Monetization of Services Provided by Public Fog Nodes Using Blockchain and Smart Contracts

MAZIN DEBE[ID]**, KHALED SALAH**[ID]**, MUHAMMAD HABIB UR REHMAN**[ID]**,
AND DAVOR SVETINOVIC**[ID]

Center for Cyber-Physical Systems, Electrical Engineering and Computer Science, Khalifa University, Abu Dhabi, United Arab Emirates

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

ABSTRACT Public fog nodes can be deployed in public places closer to the edge where many personal and commercial devices (e.g., a sensor, an application, or a device) can connect to. These public fog nodes can provide real-time and localized services for networking, computing, storage and content delivery to the connected devices. The monetization and payment of such services is typically manual, centralized, and lacks the necessary trust. The providers of the public fog nodes typically offer fixed pricing models for their services, and the customers manually select and pay for the used services, with little or no transparency and trust in the provided service in terms of the used time, network bandwidth, and quality of service (QoS). This paper presents a novel scheme to enable blockchain-based monetization and automated payment in cryptocurrency for services provided by public fog nodes. The proposed scheme is decentralized, trustworthy, automated, and with certain guarantees for QoS, customer satisfaction, and dispute resolutions through a reputation system. The proposed solution uses the Ethereum blockchain and its native smart contract features to govern the interactions between devices and fog nodes. The proposed solution is implemented, tested and evaluated to show correct behavior and functionality. We also provide cost and security analysis and show that our solution is resilient against major security attacks. Our smart contract is made publicly available on Github[1].

INDEX TERMS Blockchain, public fog nodes, fog computing, pricing models, monetization, automated payments.

I. INTRODUCTION

Fog computing extends the cloud computing services from core network infrastructures to customer-premises using a variety of fog nodes namely smart switches, micro data centers, cloud-lets, and proximal mobile edge servers, to name a few. The proximity of fog nodes with devices enables a large plethora of latency-minimal, energy-efficient, and bandwidth-optimal applications for smart cities, healthcare, agriculture, sports, etc [1]. In addition, fog nodes reduce the data redundancy and enrich fog applications with local and near-real-time intelligence. Considering the potential value, which is predicted as 700 million US dollars by the year 2024, all major cloud service providers are actively developing their fog-cloud services platforms to fulfill the computational, communication, and data management needs

of emerging data-intensive and compute-intensive applications [2]–[5]. Alternately, all the major Telecommunication service providers and content delivery networks are realizing their business models on top of another variant of fog computing i.e. multi-access edge computing [6], [7].

Fog node providers normally offer different quality of services (QoS) and tailor their subscription-based or consumption-based pricing models to maximize the revenue from their customers [8]. These monetization strategies work well in the case of centralized cloud services whereby customers are guaranteed to meet with their QoS. However, the mobility and remotely managed fog infrastructures cause service degradation and result in mistrust between fog node providers and their customers which leads to customer-churn and vendor lock-in situations where customers are forced to either unsubscribe the service providers or stop their subscriptions. For example, a content delivery network promises their users to provide real-time and personalized insights during a soccer game in the football arena and the company
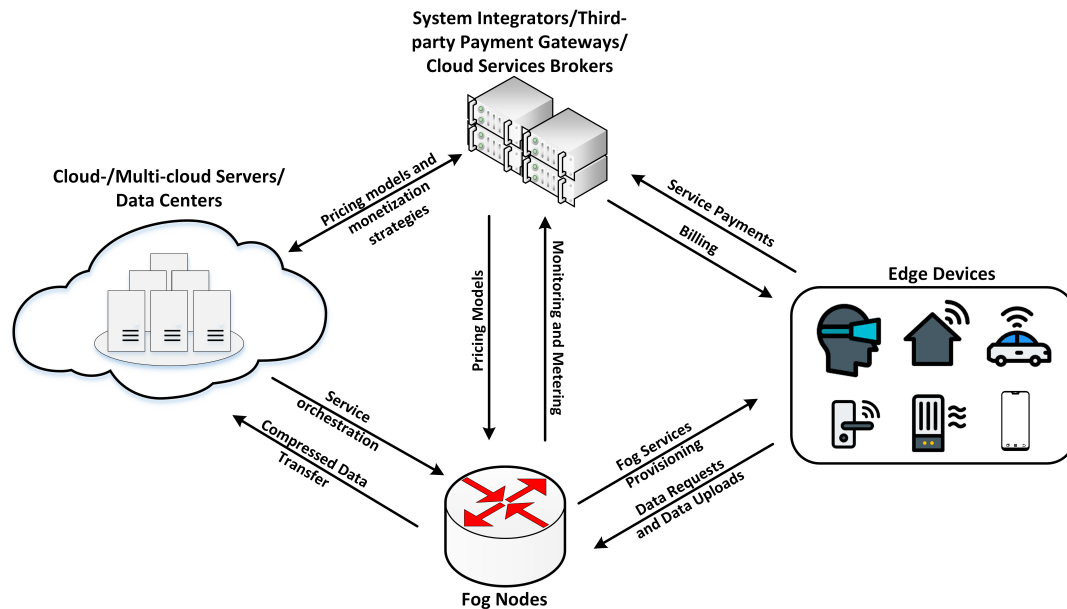
FIGURE 1. Centralized monetization of fog node services and the interactions among key stakeholders.

uses mobile fog nodes (such as drones and unmanned aerial vehicles) to capture and analyze the imaginary and push the resultant analytics to devices in their proximity. However. the service quality degrades due to personalization requests from the users. Similarly, in the case of smart cities, the service quality of stationary fog nodes may degrade on highways, remote areas, densely populated areas, or during bad weather conditions. These adverse situations create mistrust between fog node providers and their customers whereby fog node providers charge their customers in advance but they do not deliver the promised QoS.

