IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 19, 2020, accepted April 26, 2020, date of publication April 29, 2020, date of current version May 14, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2991261

Blockchain-Based Decentralized Reverse Bidding in Fog Computing

MAZIN DEBE, KHALED SALAH, MUHAMMAD HABIB UR REHMAN, (Member, IEEE), AND DAVOR SVETINOVIC, (Senior Member, IEEE)

Center for Cyber-Physical Systems, Department of 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 Fog computing systems are designed to provide localized computation, storage, and communication services in close proximity to the endpoint mobile and IoT devices. Fog service providers typically monetize their service usage via centralized payment mechanisms in unverifiable and non-transparent manner. Therefore, there exists a need for a trust-enabling payment mechanism whereby fog service providers should be incentivized or penalized based upon the continuous feedback from endpoint devices. We propose a decentralized reverse-bidding scheme developed using the key features of blockchain and smart contracts. We develop a solution that allows the users or devices to initiate the bidding process by making a request for services to be provided by nearby public fog nodes, and these fog nodes to make bid offers in return. The proposed scheme ensures that all fog nodes on the network can equally and fairly make offers to win the bid. The bidding process incorporates the automated payments at the end of the service. Our solution is implemented using Ethereum smart contracts. It also integrates a reputation system for fog nodes and imposes a penalty for misbehaving nodes. Our solution is fully decentralized and provides a high level of trust, transparency, and security. In the paper, we present the system architecture, implementation details, and show the correct functionality of the overall proposed solution. In addition, we provide performance, cost, and security analyses of the smart contract code to demonstrate its effectiveness and robustness against major security concerns. The results show that the cost of running the smart contract remained less than three cents with the current Ethereum price (i.e., 183.22 USD/Eth). We have also made our smart contract code publicly available on Github.[1]

INDEX TERMS Blockchain, smart contracts, fog computing, IoT, Ethereum, auctioning, bidding.

I. INTRODUCTION

The network traffic between IoT devices and cloud data centers is predicted to triple in 2021 compared to five years ago [1]. At the same time, the number of connected devices has already surpassed the number of humans in the world; thus various service providers have made significant efforts for deploying middle layer between the devices and the cloud data centers in order to lower the operational costs and increase revenues [2], [3]. Fog computing systems enable computation, communication, and storage services via fog nodes (also known as Edge servers) near the data sources, such as onboard sensors in endpoint mobile and IoT devices [4]. These systems are traditionally designed as centralized three-tier architectures where the fog services are centrally orchestrated, considering the performance requirements (such as low-latency and bandwidth-efficiency) of endpoint applications [5]. Considering the pace of the current research on fog computing systems, it is perceived that these systems will enable a large variety of applications in smart environments to complement various personal and industrial use-cases such as smart homes, energy, healthcare, and smart cities [4]. Fog nodes, at the middle layer of fog computing systems, serve as intermediaries between the data-intensive mobile and IoT devices and cloud servers [6]. These nodes act as filtration points to minimize the bandwidth utilization over the Internet. In addition, local data processing in fog nodes results in redundancy elimination, efficient energy consumption, and optimal data transfer between IoT devices and centralized cloud systems [7].

Ideally, fog nodes should provide a variety of cloud services with various levels of Quality of Service (QoS) agreements considering the network availability, the ability to

