IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 22, 2019, accepted December 3, 2019, date of publication December 9, 2019,
date of current version December 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2958355

IoT Public Fog Nodes Reputation System: A Decentralized Solution Using Ethereum Blockchain

MAZIN DEBE, KHALED SALAH, MUHAMMAD HABIB UR REHMAN, AND DAVOR SVETINOVIC

Center for Cyber-Physical Systems, Electrical Engineering and Computer Science, Khalifa University of Science and Technology, Abu Dhabi, UAE

Corresponding author: Davor Svetinovic (davor.svetinovic@ku.ac.ae)

This work was supported by the Center for Cyber-Physical Systems, Khalifa University of Science and Technology, UAE.

ABSTRACT Public fog nodes extend cloud services for the Internet of Things (IoT) clients and smart devices to provide additional computation capabilities, storage space, and reduce latency and response time. The openness and pervasiveness of public fog nodes leads to the requirement of using trust models to ensure reliability, security, privacy, and meet the service-level agreements (SLAs). Conventional trust models for public fog nodes are centrally configured, deployed, and maintained considering security, privacy, and SLA requirements. However, these trust models enforce centralized governance policies across the system which leads towards the single-point-of-failure and single-point-of-compromise over IoT devices' and users' personal data. This paper proposes a decentralized trust model in order to maintain the reputation of publicly available fog nodes. The reputation is maintained considering users' opinions about their past interactions with the public fog nodes. The proposed trust model is designed using public Ethereum blockchain and smart contract technologies in order to enable decentralized trustworthy service provisioning between IoT devices and public fog nodes. The proposed approach is tested and evaluated in terms of security, performance, and cost. The results show that using blockchain for decentralized reputation management could become more advantageous when compared to the existing centralized trust models.

INDEX TERMS Blockchain, fog computing, IoT, reputation, trust.

I. INTRODUCTION

Fog computing is an architecture that has been created to mediate between the cloud servers and their clients in order to support the Internet of Things (IoT) devices (see Fig. 1). IoT devices generally have weak computation power, minimal storage and limited network capabilities. Fog devices offer an external source for additional processing power and storage space [1]. With the large amount of data generated by IoT devices, fog nodes ensure better performance by reducing latency and optimizing the connection between the IoT clients and their service providers [2]. Fog nodes usually reside very close to the IoT devices (typically on nearby routers or switches). They provide a reliable connection to process, filter and store IoT data streams before forwarding them to the cloud service providers [3].

The associate editor coordinating the review of this manuscript and approving it for publication was Giovanni Merlino.

Public fog nodes are susceptible to various security and privacy attacks such as ransomware attacks, distributed denial of service (DDoS) attacks, data, location and user privacy, verifiable computing, access control, and network intrusions [4], [5]. Apart from these aforementioned attacks, embedding trust models in existing fog computing architectures is a major challenge for fog cloud service providers (FCSP). Many FCSP provide hardware-based trusted fog computing systems using trusted execution environment (TEE), secure element (XE), or trusted platform module (TPP). These platforms are tightly bounded with specific application scenarios. However, generic trust models for public fog nodes are still unexplored. We define the trust as an outcome of a reputation mechanism employed across fog computing system whereby a trust model maintains the reputations of the public fog nodes and it enables the IoT devices to rely upon them [6].

Currently, fog computing is mostly employed in scenarios where trust is not a major concern. Entities that use fog

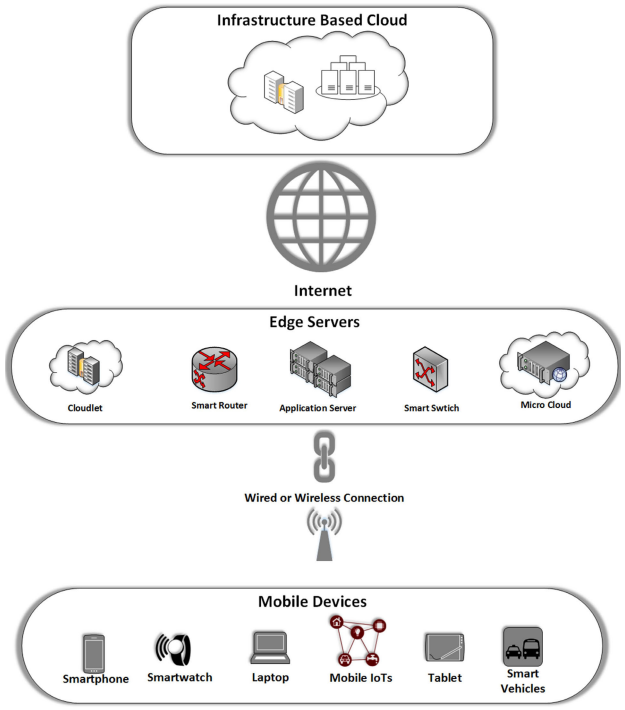


FIGURE 1. Fog cloud computing architecture.

nodes are often the ones who collect, process, and store data. Therefore, all participating members implicitly trust each other and do not need an extra overhead in establishing confidence in the data. For example, if a government provides fog nodes to gather data from various sensors or an institution like a hospital deploys fog nodes to serve their devices, they do not need to necessarily account for the trust in their system. However, the trust issues emerge when fog nodes are handling data for another party. This scenario occurs when fog computing is introduced in a public setting. Fog nodes that serve public, whether they are stationary or moving, are expected to be the points of mistrust to their clients. This requires the fog nodes to prove their ability to provide good quality of service (QoS) especially when they will likely have a direct access to the client's data. Another scenario that requires trust in fog nodes is the availability of multiple fog nodes in a single location. Different fog nodes may provide similar services but differ in the provided QoS. The physical constraints such as storage space and the processing power of the fog node in addition to the bandwidth provided all affect the performance of the fog nodes [7]. One of the common existing trust models uses reputation systems [8] which evaluate the nodes based on specific metrics. However, these centralized systems are often used to keep track of IoT clients which result in a single point of failure.

To maintain a sufficient level of security and privacy, the reputation systems often deploy analytical solutions to meet the SLA requirements [9], [10]. These algorithms and protocols seek to prevent the system attacks and the clients data compromise. Using a technology like the blockchain changes the approach to managing the systems security and privacy. Being decentralized, the blockchain removes the single point of failure and eliminates security threats such as the DDoS attacks. Moreover, all blockchain transactions are anonymous as only the account addresses involved in the transaction are visible to the public. Furthermore, blockchain is a distributed ledger which means that the user data is geographically dispersed over multiple devices. This data redundancy provides safeguards against data loss [11].

This paper presents the design and implementation of a decentralized reputation system to ensure trust in public fog nodes using smart contract technologies on public Ethereum blockchain. We developed a solution that can be employed for multiple fog nodes and multiple clients. The functionality offered is expanded and assigned to the corresponding smart contract as opposed to the single contract solution for a single fog node that was developed in our previous work [12]. The solution presented in this paper covers more complex realistic use case scenarios, and it has been tested more thoroughly. In brief, the main contributions of this paper are:

- We designed and simulated a decentralized blockchain-oriented solution to deploy a reputation-based trust model for IoT-fog applications.
- We incorporated a revised reputation scores computation technique that combines the feedback from clients with an assessment of the client's opinion about their previous interactions with public fog nodes.
- We analyzed the logic of determining the reputation score of the fog nodes along with the credibility score of the evaluating client.
- We evaluated the proposed decentralized trust model in terms of cost, performance, and security.