Despite various pricing models and payment plans, fog node providers struggle to ensure a trustworthy payment system due to the involvement of cloud service brokers and third-party system integrators who add their own commission fees and operational cost before delivering the end products to the customers [9]. One obvious solution is to utilize a third-party registration and payment system to manage the registration of all stakeholders (i.e. devices, cloud-service broker, system integrators, and cloud vendors), to customize the service level agreements, to outline the QoS metrics, to tailor pricing models, and to enable secure and trustworthy payment system. However, deploying such a centralized solution, as depicted in Fig. 1, defies the purpose of fog computing as it adds an overhead to the communication between all the stakeholders. Moreover, such a solution brings centralized point-of-monopoly and point-of-compromise whereby some stakeholders can collude and unfairly jeopardize the whole ecosystem. Hence, there is a need for a decentralized solution that automatically manages the interaction between fog nodes and devices, handles the payment and resolves mistrust issues. Using Blockchain and smart contract technologies

the payments could be automated, the service usage could be recorded, and the unspent amounts could be refunded. In addition, smart contracts could facilitate in resolving the trust issues and fairly calculate the reputation of all stakeholders in cloud ecosystems. Furthermore, Blockchain and smart contract technologies eradicate several problems regarding security and privacy, and transaction fees to third-party payment systems; all transactions on the blockchain are decentralized, secure, and immutable.

This paper presents a method to automate the payments against the used fog node services offered by public fog nodes and accessing them by leveraging smart contracts that are deployed on the blockchain. We propose a design for such a scheme using the Ethereum blockchain along with the interactions between public fog nodes and their connected devices. An implementation of the automated payment process is also explained to show the interactions between fog nodes and their connected devices through the smart contracts. In brief, the main contributions of this paper are:

- We present a blockchain cryptocurrency-based monetization and automated payment solution for public fog nodes which ensures decentralization and increases the level of trust among public fog nodes and their connected devices.
- We used the blockchain and smart contract technologies to automate the dispute-free payment for services provided by public fog nodes. The proposed smart contract guarantees QoS and customer satisfaction.
- We implemented, tested, and thoroughly analyzed our proposed scheme to ensure correct functionality and resilience against major security attacks.

The remainder of the paper is organized as follows: Section II briefly explains the concepts of fog computing, Internet of Things (IoT), and blockchain. Section III reviews some of the work done in the fields of service monetization. Section IV introduces the proposed design for monetizing the fog node services. Section V showcases the implementation details and section VI provides the testing results of the smart contract, and the costs and performance analyses. The paper is concluded in Section VII.

II. BACKGROUND

This section presents a brief overview of blockchain, smart contracts, IoT, and fog computing.

The blockchain technology is a peer-to-peer network that comprises a chain of blocks where this chain is shared between all of its users everywhere [10]. Each block consists of multiple transactions that are added based on a consensus protocol. Once the information is added to this distributed immutable ledger and broadcasted among all its users, it becomes very hard to remove or modify the distributed ledger. The blockchain has been gaining more interest from researchers and developers ever since the first cryptocurrency surfaced in 2009 by Satoshi Nakamoto [11]. It is a horizontal technology and can be implemented in various fields especially due to the development of smart contracts. The design of the blockchain that is concentrated around privacy and security as well as its decentralized nature has led to deploying it in many fields including artificial intelligence, IoT systems, supply-chain management and logistics, and health-care as explained in [12]–[15]. Smart contracts represent the logic layer of the blockchain where users can receive and transfer money and digital assets based on some pre-defined conditions. Ethereum is one of the programmable blockchain networks that support the execution of smart contracts. Developers can program their own smart contracts to run on the Ethereum Virtual Machine (EVM), which is the run-time environment for Ethereum. To develop a smart contract for Ethereum, a special language called Solidity is often used. Moreover, decentralized applications (DApps) are used to encapsulate multiple smart contracts and enable devices to read the information on blockchain and execute different rules to govern the transactions on the blockchain. The transactions in a blockchain are added according to a consensus protocol such as proof-of-work (PoW) in the case of Ethereum blockchain. In PoW, special nodes called miners compete with each other by solving a problem in order to add a group of transactions to the chain in the form of a block. According to the blockchain governance model, the miner that succeeds in adding a block gets a reward in the form of Ethereum's native cryptocurrency, called Ether.

IoT is the network of sensing devices who can collect and process data and they are accessible via an internet connection [16]. These devices have become very ubiquitous as they can be used in various fields. IoT devices are light-weight computing devices that are designed to execute a small set of tasks. Typically, IoT devices have sensors and actuators that detect some type of data such as temperature or light intensity and communicate this data to a cloud server. IoT devices connect with a server because they are very limited on processing power so they constantly record and report the data. While this data is mostly small in size, the continuous transfer to the cloud creates a substantial amount of data to be processed. To aid with that, a layer of helping devices sit between these IoT devices and the cloud, which is known as fog computing. Fog computing is an extension of the cloud computing platform to better serve IoT devices [1]. The paradigm of fog exists because in almost every application of IoT, some of the biggest challenges that face the cloud-IoT interconnection are the security, reliability, performance, and the heterogeneity of the devices [17]. Physically, fog nodes can be routers, switches, or any server under the cloud layer. They are responsible for the devices in their geographical area and provide them with services. Fog nodes are positioned close to the IoT devices and they handle the heterogeneity of the data coming from different devices.

III. RELATED WORK

This section showcases the previous work done on monetization models of IoT data. Although the available literature does not provide work on the monetization of services for fog nodes, we have tried to compile some of the closest research to this topic. The presented literature covers the monetization of IoT data and blockchain implementations.

In [19], a smart data pricing model has been developed for IoT applications. The model provides a way to set the subscription prices for data generated by sensors. It also offers a mechanism for service providers to form groups and market their services collectively. This approach helps in providing a more attractive offer for users and is expected to increase the user-base. Researchers in [20] presented a blockchain-based approach to ensure trust in the monetization and trading of IoT devices based on an automated system that reviews IoT data in order to monetize it. The Ethereum blockchain was used to implement their solution via smart contracts. In [21], four models of monetization in the cloud are presented. The explained models consist of the monetization, charging, billing, and taxation models for all of the services provided by cloud servers. The proposed models enable the cloud providers to scale their services according to the demand of their devices. Cloud servers do not have to manually maintain the scalability of services that they provide. The concepts provided in [22] explain the monetization models from the perspective of the devices as well as the cloud servers. The explained business model works on maximizing the revenue and reducing the costs.