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

[1]https://github.com/MazenDB/ReverseAuction

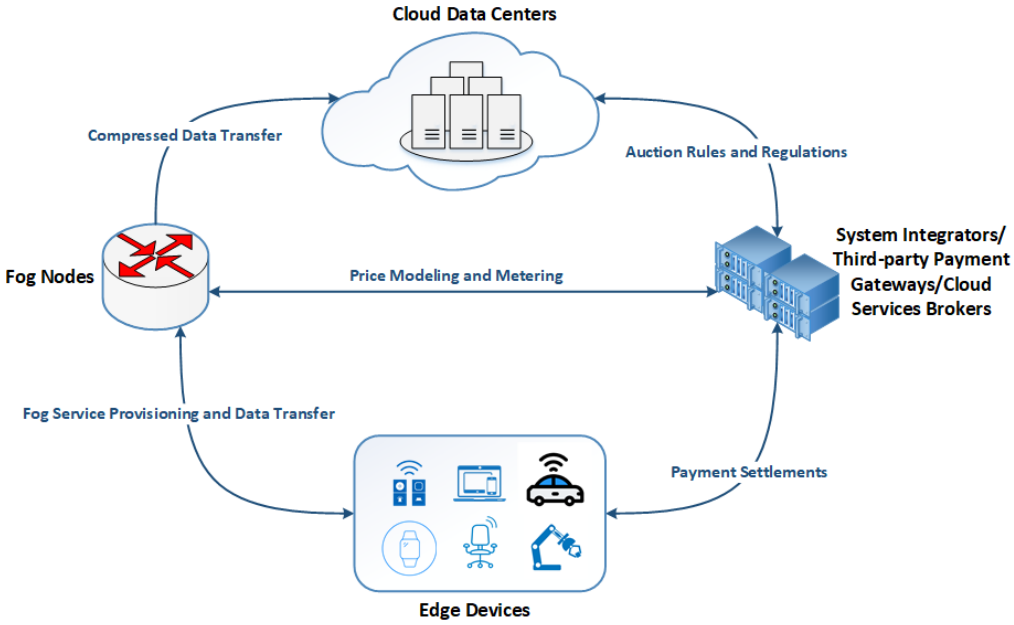

FIGURE 1. Reverse auctioning mechanism in centralized fog-cloud service architectures.

process, store, and transmit the data, and the computational and timeliness requirements of mobile applications on the connected endpoint devices [8]. We perceive that endpoint mobile and IoT devices will have several fog nodes in their proximity, which offer similar services with competitive pricing models. However, the service requirements of endpoint devices vary considering their need and affordability; thus, their preference to use the fog services change with the circumstances.

A. MOTIVATION

Fog service providers normally meter and monetize their services based upon the resource utilization by the endpoint devices. Numerous research works proposed bidding and auctioning mechanisms to maximize the profit of fog service providers and enhance user-experience for endpoint device users [9]. However, these auctioning and bidding mechanisms use forward-bidding strategies whereby the computation tasks are offloaded in fog servers considering the usage patterns, mobility, SLA requirements, and availability of computation and storage resources in fog edge servers and endpoint IoT devices [10]. Existing research work used reverse auctioning mechanisms to efficiently and collaboratively manage virtual machines on cloud data centers and fog servers [11]. However, to the best of our knowledge, existing literature still lacks the continuous feedback from users to actively incentivize or penalize the fog service providers.

In this paper, we propose a reverse-auctioning mechanism to select the reputable and reliable fog service providers. As opposed to a regular auction, the endpoint devices post the requests for required services with a maximum amount they can pay in a reverse auctioning manner (also called the reserve pricing mechanism) [12]. The auctioning process executes a reverse-bidding process where fog service providers offer their services at lower prices during the auctioning time period in order to win the customers. Generally, the winner of the auction is the fog node that claims to be able to provide required services at the lowest price, but a service user can opt for better QoS for a higher cost. Such an interaction is generally governed by a specialized server that organizes the service requests from endpoint devices, manages bids from fog service providers, takes into account the QoS of fog nodes, maintains the interaction between users of IoT devices and fog service providers, and ensures the delivery of service and secure payment system. However, a centralized authority that manages the bidding workflow, as shown in Fig. 1, poses multiple threats regarding application performance and security of data and services.

An automated decentralized system is needed in such a scenario to overlook the process of bidding and payment while mitigating the risks of centralization. Fortunately, the blockchain has been increasingly used for numerous sectors, including cryptocurrencies, IoT, and artificial intelligence [13]–[15]. The blockchain ensures secure, immutable, automatic transactions between different entities by design [16]. In addition, employing smart contracts allows for enforcing specific rules and regulations that orchestrate the bidding process in a way that conforms with the blockchain standards and constraints [17]–[19]. This means that a smart contract can overcome the security and trust concerns at both ends, i.e., the endpoint mobile devices and fog service providers. These entities entrust the system to fairly manage the transactions with no third party interference in the process [20].