The remainder of the paper is organized as follows: Section II presents a concise explaination of the fog computing, IoT, and blockchain technologies. Section III discusses some of the previous work done in the areas of trust and reputation systems. Section IV outlines the design and methodology of the reputation system. Section V presents the implementation details of the system. Section VI presents the results of testing each contract, and the analysis of the costs and performance. Section VII presents the conclusions.

II. BACKGROUND

This section presents a concise discussion on IoT, fog computing, and blockchain technologies.

IoT devices have gained significant popularity over the past 20 years, so much that Cisco is expecting around 20 billion connected IoT devices by the year 2020 [13]. The contribution of IoT is evident as shown by the number of possible applications of these devices. IoT devices are useful in vehicles, personal accessories, home appliances, medical equipment, robotics, cyber-physical systems, Industry 4.0 and many other application areas [14]. Although being resource-constrained devices intended for a simple task or a small set of tasks, they generate substantial amounts of data to be transferred to the external cloud service providers. With

the increasing numbers of IoT devices, this load of data is exponentially growing and handling it is becoming more and more challenging. To reduce the load off the cloud servers, an intermediary layer between the cloud service providers and their clients has been introduced, known as fog computing infrastructure.

Fog computing is a layer that acts as a transient computing system and an intermediary communication channel between centralized cloud servers and their client IoT devices. The fog nodes are normally routers or switches at the edge of the network situated significantly closer to the clients than the cloud servers. Fog nodes have been developed to aid IoT devices which require fast connection and low latency in mobile environments. With cloud servers, the physical distance together with the bandwidth constraint and the high traffic load lead to delayed responses which are highly undesirable. Fog nodes analyze data for the IoT clients, oversee them, provide storage, and enhance their performance.

Many companies have been delivering their services through cloud servers due to their high efficiency, scalability, and the ease of access. However, latency and bandwidth greatly affect the performance of cloud services. Most cloud services offer scalable solutions, offering increasing processing power with higher load. Nonetheless, large bulks of data are bound to affect the communication with servers which is highly undesirable. The paper in [1] presents multiple scenarios where the concept of fog computing has been adopted in different application areas such as smart grid, smart cities, and VANET.

Blockchain technology appeared with the introduction of the Bitcoin in 2009 [15]. Bitcoin became extremely popular because it solved the problems of centralization and double-spending, which are the major privacy and security concerns in centralized banking systems. Considering its potential, people started utilizing the blockchain technology in various fields like in IoT, asset management, AI, security and authentication systems, as in [5], [16]–[18]. Blockchain is a decentralized ledger that is essentially a chain of blocks, each holding a number of transactions [19]. This chain is available for the public and is continuously expanding as miners add more blocks to it. Unlike traditional databases, the same stored set of data is available at all the nodes. This eliminates the single point of failure and increases protection against certain cyber-security attacks.

Each transaction is agreed upon via a consensus protocol. The consensus protocol is a mechanism used by the blockchain to add a layer of trust to the network. One of the most popular consensus protocol is the proof-of-work (PoW). PoW is used in the Bitcoin blockchain where miners compete by solving a computationally challenging problem in order to add their block of transactions to the blockchain. Miners typically get rewards as incentives to provide mining power to Bitcoin network. Following the successful adoption of Bitcoin, many other consensus protocols including Proof-of-Stake (PoS), Proof-of-Authority (PoA), Proof-of-Burn (PoB), Proof-of-Elapsed Time (PoET) and others [20] have been implemented by different blockchain operators. Some blockchain networks cannot be edited to match the requirements of another system; in other words, they are not programmable. However, a fork from the Bitcoin called the Ethereum blockchain is an epitome of a blockchain that employs smart contracts that make it customizable. A smart contract is a piece of software that executes a certain logic and generates transactions for underlying blockchain network. Smart contracts of Ethereum are written in a high level JavaScript-like language called Solidity. To enable IoT devices to use the blockchain functionality, Decentralized Applications (DApps) are used. DApps represent the application layer or the frontend for the devices to access the blockchain. DApps are compatible with IoT devices and they can be used to connect, access, and manage IoT data.

III. RELATED WORK

This section analyzes some of the previous works on the topic as well as some implementations of reputation based trust models using blockchain technologies.

Reputation is often used to evaluate anything from web services to merchandise efficiently and effectively. The challenge in reputation systems is finding an appropriate optimized algorithm for adequate evaluation. Q-rater is a reputation system that targets electronic business models [21]. First, it measures the credibility of each user based on their experience, reliability and co-orientation. These parameters help in increasing or decreasing the user's credibility and thereby affecting his contribution when analyzing the reputation. Next, a group of best qualified raters is filtered out to be considered for the final reputation which is computed from the feedback provided by the most credible raters.

RateWeb is a system that rates web services from evaluations given by separate raters [22]. A group of raters evaluate each service and adjust the reputation score accordingly. Additionally, a very valuable credibility score for each rater is computed by analyzing all given reputation scores and how they relate to each other and then comparing them to the scores given by each rater. However, this process may be computationally expensive and could be infeasible to be employed on the blockchain.

Research in blockchain points out that adding a layer of logic (using smart contracts) that keeps track of the reputation of service providers in a decentralized manner is promissing and it can be implemented to establish increased trust. The design of the blockchain makes sure that these reputations are agreed upon and verified via its consensus protocol [23]. The system described in [23] categorizes a transaction as acceptable or unacceptable depending on the feedback of the client. This classification depends on the context but it can just mean whether the transaction has been executed properly or not. Furthermore, miners communicate with the nodes being evaluated to verify the transaction.

Another technique for evaluating reputation is to merge the client's judgment from past experiences along with the calculation of the reputation of the service provider [24].

