on Service-Oriented System Engineering (SOSE), Oxford, UK, 23–26 August 2021; pp. 144–149. [CrossRef]
3. Alamri, B.H.S.; Monowar, M.M.; Alshehri, S. Privacy-Preserving Trust-Aware Group-Based Framework in Mobile Crowdsensing. *IEEE Access* **2022**, *10*, 134770–134784. [CrossRef]
4. You, S.; Radivojevic, K.; Nabrzyski, J.; Brenner, P. Trust in the Context of Blockchain Applications. In Proceedings of the 2022 Fourth International Conference on Blockchain Computing and Applications (BCCA), San Antonio, TX, USA, 5–7 September 2022; pp. 111–118. [CrossRef]
5. Popovic, D.; Gedawy, H.K.; Harras, K.A. FedTeams: Towards Trust-Based and Resource-Aware Federated Learning. In Proceedings of the 2022 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Bangkok, Thailand, 13–16 December 2022; pp. 121–128. [CrossRef]
6. Hriez, S.; Almajali, S.; Elgala, H.; Ayyash, M.; Salameh, H.B. A Novel Trust-Aware and Energy-Aware Clustering Method That Uses Stochastic Fractal Search in IoT-Enabled Wireless Sensor Networks. *IEEE Syst. J.* **2022**, *16*, 2693–2704. [CrossRef]
7. Li, F.; White, G.; Clarke, S. A Trust Model for SLA Negotiation Candidates Selection in a Dynamic IoT Environment. *IEEE Trans. Serv. Comput.* **2022**, *15*, 2565–2578. [CrossRef]
8. Manjula, P.; Baghavathi Priya, S. Detection of Falsified Selfish Node with Optimized Trust Computation Model In Chimp -AODV Based WSN. In Proceedings of the 2022 International Conference on Electronic Systems and Intelligent Computing (ICESIC), Chennai, India, 22–23 April 2022; pp. 52–57. [CrossRef]
9. Raizada, M. Survey on Recommender Systems Incorporating Trust. In Proceedings of the 2022 International Conference on Applied Artificial Intelligence and Computing (ICAAIC), Salem, India, 9–11 May 2022; pp. 1011–1015. [CrossRef]
10. Gupta, G.; Mangla, N. Trust Aware Multi-Objective Metaheuristics for Workflow Scheduling in Cloud Computing. In Proceedings of the 2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COM-IT-CON), Faridabad, India, 26–27 May 2022; Volume 1, pp. 602–609. [CrossRef]
11. Mendonça, F.; Abdennadher, N.; El-Maliki, T.; Poleggi, M.E. Context-aware Trust Metrics for non critical IoT Applications: an Intrinsic Data Quality approach. In Proceedings of the 2022 13th International Conference on Information and Communication Systems (ICICS), Irbid, Jordan, 21–23 June 2022; pp. 109–114. [CrossRef]
12. Alvi, A.N.; Khan, S.; Javed, M.A.; Konstantin, K.; Almagrabi, A.O.; Bashir, A.K.; Nawaz, R. OGMAD: Optimal GTS-Allocation Mechanism for Adaptive Data Requirements in IEEE 802.15.4 Based Internet of Things. *IEEE Access* **2019**, *7*, 170629–170639. [CrossRef]
13. Khan, S.; Alvi, A.N.; Javed, M.A.; Bouk, S.H. An enhanced superframe structure of IEEE 802.15.4 standard for adaptive data requirement. *Comput. Commun.* **2021**, *169*, 59–70. [CrossRef]
14. Rahim, M.; Ali, S.; Alvi, A.N.; Javed, M.A.; Imran, M.; Azad, M.A.; Chen, D. An intelligent content caching protocol for connected vehicles. *Emerg. Telecommun. Technol.* **2021**, *32*, 1–14. [CrossRef]
15. Martinez, I.; Hafid, A.S.; Jarray, A. Design, Resource Management, and Evaluation of Fog Computing Systems: A Survey. *IEEE Internet Things J.* **2021**, *8*, 2494–2516. [CrossRef]
16. Rahim, M.; Javed, M.A.; Alvi, A.N.; Imran, M. An efficient caching policy for content retrieval in autonomous connected vehicles. *Transp. Res. Part A Policy Pract.* **2020**, *140*, 142–152. [CrossRef]
17. Han, Y.; Hu, H.; Guo, Y. Energy-Aware and Trust-Based Secure Routing Protocol for Wireless Sensor Networks Using Adaptive Genetic Algorithm. *IEEE Access* **2022**, *10*, 11538–11550. [CrossRef]
18. Mirzadeh, I.; Sayad Haghighi, M.; Jolfaei, A. Filtering Malicious Messages by Trust-Aware Cognitive Routing in Vehicular Ad Hoc Networks. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 1134–1143. [CrossRef]
19. Wang, J.; Yan, Z.; Wang, H.; Li, T.; Pedrycz, W. A Survey on Trust Models in Heterogeneous Networks. *IEEE Commun. Surv. Tutorials* **2022**, *24*, 2127–2162. [CrossRef]
20. Kim, D.Y.; Alodadi, N.; Chen, Z.; Joshi, K.P.; Crainiceanu, A.; Needham, D. MATS: A Multi-aspect and Adaptive Trust-based Situation-aware Access Control Framework for Federated Data-as-a-Service Systems. In Proceedings of the 2022 IEEE International Conference on Services Computing (SCC), Barcelona, Spain, 10–16 July 2022; pp. 54–64. [CrossRef]
21. Guo, J.; Liu, A.; Ota, K.; Dong, M.; Deng, X.; Xiong, N.N. ITCN: An Intelligent Trust Collaboration Network System in IoT. *IEEE Trans. Netw. Sci. Eng.* **2022**, *9*, 203–218. [CrossRef]

22. Shen, X.; Lv, W.; Qiu, J.; Kaur, A.; Xiao, F.; Xia, F. Trust-Aware Detection of Malicious Users in Dating Social Networks. *IEEE Trans. Comput. Soc. Syst.* **2022**, *10*, 2587–2598. [CrossRef]
23. Dang, T.D.; Hoang, D.; Nguyen, D.N. Trust-Based Scheduling Framework for Big Data Processing with MapReduce. *IEEE Trans. Serv. Comput.* **2022**, *15*, 279–293. [CrossRef]
24. Varadharajan, V.; Karmakar, K.K.; Tupakula, U.; Hitchens, M. Toward a Trust Aware Network Slice-Based Service Provision in Virtualized Infrastructures. *IEEE Trans. Netw. Serv. Manag.* **2022**, *19*, 1065–1082. [CrossRef]
25. Ebrahimi, M.; Haghighi, M.S.; Jolfaei, A.; Shamaeian, N.; Tadayon, M.H. A Secure and Decentralized Trust Management Scheme for Smart Health Systems. *IEEE J. Biomed. Health Inform.* **2022**, *26*, 1961–1968. [CrossRef] [PubMed]
26. Li, Z.; Chang, V.; Hu, H.; Yu, D.; Ge, J.; Huang, B. Profit maximization for security-aware task offloading in edge-cloud environment. *J. Parallel Distrib. Comput.* **2021**, *157*, 43–55. [CrossRef]
27. Wang, J.; Chang, V.; Yu, D.; Liu, C.; Ma, X.; Yu, D. Conformance-oriented predictive process monitoring in BPaaS based on combination of neural networks. *J. Grid Comput.* **2022**, *20*, 25. [CrossRef]