B. CONTRIBUTIONS

This paper presents a way for endpoint mobile and IoT devices to post their requests for certain services and start an auction using fully decentralized smart contracts. This approach allows fog service providers to place competing bids to win the right to provide the service at a decreasing price. We propose a decentralized approach developed using the public Ethereum blockchain and its smart contracts to organize such an interaction in real-time. In brief, the main contributions of this paper are:

- We propose a blockchain-based approach for decentralized auctioning in fog computing environments. The proposed approach, as described in section III, uses a timer-based reverse-bidding and automated payment settlement scheme for efficient and fair auctioning among endpoint mobile and IoT devices and fog service providers.
- We integrate a reputation system into the proposed auctioning scheme. Our solution, section III, is designed to incentivize honest and good behavior by penalizing misbehaving users and fog nodes.
- We implement the proposed auctioning mechanism using public Ethereum smart contracts, and present algorithms for different functions and operations as detailed in section IV. The smart contracts automate the auctioning mechanism, execute a reverse-bidding scheme, update reputations, and settle payments among endpoint devices and fog service providers.
- We test and evaluate our proposed smart contracts, as presented in section V, to validate and verify the functionality of the reverse-auctioning mechanism. Finally, we perform cost and security analyses.

Section II describes the related work. Section III explains the proposed approach to enable the reverse-auctioning mechanism in a fog computing environment. Section IV presents the implementation details. Section V demonstrates the results, analysis, and discussion after testing the implemented smart contracts. Section VI presents the main conclusions.

II. RELATED WORK

This section highlights some of the previous work related to reverse auctioning mechanisms in cloud-fog applications. Forward auctioning enables the bidding process where the computational tasks are judiciously assigned to fog nodes considering the optimal utilization of computational resources and the computational needs of the endpoint devices. There exist a few early studies which use forward bidding mechanisms with [9], [10] or without blockchain integration [21]–[28], however, in this research we only focus on forward auctioning mechanisms.

Reverse auctioning, as opposed to forward auctioning, is a bidding process by service providers competing to provide services to a service user [29]. The service user requests a good or service, and service providers offer decreasing bids to win the users. The lowest bidder usually wins the right to provide the required services to the service user at the given price. This type of auctioning process is different from the regular auctioning mechanisms considering the service providers race to sell their services. This type of auction is often used by service users looking for contractors or subcontractors to provide the best price for the required services with maximum QoS offerings.

Reverse auctioning is mainly employed in centralized settings using Fog Integrated Cloud Architectures. For example, a double-auction mechanism considers multiple non-price attributes such as location, reputation, and computing power and suggests to prepare the pricing models accordingly [10]. This approach is more feasible than price-based reverse auctioning mechanisms as it entails more rational prices, computational efficiency, optimal budgeting, and truthfulness of fog nodes. However, it still faces the problem of centralization, whereby all the auctioning operations are governed by a centralized cloud server.

BRAIN is a blockchain-based approach that allows an Ethereum smart contract to conduct reverse auctioning where service users request specific Virtual Network Functions-as-a-service (VNFaaS) [30]. Infrastructure Providers (IPs) then enter a bidding contest that decides which IP possesses the sufficient infrastructure to host the VNFs requested by the service user at the best price. The authors present the design of their approach that demonstrates how an end-user first acquires a VNF, and during this acquisition, service users define their requirements that are later published in a smart contract. After the deployment of the smart contract, bids are placed by IPs and processed by a bidding manager according to a set of rules and regulations. The use-case described in this paper is similar to the use-case of our solution despite some inevitable variation.

The solution discussed in [31] showcases the use of a decentralized blockchain-based model for bilateral trades. The use of blockchain and smart contract introduces the concept of trust in an essentially trustless system. The lack of a third-party application that manages these trades creates a safe and transparent bilateral resource market. In this paper, the authors implemented and tested a double auction mechanism for resource sharing to increase cost-efficiency. Their presented workflow starts with the application transferring the bids of the traders to the orderer, who then proceeds to forward it to the participating peers. After pairing applicable buyers and sellers and authorizing the transaction, bids are sent back to the orderer to be posted on the blockchain. The final step is the formation of the blocks that include the successful bids and processed in the form of transactions.