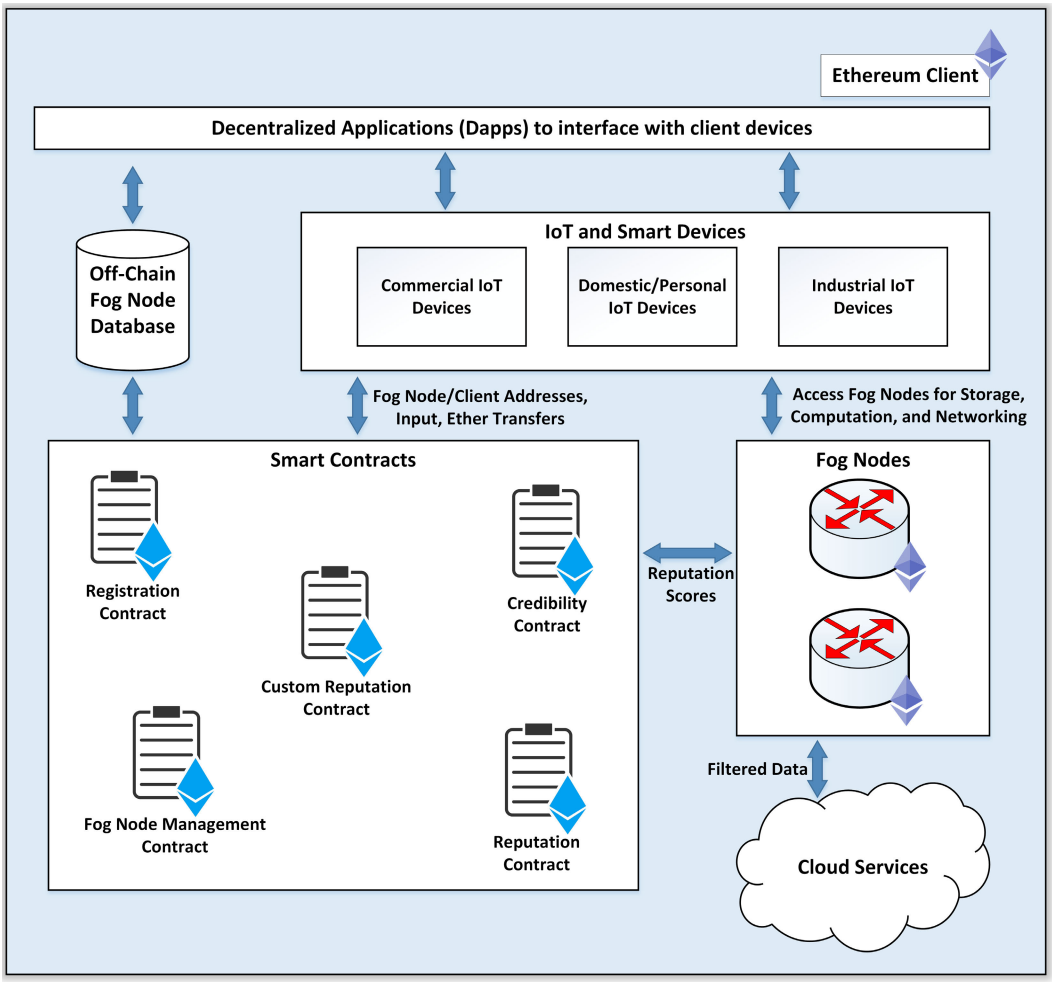


FIGURE 2. Proposed system architecture.

The positive reputation is expressed as part of the reputation that takes into consideration the experience and the values of supportive experience and the negative reputation expresses negative experience. The experience described is initially set to a specific value and changes according to feedback. Positive or cooperative feedback increases the experience value while non-cooperative feedback decreases the experience. Moreover, due to the lack of interaction between the involved nodes the experience decreases or ''decays'' hence requires information ''freshness''. It is notable from these examples that there is a need for a more robust reputation system to be developed using smart contracts to provide more trust in the services/entities being evaluated and their evaluators.

With the emergence of peer-to-peer networks, trust and reputation systems have been popular research topics. Nevertheless, a lot of the developed solutions are centralized solutions. Therefore, in this research paper we are presenting a decentralized solution based on the blockchain which is an extension of our previous work [12] that showcased a preliminary design of the system. In this paper, we are presenting a more detailed architecture, updated algorithms and more extensive testing and analysis.

IV. PROPOSED APPROACH

This section presents a detailed explanation of the proposed solution to establish trust in fog nodes.

A. SYSTEM OVERVIEW

IoT client devices need to establish trust in fog nodes that they use for storage, computation, or communication with centralized cloud data centers. We define an architecture that provides IoT client devices with the means to choose the best suited available fog nodes. As shown in Fig. 2, the system consists of the fog nodes to be evaluated, the client IoT devices that evaluate the fog nodes, and the Ethereum smart contracts that govern the communication and interaction between fog nodes and their connected client devices. The client devices use decentralized application (DApp) at the front-end in order to communicate and interact with blockchain back-end. Further discussion on the the main components and their mutual interactions is given below.

- **Fog node:** The fog nodes on the Ethereum blockchain are assigned with a unique Ethereum address EA) whereby an EA is linked with metadata (such as IP address, location, processing capabilities, average

up-time, application registry, availability of wireless interfaces and communication protocols) about their corresponding fog nodes. To evaluate the trustworthiness of each node, a list of evaluations by clients is logged and associated with that fog node's EA. These logs comprise the information about each quality being evaluated such as latency, storage, and cost of using fog services, in addition to a generic reputation score for the fog node. With each evaluation submission, the evaluator's EA is also recorded. After each score submission done by IoT client, the evaluators' lists are updated. The clients evaluate each metric of the fog node based on the interaction which directly affects its reputation score.

- **IoT client:** Following an interaction with a fog node, IoT devices provide their feedback about the fog node's performance according to SLA requirements. These evaluations are used to compute the reputation score of the fog node. The client devices need to be held accountable for the ratings they provide. To cultivate accountability, the credibility of each IoT device is considered to ensure honesty and fairness when providing reputation scores. This is done using three metrics: trustworthiness, consistency and number of provided ratings. Trustworthiness expresses the agreement of the ratings provided by this node with the ratings provided by rest of the IoT devices. Although, this gives a reasonable prediction about the intention of the rater, but several nodes can collude together and provide false scores [25]. The consistency refers to the agreement between the values provided by a certain IoT device and the previous values provided by the same IoT device for a specific fog node. This is done to avoid random fluctuations of reputation scores and malicious behavior. Moreover, a node that has submitted a lot of ratings is given more importance and has more influence on the reputation of the fog nodes. Because credible raters have a big effect on the reputation values, they are rewarded better than the new or malicious clients. Subsequently, when credible clients provide false ratings, they are punished more severely as well. While this method mitigates the harm of malicious clients, it provides no real incentive for them to give genuine feedback. An additional deposit is also required by the fog node for this purpose. This deposit is only returned to the client upon providing a proof of good intent. Moreover, The IoT device can calculate a tailored reputation according to its preference of each of the qualities that are being evaluated. For instance, an IoT client can give more weight to the latency in contrast to other desired properties.
- **Smart contracts:** There are five smart contracts in the proposed approach that govern the interaction between the clients and the fog nodes: Client Registration Contract, fog Node Management Contract, Reputation Contract, Credibility Contract and the Custom Reputation Contract. Both the fog node and the IoT client first register at the respective contract that keeps track of all the connected devices. After registering the clients and fog nodes, the reputation contract takes in the input of the IoT device and appends it to the previous scores. Then, this contract divides all evaluations into clusters and picks the most popular reputation. This can be done by either calculating the centroid of the most popular cluster or an actual value that is considered to be the head of the cluster; we use the latter. The Credibility contract calculates the credibility of the rater based on the aforementioned metrics. The Custom Reputation Contract is used for providing special reputation values based on a specific preference by the client.

B. CALCULATING REPUTATION

IoT devices submit ratings after connecting with and executing the required services on proximal fog nodes, and then the process of computing reputation is started on chain. At this stage, the reliability and redundancy of reputation scores provided by each IoT device are considered. When the IoT devices and fog nodes first register themselves, they are given an initial credibility and reputation score respectively. These scores are public and not hidden from all participating IoT devices and fog nodes so that all participants know exactly the reputation and credibility of fog nodes to meet the SLA requirements. The reputation manager groups the ratings for the different properties such as latency, availability, and cost of using fog node services and divides them into clusters. Each cluster of those ratings is headed by a representative rating based on the median value of the provided ratings. Afterwards, the most populated cluster is selected and its corresponding ratings are stored for reputation calculations. The best cluster is not chosen only based on the number of scores but also on the credibility of its contributors. This means that a cluster is the most popular if it has the highest number of ratings by reputable raters. It is worth noting that the obtained rating is not yet the reputation value assigned to the fog node. The actual assigned reputation is an accumulation of the reputation scores provided from all clients for different properties paired with the credibility of the raters. This allows giving more weight to the most valuable raters.