Amazon owns one of the world's most endorsed cloud computing platforms, Amazon Web Services (AWS) offers a large number of services relied upon by millions of users. AWS provides the pricing model for each service [23]. For most services, amazon charges the users per usage where amazon itself calculates the bill for each customer upon usage. In AWS, customers pay for the services they have

used and for as long as they have been using them only with no commitment. AWS offers services for computation, storage, machine learning, data analytics, mobile and gaming services, and IoT services. Depending on the service, AWS' pricing model can be categorized into four main categories:

- **Time based:** Users are charged per second or per minute of usage. A minimum charge is often introduced in this case, where customers are requested to pay for at least 60 seconds of usage for example at minimum.
- **Number of devices:** This model is typically used with connected IoT devices. IoT devices generally exist in bulks and need to communicate their data to a cloud server.
- **Data transfer:** This model charges users per KB/MB/GB of data transferred. Data is mostly split into blocks. This means that if the block size is 10 KB, processing 4KB of data costs as much as processing 10 KB of data.
- **Subscription based:** Users are required to pay for a monthly or yearly subscription to benefit from this kind of service.

AWS IoT Core provides services for connecting IoT devices, messaging, storage, and routing. These services are very cost-efficient as the connectivity price, for instance, at the time of this writing is $0.08 per million minutes of connection which amounts to only $0.042 per device per year in East US region. Other services are priced per data transfer as opposed to the per-device policy. IoT device management service for example charges $2.25 per 1KB of index update and $0.05 for each search query. Other services that charge per data usage include IoT Analytics, Events, and IoT Site-Wise Pricing. IoT Device Defender requires a monthly subscription while FreeRTOS service is under an open-source license. However, one of the most popular amazon services is the EC2 on-demand virtual server instances on the cloud. With EC2, users can customize their own Windows or Linux instances and pay for the time of usage only. The size of instances varies greatly to account for the needs of all customers. The price of an hour of usage for a Linux instance, for example, can range from $0.0047 to $14.40. Another set of services that Amazon charges for on-time basis are the data analytics services including the Amazon Elastic MapReduce (EMR) and the CloudSearch which are useful for big data.

The authors have worked previously on a reputation system to establish trust in public fog nodes. Such a reputation system presents an ideal extension of the functionality of the monetization system proposed in this paper. The proposed reputation solution in [24] was decentralized and was implemented on the Ethereum blockchain using smart contracts. In this solution, the fog nodes provide services to the devices and the devices provide their feedback about these services. Then, the smart contract evaluates the reputation score for the fog node. Moreover, the honesty of the device is also taken into consideration to count for malicious IoT devices. The monetization smart contract can optionally provide some precious information for the smart contracts responsible for the

reputation and credibility. Some of this information includes the fog pricing models termed as fog rates in this research, devices' commitment to payment delivery, and conflict between fog nodes and devices. These metrics influence the fog node reputation and device credibility score computation as per the corresponding smart contracts.

IV. PROPOSED APPROACH

In this section, we will present a novel monetization and automated payment scheme for the public fog nodes. The proposed scheme uses blockchain and smart contract technologies to automate payments and it is integrated with our previous work on blockchain-based reputation system for fog nodes. The Ethereum smart contracts are used to implement the logic needed by the smart contract owner which enables fog node providers to monitor and meter their services and devices to automatically pay their bills.

The Ethereum smart contracts support the functionality of payments and deposits as they can receive and transfer money, Etherum blockchain's native currency i.e. Ether, to and from Externally Owned Accounts (EOA) as well as other smart contract accounts. Fig. 2 shows the proposed system that consists of the IoT devices, the public fog nodes that provide services to those devices as well as the monetization smart contract that manages the interaction between them. The owners of smart contracts, normally the fog node providers, register the fog nodes, configure the QoS offerings, and tailors the pricing models accordingly. However, devices can be explicitly registered by device owners or application users. This is because the fog node providers need to control the number of fog nodes in a particular location. The device transfers a deposit to the smart contract to be able to access the fog services. After discovering available fog nodes, a device chooses one of the fog nodes to communicate with. While fog nodes can service multiple devices, however, these devices can only connect to one of the fog service providers. The device has to subscribe to a fog node before being able to connect to it. This workflow is entirely managed by the smart contract. The smart contract receives deposits, manages subscriptions and connections, and maintains the connection between two entities whenever needed. Moreover, it manages subscription fees, receives payments from devices, transfers them to the fog nodes, and resolves the conflicts between fog nodes and devices. In the smart contract, each entity has its privileges and no entity can access data or methods that are restricted to other entities. This level of authorization has three layers: fog node provider-only, fog-only and device-only. As the names suggest, only the fog node providers can access methods that have the fog node provider level restrictions and the same goes for the devices and the fog nodes. However, the owner has full access to all methods and data as it is considered a trusted operator.

The proposed scheme involves multiple participants (i.e. fog nodes, devices, and monetization smart contract) whereby fog nodes and devices access the monetization smart contract through their unique Ethereum addresses.
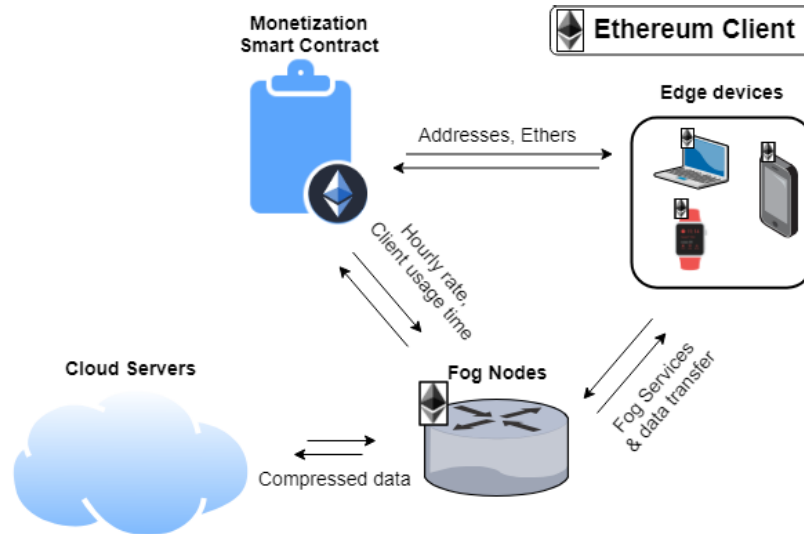
FIGURE 2. Proposed Decentralized Monetization and Automated Payment Scheme.