The authors in [10] present a mechanism to improve the blockchain by sharing resources between blockchain miners. The paper proposes an auctioning-based resource market for fog/cloud service providers taking into consideration the blockchain network effects function. This function describes the security of a blockchain as a function of the computational power available in the network. Among other external factors, the suggested model also considers the competing

miners or the hash power function. The hash power function correlates the possibility of a miner being able to mine a block on the blockchain, depending on its computational capabilities. The approach allows decentralized applications (DApps) to be deployed for the process of allocating the appropriate resources for the corresponding entity. The paper describes the social welfare maximization problem for two types of bidding:

- *Constant demand scheme:* In this type of bidding, a limitation is imposed on each miner where it can only bid for an equal amount of computing resources. In this case, the proposed approach was able to reach the optimum social welfare using a constant demand bidding auction mechanism.
- *Multi demand scheme:* As opposed to the previous scheme, in the multi-demand scheme, miners get more freedom in bidding for computing resources. Miners here can submit bids for an arbitrary number of resources, but this results in an NP-hard optimization problem. Two efficient approximate algorithms have been developed to solve this problem (which are FRLS and MDB), both of which are efficient, truthful, and provide sub-optimal social welfare.

In the prior work, a monetization model has been developed for services provided by public fog nodes and implemented on the Ethereum blockchain [32]. The solution enabled endpoint IoT and mobile devices to subscribe to public fog nodes in order to take advantage of their services. The decentralized blockchain-based approach offered an automated method whereby endpoint devices subscribe, connect, and make use of services of fog nodes via smart contracts. The paper also defined a way for endpoint devices to resolve disputes with public fog nodes in a safe, transparent, and decentralized manner, which ensures trust in the system. Automated payments are supported at the end of each interaction through the smart contract itself. In addition, a reputation system can be integrated with this solution, which enables endpoint devices to automatically select and access their preferred fog nodes based on their reputation scores. The reputation score and the credibility score of endpoint devices are continuously updated, considering the feedback following each round of fog-device interactions. However, due to the abundance of fog service providers, the mentioned solution lacks a systematic way for endpoint devices to choose the best fog service provider for specific tasks considering the fairness and efficiency requirements of endpoint mobile and IoT applications [33], [34]. Hence, a smart contract-based auctioning mechanism would be a suitable solution to manage the communication between fog nodes and endpoint devices until they reach an agreement on the expected QoS and prices of required services.

III. PROPOSED APPROACH

This section presents the design and system architecture for the proposed reverse auctioning mechanism. The proposed approach enables endpoint mobile and IoT devices to request a preferred set of fog services at the lowest feasible rates. It utilizes smart contract technology and deploys it on the public Ethereum blockchain. The smart contract executes all auctioning rules and governs the interactions between endpoint devices and fog service providers. These interactions include operations for formulating the auction, placing bids and evaluating them, closing the auction, connecting the client to the fog node, and automatically settling the payments. All the participating entities (such as endpoint devices, fog service providers, and smart contracts) on the blockchain are identified by their unique Ethereum Address (EA) which is used to access the functionalities of each specific smart contract and transfer Ethers, which represent the native cryptocurrency, on Ethereum blockchain.

Fig. 2 presents the overview of system architecture comprising the smart contract that manages the bidding process between the endpoint devices and the fog service providers. After registering as service users, endpoint devices request the smart contract to set the preferences for QoS requirements and start auctioning for fog service providers. In addition to the QoS requirements, the endpoint devices also set a minimum expected reputation value of required fog nodes so that any fog node with a lower reputation score could be denied from participating in the auctioning process. As the endpoint devices request to start the auctioning, they deposit an amount, at least double the asking price, for the service. As a worst-case scenario, if the lowest bid equals the asking price, the endpoint devices still have an equal amount of money at stake in case of malicious behavior. This deposited amount serves as collateral to incentivize honest feedback by the endpoint devices after the connection termination. The auction is open for bidding, and preregistered fog service providers can make bid offers in order to gain the right to provide the specified services to the endpoint devices. The fog service providers also deposit an amount equal to their bidding offer as a guarantee of QoS-compliant service delivery. When the duration of the auction finishes, the endpoint devices stop the auctioning, which finalizes the winning bid and selects the fog service provider. The smart contract transfers the fees and establishes the connection. Upon connection termination, the remainder of the balance is returned back to each of the endpoint devices and the fog service providers accordingly. In addition, the auctioning smart contract can access the reputation and credibility scores of service users and fog service providers. Endpoint devices need to access the reputation scores of fog nodes and evaluate the bids accordingly.