After determining the reputation of the fog node, the system involves the client as well. The currently submitted score is compared with the value of the best cluster previously calculated. For this purpose, the scores are not compared with the assigned general reputation of the node but rather with the head of the best cluster. This comparison yields the trustworthiness factor which accounts for the closeness of this client's ratings with the majority. Previous scores, if available, are compared to the latest scores and the consistency factor is determined. When a client is deemed trustworthy, his credibility increases. Otherwise, the credibility of the provider of a reputation score that is far from other scores decreases. Closeness is favored over trustworthiness because the performance of the fog device could have actually deteriorated or improved from the last transactions. As long as this score diverges from the majority, it needs not to be trusted.

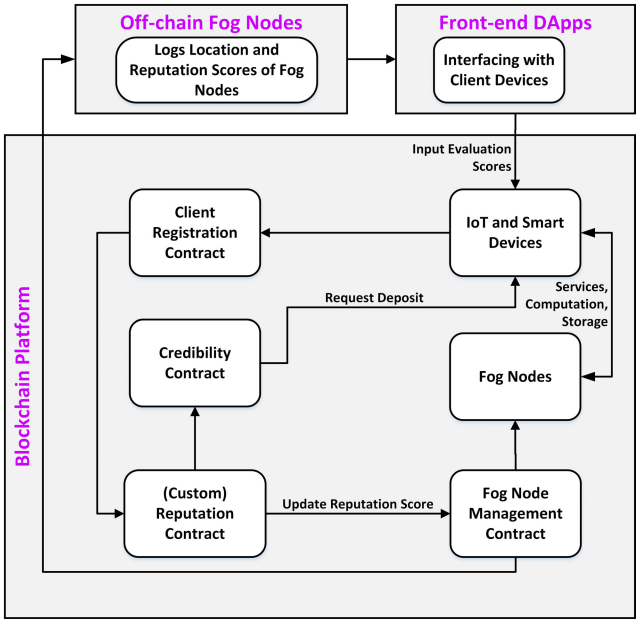


FIGURE 3. Workflow for computing reputation of fog nodes.

V. IMPLEMENTATION

The aforementioned system architecture was implemented for validation and testing. The back-end was developed and tested on Remix IDE using solidity language. Remix IDE is an online tool to develop, debug and test code on a virtual Ethereum blockchain. The front-end was developed on Truffle Suite that acts as an interface for accessing the blockchain layer.

As mentioned earlier, several contracts were deployed each for a specific purpose. The smart contracts manage and control all clients and fog nodes in the network. They register devices and evaluate the reputation scores of fog nodes based on the input from their clients as well as the credibility of those clients. Information about fog devices and IoT clients are kept hidden from others and only necessary data is exposed.

As shown in the workflow in Fig. 3, the fog and IoT devices are first registered with their Ethereum addresses in the corresponding smart contracts. The off-chain database of fog nodes contains information about fog nodes' Ethereum addresses linked to their locations. Client IoT devices provide their feedback regarding the performance of a fog node which requires the fog node reputation contract and the client credibility contract to re-evaluate the relevant scores.

Fig. 4 shows how different entities are related to each other. The FogNode entity maps the address of the fog node to multiple arrays of the most recent evaluations for the attributes such as response_time and availability. The address of each rater and the location of each fog node is stored. The ClientNode maps the address of each IoT client to its credibility score, balance, and number of ratings. Fog nodes are registered on the fog node management smart contract only by the contract owner.

Meanwhile, all clients can access the Client Registration contract to self-register as an IoT client. Clients can also send their evaluation scores to the reputation contracts. Reputation and custom reputation smart contracts hold the threshold values and methods to compute the reputation scores. The reputation contract also provide input for the credibility contract methods to compute credibility of IoT devices.

Fig. 5 exhibits the interactions between IoT devices and fog nodes on the Ethereum blockchain starting from the evaluation of the fog node by its client to the computation of reputation and credibility scores. The fog node and the client involved in the interactions need to register at the corresponding smart contract. As clients register, they transfer a deposit to ensure they provide truthful evaluations. Client IoT devices evaluate the fog nodes and submit their reputation ratings to reputation smart contract that validates them and proceeds to compute the reputation. The result of this process is broadcasted to all devices as events. Then the reputation contract provides ratings to the credibility contract. Lastly, the off-chain database is updated with the reputation and credibility scores.

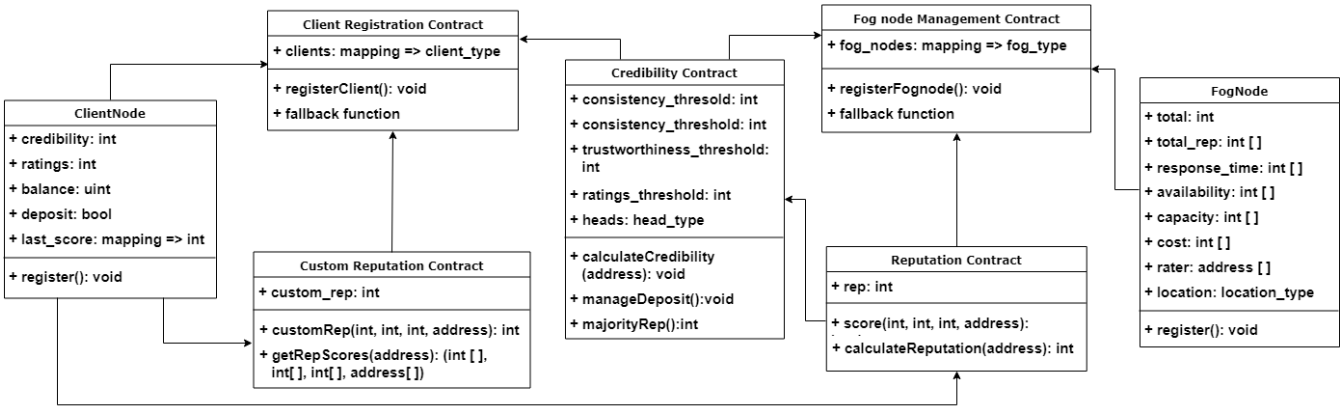


FIGURE 4. Relationship among different entities of the smart contract.