Any Ethereum device or smart contract requires an address on the blockchain network. Fog nodes and devices, therefore, have their own Ethereum addresses that the smart contract uses to keep track of their metadata. The participants of the system are explained in further detail below:

- **Public fog nodes** are devices in the layer between devices and their cloud servers such as routers and switches. These devices offer services to improve performance and reduce latency and can serve multiple devices at the same time. However, the number of connected devices need to be monitored as they will be sharing the bandwidth. While more devices will generate more revenue for the fog node, increasing the number of devices will result in higher latency and reduced performance. The bad performance will encourage devices to provide negative feedback about the fog node which is undesirable. Therefore, the fog node needs to adjust the number of devices with its capabilities as well as its rate. The fog node sends its rate per hour to the smart contract with a deposit as a guarantee of service delivery. Moreover, the fog nodes can adjust their rate based on the device's feedback.
- **Customer devices** are mainly IoT devices but it include all other applications and devices, either mobile or stationary, which could be connected to fog nodes and use their services. These devices presumably have multiple available fog nodes in their vicinity who are willing to offer them their services. The devices choose the fog node depending on their QoS offerings and pricing models. To access any fog node, devices need to deposit money equal to the minimum required fee by the fog node. However, the final fee that the device owes the fog node is determined by the fog node by the end of the connection. If the device has an amount overdue, the smart contract flags the user and blocks the

device from accessing the fog nodes. When deploying in conjunction with a reputation system, devices can also choose a fog node depending on its reputation value. In addition to that, failing to pay for the services affects the device's credibility score. IoT devices can only be connected to one fog node at once. So starting a connection with a fog node requires terminating the previous connection.
- **Monetization smart contract** governs the connection between the fog nodes and their connected devices. It manages the device subscriptions as well as the connected devices to each fog node in an efficient, organized manner. The devices transfer the money to the smart account's Ethereum address to be able to access the services of fog nodes. Not all devices can access the full features of the smart contract. Other than the owner, the smart contract restricts the devices from accessing unauthorized information. To add a level of trust in the smart contract, it issues a random key upon each connection between a device and a fog node and shares this key with the fog node. The smart contract also triggers events to mark the successful connection requests to notify both parties.

Fig. 3 depicts the relationship among different participants in this study. Each fog node has a mapping or a list of all subscribed devices that can be connected to this fog node. A device can subscribe to a fog node if it is in the same geographical location and if it has paid the subscription fee (if required by the fog node). The devices are connected to only one fog node at a particular instance of time. The devices are given an access key to authenticate with the fog node for security. The smart contract registers and manages both entities. It handles all required processes like subscribing to fog nodes, connection, terminating connections, and refunding devices.
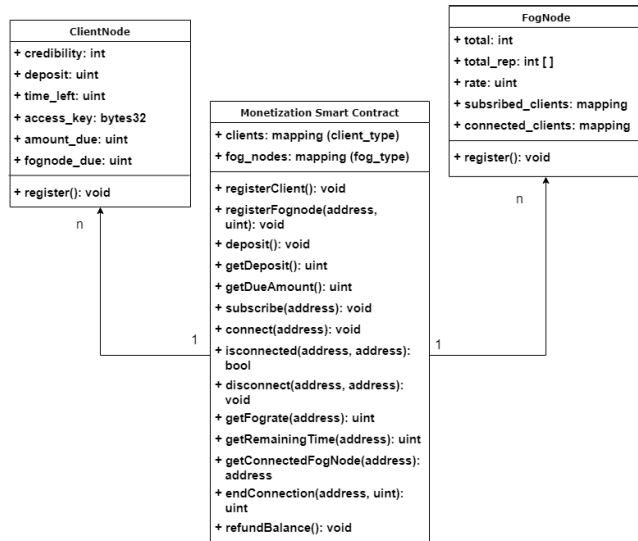
FIGURE 3. Relationship among different entities of the smart contract.

Fig. 4 shows the interactions between fog nodes and devices with the monetization smart contract in an arranged time sequence of a typical scenario. The interaction requires both fog nodes and devices to be registered in the system. Fog nodes are registered by the smart contract owner but devices can register themselves in the system by triggering the required method or by simply transferring a deposit to the address of the smart contract. When a device decides to access a fog node, it has to subscribe to it first. This way the smart contract can validate the device details such as any outstanding payments, connected fog service providers, and available balance. When the connection is initiated, the smart contract computes the estimated time left for the user to access the fog node based on its available balance. The user is assigned a hashed access key so that the fog node can verify that device using that key. When the device requests to end the session or if the balance runs out, the fog node terminates the connection. The fog node provides the time of usage to the smart contract so that the bill is computed and the connection ends.

Fig. 5 shows the interactions resulting from misbehavior or disagreement between fog nodes and devices. After the connection has ended, if the amount deposited by the device is insufficient, the device is blocked. Optionally, the credibility of the device can be modified to prompt cooperation by the devices. After the smart contract verifies the amount requested by the fog node, it offers a way for the device to object to the amount due. If the device claims that the amount is wrong, the payment is frozen and the trust is established by using the reputation score. As described in [24], this method assumes that the majority of devices are honest as it proposes a method to avoid collusion.

V. IMPLEMENTATION

The proposed solution was implemented using Solidity language on Remix IDE. The Remix is a development environment that offers a virtual Ethereum platform for
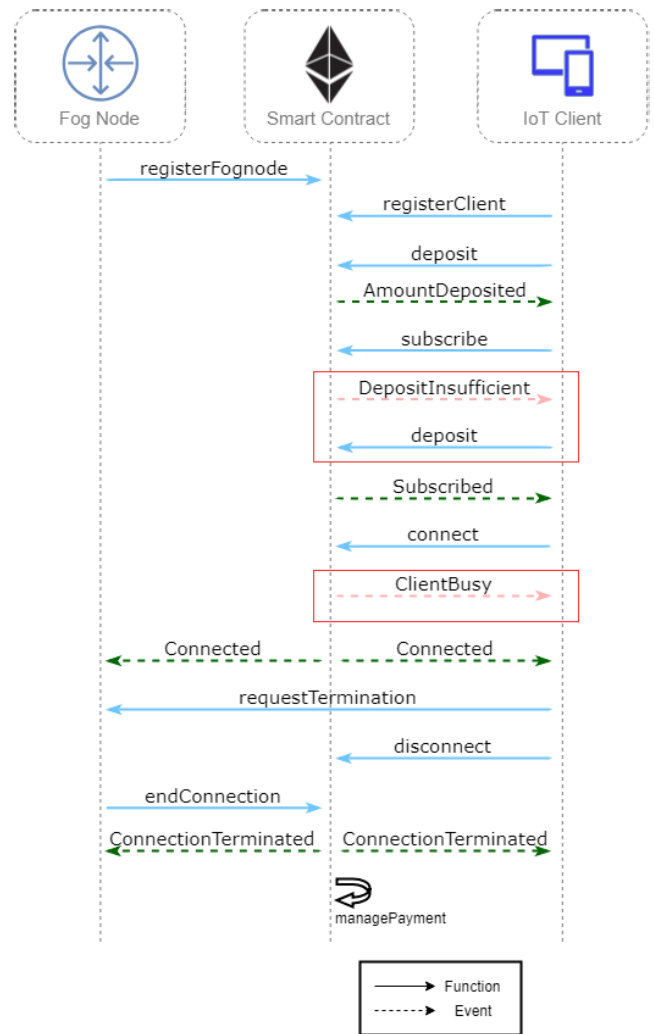

FIGURE 4. Interactions showing the function calls for service monetization.

deploying and testing smart contracts in addition to multiple plugins for debugging and analysis of the code. In the implementation of the smart contract, the fog nodes and devices are presented as mappings from the Ethereum addresses of the nodes to objects that hold the information about that entity. The Ethereum address of the contract owner is initialized at the time of smart contract deployment. The owner represents the highest authority and has access to all the system's functionality.

After the fog nodes and devices have registered in the smart contract, the fog node first transfers a deposit as a guarantee for the QoS. Then, the device deposits an amount of money into its account to access the fog nodes. As shown in Fig. 6, this method is only accessible by devices. If the device does not have any amount due, the deposit is added to the account. However, this balance can be used to cover previous expenses. This means that the device was blocked until it pays off its debt. If the deposit was enough to clear the amount due, the device is unblocked and it can access the fog node services, otherwise, it remains blocked.
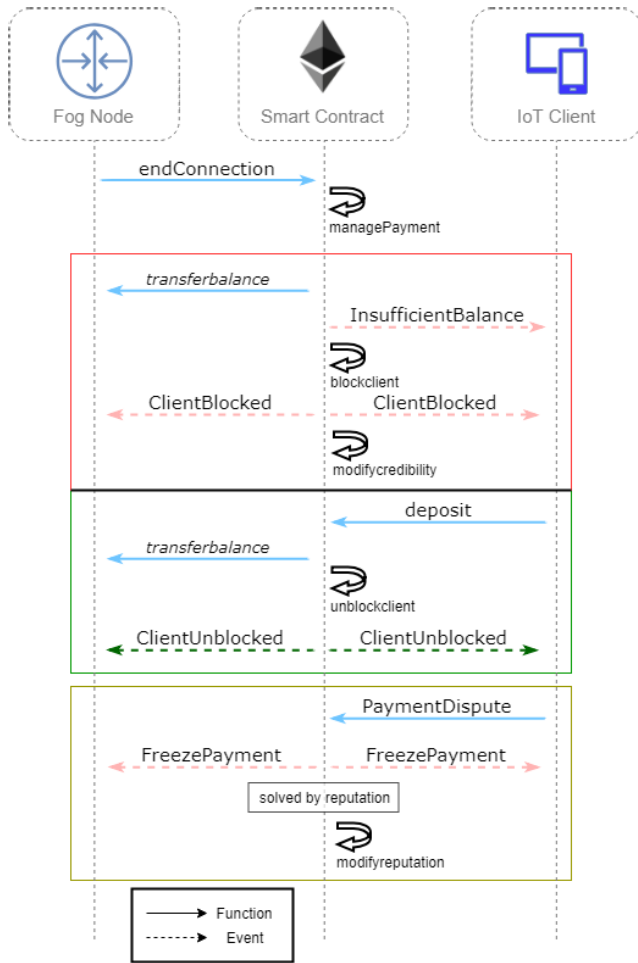
FIGURE 5. Interactions sequence for conflict and mistrust.

FIGURE 6. Smart contract code for depositing money into device's account.

Algorithm 1 explains the steps of how a device requests to establish a connection with a fog node to access its services. This algorithm validates the status of the device before

connecting it to the desired fog node. It checks if the device has sufficient balance in its account and it does not have any overdue amount. If these requirements are met, the device can connect to the fog node. Initially, a device should be subscribed to the fog node. After the subscription, the smart contract generates a random access key for its connection. The key is provided to the device and the fog node as they will be connecting off-chain which requires authentication. The key is generated using the keccak-256 hashing function or also known as SHA-3. This function creates a random 256-bit (32-byte) integer for each connection. To ensure the uniqueness of the key, the timestamp of the block, the sender's address, the gas left, and the hash of the block are used in generating the key.