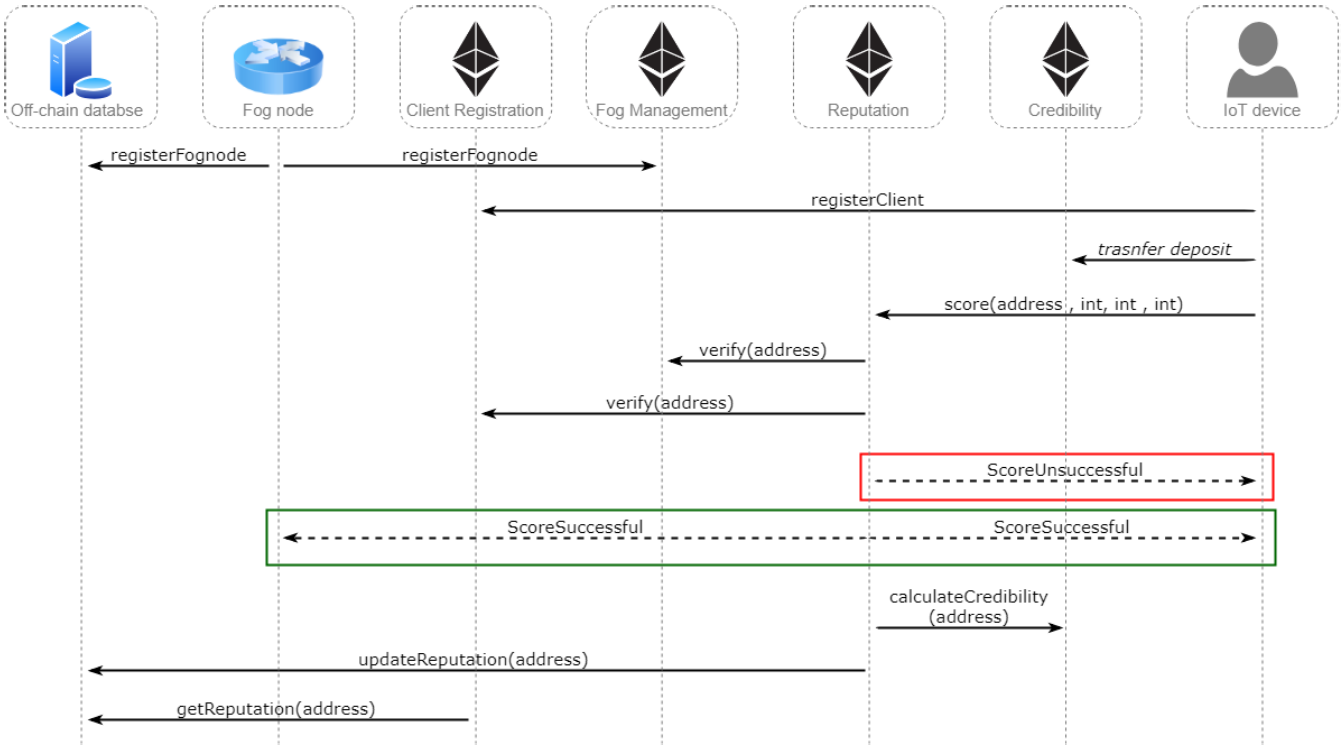


FIGURE 5. Interactions showing the function calls for registration, scoring, verification and reputation calculation.

A. CLIENT REGISTRATION CONTRACT

A client needs to register in order to retrieve the addresses of the fog nodes in his proximity and benefit from their services. Because the client's Ethereum account is an externally owned account (EOA), it can only hold and transfer money to other accounts. Therefore, a client transfers a deposit to the EA of the Client Registration smart contract. If the sender's address (the client) has not already registered and the amount transferred is correct, the smart contract creates a mapping for this client's EA and accepts the deposit to ensure it does not behave maliciously. If the client already exists, the deposit would have been needed because the credibility of the client has dropped after its deposit has been returned. This process can be seen in Algorithm 1. The client has a DApp frontend that is connected to the Ethereum blockchain as its backend. A request from the DApp transfers the deposit to the Client Registration smart contract for registration. To have access to the fog nodes available in the client's proximity, the client contacts the off-chain database. The database provides the EAs of fog nodes along with their metadata.

Algorithm 1 Client Registration

```
Account of client transfer to account of smart contract.
Verify the value transferred.
Verify if client exists
if value is incorrect then
    Transfer rejected.
    Revert.
else
    if client already exists then
        /* Client's credibility dropped.
           Deposit required.                */
        Transfer received.
        Client allowed to access fog node services.
    else
        Initialize client's data.
        Link data to address of sender.
        Append to list of clients.
    end
end
```

B. FOG NODE MANAGEMENT CONTRACT

Just like IoT clients, fog nodes also hold EA for an EOA account. However, the fog registration is not autonomous and can only be done by the owner of the smart contracts. Upon initialization, the fog node is given an initial reputation value which keeps changing considering the ratings provided by IoT clients. Primarily, the fog node data (i.e. their EAs and metadata) are stored in off-chain database. This database also holds information about client deposits and their access rights for fog nodes.

C. REPUTATION CONTRACT

Clients contact fog nodes to benefit from their services. In a public setting, several fog nodes are generally available

with different QoS-compliant configurations. After interacting with the fog nodes, clients evaluate those nodes based on their latency, cost and/or any other metric that affects them individually. The reputation score of any fog node is derived from the evaluation of clients. Due to the fact that some clients may behave in an ill-behaved manner, the integrity of those clients is taken into consideration. The reputation is hence affected more by honest and reputable clients. Therefore, clients would tend to evaluate the fog providers fairly to increase the significance of their assessment. Unlike the credibility of clients, reputation scores are not affected by the cold start problem. Reputation scores are the result of the accumulation of all scores provided by clients along with their credibility scores. As such, no initial reputation values are required for fog nodes. Moreover, the level of honesty of clients do not affect the initial reputation scores greatly as all clients are still new to the system. The reputation score given to a fog node is calculated as shown in Eq. 1.

$$Rep(fn) = \sum_{n=0}^{N} Cr(n) * Rep_n(fn) \tag{1}$$

fn is the address of the fog node being evaluated.

$Rep_n(fn)$ is the reputation of node fn provided by client n.

$Rep(fn)$ is the total reputation of node fn.

$Cr(n)$ is the credibility of client n.

N is the number of raters of fog node fn.

$Rep(fn)$ now holds the reputation score for the fog node. To normalize the score, it is divided by the total credibility of its raters as shown in Eq. 2.

$$Rep(fn) = \frac{Rep(fn)}{\sum_{n=0}^{N} Cr(n))} \tag{2}$$

D. CREDIBILITY CONTRACT

This smart contract is concerned with finding the intentions of the client based on its feedback. To account for the cold start problem, clients are giving a credibility score of 80. This initial score is then modified depending on consequent score submissions provided by the client. To find whether the client is giving honest feedback about the fog node, we compare it to the evaluation scores of the most reputable clients. As explained in algorithm 2, the reputation scores provided for this fog node are grouped into clusters. This is done by assigning cluster heads to each group of scores within a specific range from the cluster head. If the new score does not belong to any cluster, it forms a cluster on its own. The cluster head can be determined either by taking a member of the cluster that is approximately in the middle or taking the centroid of the values. In this implementation, we use the centroid. The credibility of each cluster is calculated from the credibility of its members. Having many similar scores by unreliable clients does not mean that their cluster will be the most popular one. The centroid of the cluster whose raters have the highest credibility is considered to be the majority reputation. The credibility of the cluster increases when its members are reputable. This majority reputation score is considered trusted and used as a benchmark for any new evaluation.

Algorithm 2 Evaluating Client Credibility and Deposit Management

Input: Fog_node
1 Fog_node is an address.
2 Split reputation scores into clusters.
3 Find cluster with highest credibility score.
4 Find centroid of the cluster.
5 Compute consistency.
6 Compute trustworthiness.
7 Credibility of client is modified and normalized.
8 **if** *client has good credibility* $\wedge$ *has remaining balance* **then**
9 | Return deposit to client.
10 **else**
11 | Deny client from fog services.
12 **end**

The consistency of the client in providing ratings is also an evidence of reputation as the QoS of a fog node is not expected to fluctuate regularly. With each positive rating, the rater's credibility increases. It is worth noting that a rater whose ratings are agreeing with the majority but are inconsistent typically indicates a decay in the performance of the fog node being evaluated. In addition, the more ratings a client provides the more his credibility grows due to his positive ratings. On the other hand, if a reputable client with high number of ratings starts acting maliciously, his credibility score drops rapidly. This technique helps to stop bad clients from affecting the reputation score of fog nodes but it still does not give them enough incentive to correct their behavior. Therefore, clients are asked to transfer a deposit that can only be returned once they have proven the reliability of their scores. Even after returning their deposit, their behavior is still monitored and failing to cooperate will require them to transfer a deposit. These interactions are shown in Fig. 6.