Algorithm 1 Establishing Connection

Input: fog_node
1 fog_node is an Ethereum Address (EA).
2 **Modifier**: onlydevice
3 **if** *fog node registered* ∧ *device registered* ∧ *device has sufficient deposit balance* ∧ *device does not have outstanding balance* ∧ *device is not connected to a fog node* **then**
4   **if** *device is not subscribed to fog node* **then**
5   Subscribe to fog node.
6   **end**
7   Compute estimated remaining access time of the device.
8   Generate a random access key for the device.
9   Add device's EA to the list of connected devices of the fog node.
10   Set the device's status as connected.
11 **else**
12   revert.
13 **end**

Algorithm 2 shows the procedure of ending a connection between the device and the fog node. When the fog node terminates the connection, it provides the time of usage which the smart contracts use to compute the due amount. Although the fog node can only terminate the connection, the device can send a request directly to the fog node terminate the service. In both scenarios, the fog node is only authorized to end the connection. If the smart contract calculates the required fees and finds that the device does not have enough balance, the device is blocked by the contract. If the device is blocked, it no longer has access to the smart contract, the fog nodes, or their services. Moreover, when the reputation system previously mentioned is employed, the credibility of the device is modified when it utilizes the fog node services but does not pay for them. A device with a low credibility score is regarded as dishonest and its future feedback is given less weight. The devices also can object to the payment requested by the fog node if they believe that it is unfair. In this case, the reputation of the fog node and the device

```
function refundBalance() onlyClient public{
    require(clients[msg.sender].connected_fognode == address(0),
    "Client still connected to a fog node, cannot refund"
    );
    require(getDueAmount()==0,
    "Client has outstanding balance, cannot refund"
    );
    msg.sender.transfer(getDeposit());
    clients[msg.sender]=(client_type(0,address(0),true,0,0,0,address(0)));
    emit AmountRefunded(msg.sender,getDeposit());
}
```

FIGURE 7. Smart contract code for refunding device's balance.

credibility are taken into consideration. If the fog node is considered to be more trustworthy than the device, the payment is considered to be true. Nevertheless, the reputation score is deducted by a small amount related to the device's credibility. If many devices start objecting to the bills of a certain fog node, the reputation of that fog node is eventually reduced and is considered dishonest. The fog node is then punished by deducting from its balance that was deposited upon registration. This technique compels both the fog nodes and devices to act honestly. The monetization smart contract offers a way for the devices to retrieve the deposit upon their request. However, before the device is allowed to refund the rest of his balance, the smart contract makes sure it is not using any of the fog node service. Figure 7 shows the solidity code for refunding the device. The smart contract ensures that the device is not connected to any fog node and it is not utilizing any of their services. Moreover, the device should not have any amount overdue to any fog node. If these conditions are met, the device is able to refund its balance.

Algorithm 2 Terminating Connection

  Input: fog_node, amount
1 fog_node is an Ethereum Address (EA).
2 Amount is the total cost of service.
3 **Modifier**: onlyFog.
4 **if** *fog node registered* ∧ *device registered* ∧ *device and fog node are connected* **then**
5   **if** *device's balance > amount due* **then**
6   Deduct amount from device's account.
7   Transfer the amount to the fog node's EA.
8   **else**
9   Outstanding balance = Amount due - device deposit.
10   Record the fog node's EA that the devices owes.
11   transfer device's deposit to fog node.
12   Set device's balance to 0.
13   Flag device as blocked.
14   **end**
15 **else**
16   revert.
17 **end**

```
[vm] from:0x4b0...4d2db to:Monetization.connect(address)

transact to Monetization.connect errored: VM error: revert.
revert  The transaction has been reverted to the initial state.
Reason provided by the contract: "Deposit insufficient".
```
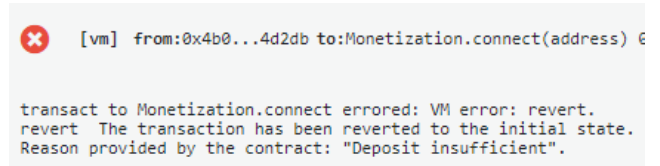
FIGURE 8. Exception message - Insufficient balance.

VI. TESTING AND EVALUATION

After the smart contract was deployed on Remix IDE, we performed exhaustive testing to check for inconsistencies, bugs, and vulnerabilities. The logic of the smart contract was validated and multiple scenarios were examined. The smart contract was deployed on a virtual Ethereum blockchain with multiple devices and fog nodes. The debugger in the Remix IDE was useful in inspecting all of the transactions and interaction between the different system participants. To observe the output of each step, events were triggered with sufficient information.

A. VALIDATION

In order to validate the logic of the smart contract code, multiple scenarios were tested and their outcomes are presented below. The owner deploys the smart contract on the virtual blockchain. The owner registers the fog node using its Ethereum address (EA) and its rate per hour in Wei (1 Wei = 0.000000000000000001 ether). The device registers in the smart contract and proceeds to deposit an amount of money in the account. After depositing, the device tries to connect to a fog node by providing its EA. It is worth noting that the device gets the EA of the fog node from off-chain resources which was explained in the previous work preceding this research paper [24]. If the device does not have enough balance in its account, an error message is going to appear as seen in Fig. 8 where the exception is triggered by the connect function.

After depositing the required amount, the device connects to the required fog node and the information about the device can be seen in Fig. 9. The figure shows the deposited amount, the EA of the connected fog node, estimated remaining time of connection, the 32-bit access key and information about overdue balance. In this case, the device does not owe any fog node and hence has 0 outstanding balance.

When the fog node wants to terminate the connection, it provides the smart contract with the amount that the device owes. In this scenario, the fog node requests more money that the device has deposited. As seen in Fig. 10, this triggers two events: first regarding the termination of the connection and second regarding the device blocking. The device can no longer access the fog nodes or smart contract services. To settle the issue, the device performs another deposit in the account, which will further trigger two events: one for announcing the deposit and the other for unblocking the device. In Fig. 11, the amount that has been deposited and the total account balance can be seen. In this case, the account
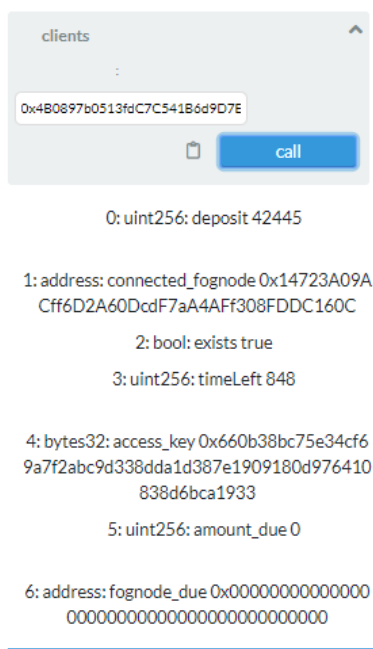
FIGURE 9. Device information after connecting to a fog node.

FIGURE 10. Event showing connection termination.

balance is less than the amount deposited as the rest has been used to settle the overdue amount.

B. COST ANALYSIS

In addition to verifying the logic of the smart contract, a cost analysis was performed to find the cost of operation. When the transactions of the smart contracts are executed on the blockchain, the cost is represented as gas. Remix IDE offers an estimated gas cost and execution cost when executing a function. Execution gas is a measure of the cost of executing the smart contract. While the transaction gas comprises the gas cost of executing the contract and sending it to the blockchain. Table 1 shows the transaction gas and the execution cost in USD. The gas cost interpretation is measured as of November 23, 2019, on the ETH Gas Station [25]. The gas price is assumed to be 1.5 Gwei. We can notice that the

FIGURE 11. Event showing deposit from device.

TABLE 1. Gas cost of Ethereum functions in USD.

Method name	Transaction cost	Execution cost	Cost (USD)
registerFog	66126	43254	0.00967
registerdevice	45737	24465	0.00547
deposit	45257	23985	0.00536
subscribe	45134	22454	0.00502
connect	96087	73407	0.0164
endConnection	51147	79422	0.01775
refundBalance	25694	19422	0.00434

methods spend less than $0.02. The connection initiation and termination evidently spend the most amount of ether as they include the most amount of processing.

C. SECURITY ANALYSIS

This section provides a concise security analysis of the implemented solution. We discuss how using blockchain technology helps secure the system against major security concerns. In addition, we perform a brief security analysis on the developed smart contract to ensure it is free from vulnerabilities that may be exploited.

- **Authorization and Accountability:** Different types of users in the system have access to different methods. In Solidity, modifiers can be used to check the identity of whoever is accessing the smart contract. Fog nodes and devices do not have the same level of privilege. Moreover, some aspects of the system are only accessible by the owner of the smart contract. For example, a device can request a connection with a fog node but only the fog node can terminate the connection. Moreover, each function call can be linked to the EA of the sender, therefore, all entities are held accountable for their actions.

- **Data Integrity:** All data stored on the blockchain is immutable and cannot be modified. When a transaction has been made or a function has been called, that record is not erasable. Moreover, to authenticate devices and fog service providers, a 32-bit access key is created by hashing the timestamp of the block, the address of the sender, the gas left, and the address of the smart contract.

This means that starting a new connection will require a new key and the integrity is maintained.

- **Non-repudiation:** By design, the transactions on any blockchain platform are recorded on a permanent ledger with the address of the sender. The devices and fog nodes cannot repudiate the initiation of any function call or money transfer. The transactions are tamper-proof and hashed in the block.
- **Availability:** Deploying our smart contract on the Ethereum network means it is available with all participants of this platform. Hence, the smart contract is stored in a decentralized manner and is always available on thousands of locations for the usage of devices and fog nodes.

VII. CONCLUSION

Currently, the pricing of the public fog nodes' services is highly inefficient as it typically involves manual payment and lacks necessary visibility, transparency, and trust. This paper has presented a blockchain-based cryptocurrency automated solution to enable consumption-based and dispute-free payment for fog node services. We used smart contracts to automate the payments, maintain the reputation of fog nodes, and to enable the customers to select their preferred fog nodes. The smart contracts ensured transparency and increased level of trust among public fog node providers and their customers. We used Ethereum blockchain platform and Solidity language to implement, deploy, test, and analyze the performance of our proposed smart contract and made it publicly available on Github. In addition, we calculated the Ethereum gas cost consumption to find the economic impact of the proposed smart contract. Finally, we performed security analysis to ensure the resilience of the smart contract against major security attacks.

REFERENCES

[1] F. Bonomi, M. Rodolfo, Z. Jiang, and A. Sateesh, "Fog computing and its role in the Internet of Things," in *Proc. ACM 1st Ed. MCC Workshop Mobile Cloud Comput.*, 2012, pp. 13–16, doi: 10.1145/2342509.2342513.

[2] AWS IoT. (2020). *AWS IoT for the Edge*. [Online]. Available: https://aws.amazon.com/iot/solutions/iot-edge/?nc=sn&loc=2&dn=4

[3] Google Cloud. (2020). *Edge TPU*. [Online]. Available: https://cloud.google.com/edge-tpu/

[4] IBM Cloud. (2020). *IBM Edge Computing*. [Online]. Available: https://www.ibm.com/downloads/cas/O8YVJY9Z

[5] Microsoft Azure. (2020). *Microsoft Azure Enables a New Wave of Edge Computing. Here's How*. [Online]. Available: https://azure.microsoft.com/en-us/blog/microsoft-azure-enables-a-new-wave-of-edge-computing-here-s-how/

[6] Cisco Ultra Services Platform. (2020). *Cisco's Multi-Access Edge Computing at a Glance*. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/ultra-services-platform/at-a-glance-c45-741450.html

[7] Nokia. (2020). *Nokia's Edge Cloud*. [Online]. Available: https://www.nokia.com/networks/portfolio/edge-cloud/

[8] D. Kim, H. Lee, H. Song, N. Choi, and Y. Yi, "Economics of fog computing: Interplay among infrastructure and service providers, users, and edge resource owners," *IEEE Trans. Mobile Comput.*, to be published, doi: 10.1109/tmc.2019.2925797.

[9] X. Li, C. Zhang, B. Gu, K. Yamori, and Y. Tanaka, "Optimal pricing and service selection in the mobile cloud architectures," *IEEE Access*, vol. 7, pp. 43564–43572, 2019, doi: 10.1109/access.2019.2908223.

[10] M. Crosby, P. Pattanayak, S. Verma, and V. Kalyanaraman, "BlockChain technology: Beyond Bitcoin," *Appl. Innov. Rev.*, vol. 2, nos. 6–10, p. 71, 2016. [Online]. Available: https://j2-capital.com/wp-content/uploads/2017/11/AIR-2016-Blockchain.pdf

[11] S. Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. Accessed: Dec. 22, 2019. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[12] K. Salah, M. H. U. 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.