E. CUSTOM REPUTATION CONTRACT

Clients often have different priorities when choosing a service provider. Some clients that offer real time services may compromise the high cost of a fog node for the highest latency available while others might not have the same requirements. The generic reputation score assigned to the fog node accounts for all the metrics that clients evaluate the fog node on. If the client requires a customized reputation score for a fog node based on their preference, the client provides the smart contract with a preference score for each metric and the reputation is adjusted accordingly. The reputation score is calculated as mentioned earlier in Eq. 1 and Eq.2, but with adapting it to the preference of the client as shown in Eq. 3.

$$Rep(fn) = \sum_{n=0}^{N} p_n(0) * Rep_n(fn, 0) + p_n(1) * Rep_n(fn, 1) + \cdots + p_n(M) * Rep_n(fn, M) \tag{3}$$

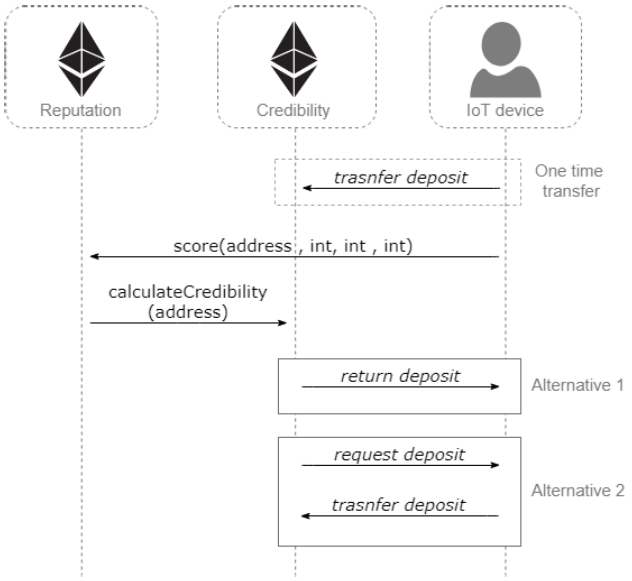


FIGURE 6. Interactions that govern handling the deposit.

$p_n(m)$ is the preference score from client n of metric m.

$Rep_n(fn, m)$ is the reputation score of node fn give by client n for metric m.

M is the number of metrics to be evaluated.

VI. TESTING AND EVALUATION

This section provides details about validating and verifying the logic explained to establish trust in public fog nodes.

A. VALIDATION

All smart contracts were deployed at the same address to provide accessibility of client and fog nodes' data from any contract. Initially, two clients with EAs "0xCA35b7d915458EF540aDe6068dFe2F44E8fa733c" and "0x14723A09ACff6D2A60DcdF7aA4AFf308FDDC160C" evaluate the performance of a fog node with the EA "0x4B0897b0513fdC7C541B6d9D7E929C4e5364D2dB". After validating the functionality of the smart contracts, more clients and fog nodes were added incrementally. The smart contracts were subjected to extensive testing with numerous scenarios that resemble the real life cases.

1) REGISTRATION CONTRACTS

The fog node and client registration contracts hold the information of the registered fog nodes and clients respectively. When a device makes a transfer to one of these contracts, the contract registers the device by its EA. In the case of a client device registration, the EA is linked to data such as an initial credibility score and the deposit amount available. The fog node does not have any initialized data upon registration as no client has evaluated it yet. However, another major difference is that a fog node cannot register itself as shown in Fig. 7. If a fog node tries to access the registration function, the transaction will be reverted as shown in Fig. 8.

```
modifier onlyOwner(){
    require(
        msg.sender==owner,
        "Sender not authorized."
    );
    _;
}
mapping (address => fog_type) public fog_nodes;
function register_fog(address fog_node) public onlyOwner{
    require(
        !fog_nodes[fog_node].exists,
        "Fog node exists"
    );
    fog_type memory temp;
    temp.exists = true;
    fog_nodes[fog_node] = temp;
}
```

FIGURE 7. Registering fog nodes is limited to contract owner.

```
[vm] from:0x147...c160c
to:reg_fog.register_fog() 0x692...77b3a
value:0 wei data:0x86b...17d17 logs:0
hash:0x331...cb6f5
Debug

transact to reg_fog.register_fog errored: VM error: revert.
revert  The transaction has been reverted to the initial state.
Reason provided by the contract: "Sender not authorized.".
Debug the transaction to get more information.
```

FIGURE 8. Restricting fog nodes to self-register.

2) REPUTATION CONTRACT

The reputation contract collects evaluations from clients, verifies them and appends them to the list of evaluations of the target fog node. If the address of the evaluator or the fog node to be evaluated does not exist, the transaction is reverted. Moreover, if the scores provided are out of range, the transaction fails as well. Clients provide ratings for different properties of the fog node. In this simulation, clients evaluate the response time of the fog node, its availability, the storage of fog node, and the cost of its services. Several conclusions can be drawn from testing the reputation contract:

- A large group of clients that continuously interact with and evaluate the fog node suppress the bad evaluation of few members even if they have high credibility.
- If a fog node wants to maintain a high reputation score, it has to provide consistently good service as good fog nodes attract many credible clients. Several reputable clients can bring down the reputation score of a fog node quickly if they notice a deterioration in its performance. This gives additional incentive for the fog nodes to provide the best possible services at low cost to attract and keep more clients.
- The first few evaluations resulted in extreme fluctuations in the reputation score of the fog node. With more evaluators, the reputation score tends to converge to a small

TABLE 1. Gas cost of ethereum functions in USD.

Function name	Transaction gas	Execution gas	Cost (USD)
registerFog	26111	4839	0.001
registerClient	23513	2241	0.001
score	233173	209917	0.048
manageDeposit	37825	54377	0.012
calculateCredibility	33221	10541	0.002

TABLE 2. Execution cost of reputation functions in USD.

Function name	Execution gas for different numbers of scores submitted for a fog node (USD)			
	10 scores	20 scores	30 scores	40 scores
calculateReputation	0.011	0.014	0.021	0.027
majorRep	0.101	0.128	0.120	0.166
customRep	0.019	0.028	0.037	0.09

range of values with the exception of unusual change in the services or intrinsic properties of the fog node.

3) CREDIBILITY CONTRACT

Computing the credibility of raters depends on the scores provided in contrast with other raters, the consistency of the client itself, and its contribution to the reputation system. On top of that, an adjusting factor is used. The adjusting factor measures the forgiveness of the system. Increasing this factor means that raters gain trust slower whereas decreasing this factor results in rewarding good ratings generously and forgiving occasional inaccurate ratings. In this implementation, the factor is set to 16.

B. COST AND PERFORMANCE ANALYSIS

When a transaction is executed on the Ethereum blockchain, it costs Ethers paid as gas. Table 1 shows the transaction and execution gas spent when calling some of the functions. These functions spend the same gas each time they are executed because they do not depend on the number of clients, fog nodes or the number of evaluations. Execution gas represents the virtual machine cost which is the cost of executing actual code using Ethereum virtual machine (EVM). The transaction cost includes the execution in addition to sending the contract to the Ethereum blockchain [26]. Moreover, the algorithms of computing the reputation of fog nodes and credibility of clients spend different amounts of gas depending on number of clients, number of evaluations, how close those evaluations are and other factors. Generally, with more clients using this fog node and providing feedback about it, the smart contract takes more time in calculating its reputation as well as the credibility scores of involved clients. We simulated multiple scenarios of possible feedback data and presented the execution gas cost in table 2 with different number of evaluations. When there are 10 scores, *majorRep* spends the most at \$0.101. The gas cost for all methods increase as more scores are submitted for a fog node as shown in the table. Table 1 and Table 2 show the gas costs of the functions as of August 4, 2019 enlisted on ETH Gas Station [27]. All functions spend less than \$0.05 when executed. The function that calculates the majority reputation spends the most amount of gas (\$0.166 at 40 score submissions) because it involves a clustering algorithm that involves more computations than other functions. This cost is further reduced if clients are more honest with their scores. When scores are close to each other, the majority reputation is easier to compute and calculate the credibility of the client.

The solution discussed in this paper uses the Ethereum blockchain to deploy the reputation system. On the blockchain, all clients and fog nodes have unique EAs. Before being able to interact with each other, the devices have to register with their corresponding EAs. All score submissions from clients are linked to their corresponding EAs. This results in non-repudiation where no client can deny the evaluation given to a fog node. All client devices are held accountable for their feedback and which is reflected on their credibility score. Moreover, the clients cannot directly change the reputation value given to a fog node nor can they change their own credibility score. Clients can provide their feedback to the smart contract which decides the weight it will give to this score and how much it can affect the final reputation score of the fog node. Therefore, the integrity of the data is maintained as there is no way for users to directly alter it. In addition, some responsibilities are only given to certain authorities. For example, only the contract owner is allowed to register new fog nodes and no client can provide a score submission on behalf of another client node. This ensures every function is executed by the appropriate entity.

C. SECURITY ANALYSIS

Using the blockchain technology results in inheriting its threats and security challenges. Blockchain is decentralized and so it mitigates the risk of DDoS attacks. The data on the blockchain is immutable and are kept unchanged forever. This data could be compromised if an attacker or more possibly a group of attackers were able to take over 51% of the network, this is known as the 51% attack or the majority attack [28]. However, considering the PoW based consensus mechanism it is quite hard to attack the Ethereum network.

The proposed smart contracts were analyzed using multiple security analysis tools including Oyente, MythX, Securify, and SmartCheck. First, the smart contracts were evaluated using Oyente smart contract analyzer and the results are reported in the Table 3. Oyente analyzes EVM byte codes and generates corresponding call graph of each smart contract. It performs the block-level analysis of smart contract code following the code analysis guidelines provided in the Ethereum's yellow paper [29]. Each smart contract was analyzed separately and their EVM code coverage is presented in the Table 3. The results show that our smart contracts are safe against known security vulnerabilities. There was no unhandled exception which could lead towards underflow or

TABLE 3. Security analysis of smart contract.

Parameters	Client Registration Contract	FogNode Management Contract	Reputation Contract	Credibility Contract	Custom Reputation Contract
EVM Code Coverage	68%	31%	76%	54%	49%
Integer Underflow	False	False	False	False	False
Integer Overlfow	False	False	False	False	False
Parity Multisig Bug 2	False	False	False	False	False
Callstack Depth Attack Vulnerability	False	False	False	False	False
Transaction Ordering Dependence	False	False	False	False	False
Time Stamp Dependency	False	False	False	False	False
Re-Entrancy Vulnerability	False	False	False	False	False

overflow of integer operations in both caller and callee smart contracts. In addition, the external calls were minimized and all checks were performed to ensure gas availability while executing callee smart contracts in order to prevent the re-entracy attacks. Similarly, it was ensured to minimize the number of external calls and their resultant operations in order to avoid the call stack attacks. The results also show that there was no vulnerability related to the Ethereum platform which may result in the form of parity multisig bug 2, transaction ordering dependency, or time-stamp dependency. Moreover, MythX security verification tool was also used to analyze the smart contracts. MythX is a built-in plugin in Remix that can be activated and used to examine Ethereum Smart contracts and highlight the potential security gaps and bad practices in the smart contract code. Finally, Sercurify analyzer [30] and SmartCheck were used to check for potential vulnerabilities. All three of those analyzers reported no major security issue in our smart contracts. Nevertheless, these tools brought few non-critical issues to our attention which were mitigated to ensure confidence in the smart contracts.

VII. CONCLUSION

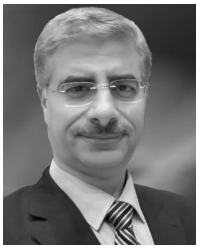
In this paper, we presented a decentralized reputation system for the fog nodes based on the blockchain. We have used Ethereum smart contracts to implement the system logic. The implemented solution is general enough to accommodate for any changes in the evaluated metrics and it can be applied to multiple domains. The solution has been optimized to ensure minimum cost, and it was tested on the Remix IDE using solidity. The entity relation diagram, sequence diagram and algorithms were provided if needed for manipulation to apply the solution in other applications. The Ethereum blockchain supports decentralization and ensures our solution is validated, immutable and secure. The solution explains how the reputation scores are adjusted after each interaction. The credibility of raters is also accounted for to guarantee honest feedback from the IoT devices. Multiple scenarios were tested to see whether the reputation scores of fog nodes and credibility of clients are adjusted in contrast to their respective performance. We approximated the cost of different operations and analyzed their performance.