[13] 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, doi: 10.1016/j.future.2017.11.022.

[14] H. R. Hasan and K. Salah, "Combating deepfake videos using blockchain and smart contracts," *IEEE Access*, vol. 7, pp. 41596–41606, 2019, doi: 10.1109/access.2019.2905689.

[15] K. Christidis and M. Devetsikiotis, "Blockchains and smart contracts for the Internet of Things," *IEEE Access*, vol. 4, pp. 2292–2303, 2016, doi: 10.1109/access.2016.2566339.

[16] F. Wortmann and K. Fluchter, "Internet of Things," *Bus. Inf. Syst. Eng.*, vol. 57, no. 3, pp. 221–224, 2015, doi: 10.1007/s12599-015-0383-3.

[17] A. Botta, W. De Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and Internet of Things: A survey," *Future Gener. Comput. Syst.*, vol. 56, pp. 684–700, Mar. 2016, doi: 10.1016/j.future.2015.09.021.

[18] J. Ju, M.-S. Kim, and J.-H. Ahn, "Prototyping business models for IoT service," *Procedia Comput. Sci.*, vol. 91, pp. 882–890, 2016, doi: 10.1016/j.procs.2016.07.106.

[19] D. Niyato, D. T. Hoang, N. C. Luong, P. Wang, D. I. Kim, and Z. Han, "Smart data pricing models for the Internet of Things: A bundling strategy approach," *IEEE Netw.*, vol. 30, no. 2, pp. 18–25, Mar. 2016, doi: 10.1109/mnet.2016.7437020.

[20] A. Javaid, M. Zahid, I. Ali, R. Khan, Z. Noshad, and N. Javaid, "Reputation system for IoT data monetization using blockchain," in *Advances on Broad-Band Wireless Computing, Communication and Applications* (Lecture Notes in Networks and Systems). Cham, Switzerland: Springer, 2019, pp. 173–184. Accessed: Nov. 20, 2019, doi: 10.1007/978-3-030-33506-9_16.

[21] A. Talukder and L. Zimmerman, "Cloud economics: Principles, costs, and benefits," *Cloud Computing* (Computer Communications and Networks). London, U.K.: Springer, pp. 343–360, 2010. Accessed: Nov. 20, 2019, doi: 10.1007/978-1-84996-241-4_20.

[22] H. Katzan, "Cloud computing economics: Democratization and monetization of services," *J. Bus. Econ. Res.*, vol. 7, no. 6, 2011. Accessed: Nov. 20, 2019, doi: 10.19030/jber.v7i6.2301.

[23] Amazon Web Services, Inc. (2019). *Pricing*. Accessed: Dec. 18, 2019. [Online]. Available: https://aws.amazon.com/pricing/

[24] M. Debe, K. Salah, M. H. U. Rehman, and D. Svetinovic, "IoT public fog nodes reputation system: A decentralized solution using Ethereum blockchain," *IEEE Access*, vol. 7, pp. 178082–178093, 2019, doi: 10.1109/access.2019.2958355.

[25] Ethereum. *ETH Gas Station*. Accessed: Jul. 27, 2019. [Online]. Available: https://ethgasstation.info/

MAZIN DEBE received the B.S. degree in computer engineering from the Khalifa University of Science and Technology, Abu Dhabi, United Arab Emirates, in 2018, where he is currently pursuing the degree with the Department of Electrical Engineering and Computer Science. He is also associated with researchers with the Center for Cyber-Physical Systems, Khalifa University. His research interests include blockchain technology, the Internet of Things (IoT), and fog computing.

KHALED SALAH received the B.S. degree in computer engineering (minor in computer science) from Iowa State University, USA, in 1990, and 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, United Arab Emirates. 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 with the Department of Information and Computer Science, King Fahd University of Petroleum and Minerals (KFUPM), Saudi Arabia, for 10 years. He has more than 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 is the Track Chair of the IEEE GLOBECOM 2018 on Cloud Computing. He serves on the editorial boards for many WoS-listed journals, including *IET Communications*, *IET Networks*, Elsevier's JNCA, Wiley's SCN, Wiley's IJNM, J.UCS, and AJSE. He is an Associate Editor of the IEEE BLOCKCHAIN NEWSLETTER.

MUHAMMAD HABIB UR REHMAN the bachelor's and master's degrees in Pakistan, whereby he received gold medals and 100% fee-waiver scholarships, from COMSATS University Islamabad, Pakistan, and the Ph.D. degree from the Faculty of Computer Science and Information Technology, University of Malaya, Malaysia. He is currently working with the Center for Cyber-Physical Systems, Khalifa University, United Arab Emirates, as a Postdoctoral Research Fellow. He is currently working on trustworthy Blockchain technologies for intelligent cyber-physical systems. His main research activities include research and development of trust models for decentralized and trustworthy artificial intelligence applications for cyber-physical systems. He has been an alumnae of DAAD's postdoctoral network, since September 2019. He is a Bright Spark Fellow. He has authored or coauthored 38 international publications including journal articles, conference proceedings, book chapters, and 3 magazine articles whereby his 3 articles are categorized as highly cited publications by Web-of-Science. His research interests include blockchain technologies, cyber-Physical Systems, secure key management, big data, edge computing, and industrial IoT. He is a Professional Member of IEEE.

DAVOR SVETINOVIC received the Ph.D. degree in computer science from the University of Waterloo, Canada, 2006. He worked as a Visiting Professor with the Massachusetts Institute of Technology (MIT) and a Postdoctoral Researcher with 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). He is currently an Associate Professor of computer science with the Khalifa University, United Arab Emirates, and a Research Affiliate with the Media Lab, Massachusetts Institute of Technology (MIT), USA. He has extensive experience working on complex multidisciplinary research projects. He has published more than 75 articles in leading journals and conferences. His current research interests include systems security and privacy, blockchain engineering, and artificial intelligence. He is a Senior Member of ACM.

- - -