REFERENCES

[1] I. Stojmenovic and S. Wen, "The fog computing paradigm: Scenarios and security issues," in *Proc. Federated Conf. Comput. Sci. Inf. Syst.*, vol. 2, Sep. 2014, pp. 1–8.
[2] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog computing: A taxonomy, survey and future directions," in *Internet of Everything*. Singapore: Springer, 2017, pp. 103–130.
[3] M. H. Rehman, L. C. Sun, T. Y. Wah, A. Iqbal, and P. P. Jayaraman, "Opportunistic computation offloading in mobile edge cloud computing environments," in *Proc. 17th IEEE Int. Conf. Mobile Data Manage. (MDM)*, vol. 1, Jun. 2016, pp. 208–213.
[4] I. Yaqoob, E. Ahmed, M. H. Rehman, A. I. A. Ahmed, M. A. Al-garadi, M. Imran, and M. Guizani, "The rise of ransomware and emerging security challenges in the Internet of Things," *Comput. Netw.*, vol. 129, pp. 444–458, 2017.
[5] M. A. Khan and K. Salah, "IoT security: Review, blockchain solutions, and open challenges," *Future Gener. Comput. Syst.*, vol. 82, pp. 395–411, May 2018.
[6] Y. Hussain and Z. Huang, "TRFIoT: Trust and reputation model for fog-based IoT," in *Proc. Int. Conf. Cloud Comput. Secur.*, Haikou, China, Springer, 2018, pp. 187–198.
[7] M. H. Rehman, P. P. Jayaraman, and C. Perera, "The emergence of edge-centric distributed IoT analytics platforms," in *Internet Things*. Orange, CA, USA: Chapman, 2017, pp. 213–228.
[8] Y. Wang and J. Vassileva, "Trust and reputation model in peer-to-peer networks," in *Proc. 3rd Int. Conf. Peer-to-Peer Comput.*, 2003, pp. 150–157.
[9] F. G. Mármol and G. M. Pérez, "Security threats scenarios in trust and reputation models for distributed systems," *Comput. Secur.*, vol. 28, no. 7, pp. 545–556, Oct. 2009.
[10] O. Hasan, L. Brunie, and E. Bertino, "Preserving privacy of feedback providers in decentralized reputation systems," *Comput. Secur.*, vol. 31, no. 7, pp. 816–826, 2012.
[11] A. Panarello, N. Tapas, G. Merlino, F. Longo, and A. Puliafito, "Blockchain and IoT integration: A systematic survey," *Sensors*, vol. 18, no. 8, p. 2575, 2018.
[12] M. Debe. K. Salah, M. H. Rehman and D. Svetinovic, "Blockchain-based decentralized reputation system for public fog nodes," in *Proc. ACS/IEEE Int. Conf. Comput. Syst. Appl. (AICCSA)*, Abu Dhabi, UAE, Nov. 2019, pp. 1–6.
[13] L. Horwitz. *Connected Devices Push Business to the Edge (Edge Computing Architecture, That is)*. Accessed: Aug. 10, 2019. [Online]. Available: https://www.cisco.com/c/en/us/solutions/internet-of-things/edge-computing-architecture.html
[14] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.
[15] S. Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. [Online]. Available: https://bitcoin.org/bitcoin.pdf
[16] K. Salah, M. Rehman, N. Nizamuddin, and A. Al-Fuqaha, "Blockchain for AI: Review and open research challenges," *IEEE Access*, vol. 7, pp. 10127–10149, 2019, doi: 10.1109/ACCESS.2018.2890507.
[17] R. Almadhoun, M. Kadadha, M. Alhemeiri, M. Alshehhi, and K. Salah, "A user authentication scheme of IoT devices using blockchain-enabled fog nodes," in *Proc. IEEE/ACS 15th Int. Conf. Comput. Syst. Appl. (AICCSA)*, Aqaba, Jordan, Oct./Nov. 2018, pp. 1–8.
[18] H. Hasan and K. Salah, "Combating deepfake videos using blockchain and smart contracts," *IEEE Access*, vol. 7, no. 1, pp. 41596–41606, Dec. 2019.

[19] M. Iansiti and K. R. Lakhani, "The truth about blockchain," in *Harvard Business Review*. Boston, MA, USA: 2017.
[20] M. Swan, *Blockchain: Blueprint for a New Economy*, Sebastopol, CA, USA: O'Reilly Media, 2015.
[21] J. Cho, K. Kwon, and Y. Park, "Q-rater: A collaborative reputation system based on source credibility theory," *Expert Syst. Appl.*, vol. 36, no. 2, pp. 3751–3760, 2009.
[22] Z. Malik and A. Bouguettaya, "RATEWeb: Reputation assessment for trust establishment among Web services," *VLDB J.*, vol. 18, no. 4, pp. 885–991, 2009.
[23] R. Dennis and G. Owen, "Rep on the block : A next generation reputation system based on the blockchain," in *Proc. 10th Int. Conf. Internet Technol. Secured Trans.*, London, U.K., 2015.
[24] N. B. Truong, T.-W. Um, B. Zhou, and G. M. Lee, "Strengthening the blockchain-based Internet of value with trust," in *Proc. IEEE Int. Conf. Commun.*, Kansas City, MO, USA, May 2018, pp. 1–7.
[25] L.-H. Vu, M. Hauswirth, and K. Aberer, "QoS-Based service selection and ranking with trust and reputation management," in *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE*, vol. 3760. Berlin, Germany: Springer, 2005, pp. 466–483.
[26] B. Skvorc. (Dec. 5, 2017). *Ethereum Gas and Transaction Fees Explained!* [Online]. Available: https://www.sitepoint.com/ethereum-transaction-costs/
[27] *Eth Gas Station*. Accessed: Jul. 27, 2019. [Online]. Available: https://ethgasstation.info/
[28] I. C. Lin and T. C. Liao, "A survey of blockchain security issues and challenges," *Int. J. Netw. Secur.*, vol. 19, no. 5, pp. 653–659, 2017.
[29] G. Wood. (2014). *Ethereum Yellow Paper*. Accessed: Oct. 30, 2019. [Online]. Available: https://github.com/ethereum/yellowpaper
[30] P. Tsankov, D. Andrei, D.-C. Dana, G. Arthur, B. Florian, and V. Martin, "Securify: Practical security analysis of smart contracts," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2018, pp. 67–82.

MAZIN DEBE is currently a Graduate Student in computer science with the Khalifa University of Science and Technology, Abu Dhabi, UAE. His research interests include blockchain technology, the Internet of Things (IoT), and fog computing.



KHALED SALAH received the B.S. degree in computer engineering with a minor in computer science from Iowa State University, USA, in 1990, the M.S. degree in computer systems engineering and the Ph.D. degree in computer science from the Illinois Institute of Technology, USA, in 1994 and 2000, respectively. He is currently a Full Professor with the Department of Electrical and Computer Engineering, Khalifa University, UAE. He joined Khalifa University, in August 2010, where he is teaching graduate and undergraduate courses in the areas of cloud computing, computer and network security, computer networks, operating systems, and performance modeling and analysis. Prior to joining Khalifa University, he worked at the Department of Information and Computer Science, King Fahd University of Petroleum and Minerals (KFUPM), Saudi Arabia, for ten years. He has over 190 publications and holds three patents, has been giving a number of international keynote speeches, invited talks, tutorials, and research seminars on the subjects of blockchain, the IoT, fog and cloud computing, and cybersecurity. He is a member of the IEEE Blockchain Education Committee. He was a recipient of the Khalifa University Outstanding Research Award, in 2014 and 2015, the KFUPM University Excellence in Research Award, in 2008 and 2009, and the KFUPM Best Research Project Award, in 2009 and 2010, and the departmental awards for Distinguished Research and Teaching in prior years. He serves on the editorial boards of many WoS-listed journals, including *IET Communications*, *IET Networks*, Elsevier's JNCA, Wiley's SCN, Wiley's IJNM, J.UCS, and AJSE. He is the Track Chair of the IEEE Globecom 2018 on Cloud Computing. He is an Associate Editor of the IEEE Blockchain Newsletter.

MUHAMMAD HABIB UR REHMAN received the Ph.D. degree in mobile distributed analytics systems from the Faculty of Computer Science and Information Technology, University of Malaya, Malaysia. He is currently a Postdoctoral Research Fellow with the Center for Cyber Physical Systems, Khalifa University, UAE. His research interests include blockchain, edge computing, big data, and the Internet of Things.

DAVOR SVETINOVIC received the Ph.D. degree in computer science from the University of Waterloo, Canada, 2006. He is currently an Associate Professor of computer science with the Masdar Institute, Khalifa University of Science and Technology, UAE, and also a Research Affiliate with the Massachusetts Institute of Technology (MIT), USA. Previously, he worked as a Visiting Professor at the Massachusetts Institute of Technology (MIT) and a Postdoctoral Researcher at the Lero–the Irish Software Engineering Center, Ireland, and the Vienna University of Technology, Austria. He leads the Strategic Requirements and Systems Security Group (SRSSG), and he has extensive experience working on complex multidisciplinary research projects. He has published over 75 articles in leading journals and conferences. His current research interests include systems security and privacy, blockchain engineering, cryptocurrencies, and requirements engineering. He is a Senior Member of ACM.