The aforementioned system architecture comprises the smart contract, the bidding fog service providers, and the Ethereum-enabled endpoint mobile and IoT devices. The latter two do not have a logic layer but are also identified by unique EAs like the smart contract and are referred to as Externally Owned Accounts (EOAs). However, EAs and EoAs are used to transfer and receive cryptocurrency on the Ethereum blockchain and are linked to their data in the smart contract. More explicit details about the

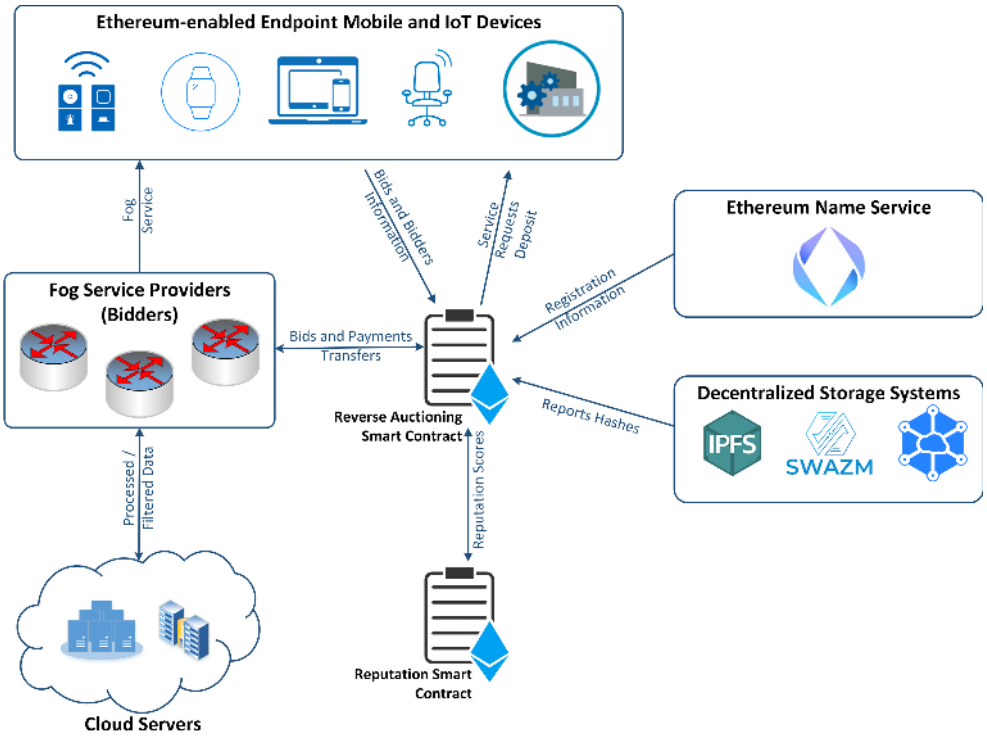

FIGURE 2. Proposed decentralized architecture for reverse auctioning and automated payments.

different entities, involved in the auctioning mechanism, are explained:

- *Fog service providers:* Fog service providers manage and configure fog nodes that are available publicly and act as intermediaries between cloud servers and endpoint devices. These nodes offer services for endpoints to enable data processing, data storage, and data analytics. Fog services reduce the latency, minimize the bandwidth utilization, and improve the overall performance of endpoint applications. In a public setting, the large number of fog service providers are available, which provides similar services at almost similar rates; hence they call for an auctioning mechanism to regulate the competition. In such a case, each fog service provider can leverage its strong characteristics. In addition, fog nodes can serve as much endpoint devices as its bandwidth allows and are not forced to surpass that which might degrade its performance and thereby its credibility.
- *Ethereum-enabled Endpoint Mobile and IoT devices:* Endpoint mobile and IoT devices are the most targeted type of device for this approach in addition to other fixed and stationary IoT applications. Endpoint devices are required to perform tasks, including sensing and collecting data from their proximal environment. The data generated by these devices gradually accumulate into huge datasets hence require efficient and timely data processing in order to reduce bandwidth consumption and data storage requirements. However, endpoint devices typically do not want to compromise on the QoS; that is why they require the services of an external fog layer. Endpoint devices broadcast their request to the available fog service providers and state their maximum reserve price and host a reverse auction where fog service providers bid on servicing the endpoint devices at a lower cost. Endpoint devices specify their priorities when asking for a service. For instance, an endpoint device would give high importance to latency and response time for a real-timeliness, and it may compromise on its storage requirements. After selecting a specific fog node, the endpoint closes the auction and connects to the fog node and provides its feedback after the connection ends. An endpoint device might not simply choose the lowest bidder. The discussed reputation system can be utilized, and a decision is based on the reputation score of the selected fog node and the pricing model it offers. Moreover, any suspicious behavior from the endpoint device, who is trying to provide false feedback, could result in holding the deposited amount and lowering its credibility score.
- *Reverse Auctioning Smart Contract:* The Ethereum smart contract manages the auctioning mechanism held by the endpoint devices. The smart contract has the ability to accept and hold deposits from endpoint devices and fog service providers. After the auction is closed,

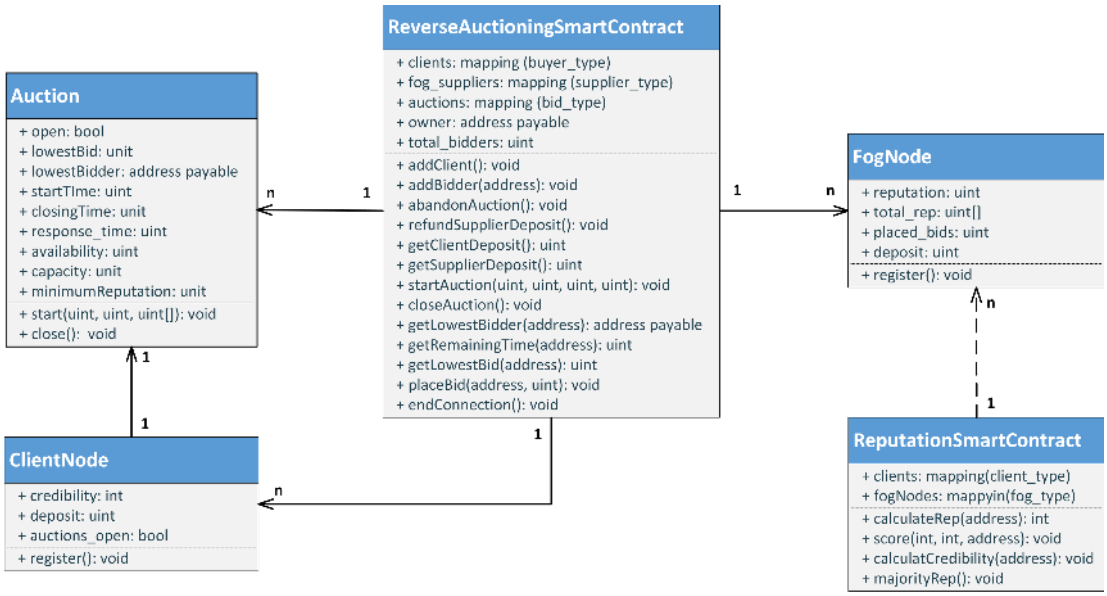

FIGURE 3. Relationship among different entities of the smart contract.

the smart contract transfers the due amount to the fog service provider and returns the deposit to corresponding entities. To add another level of security to the contract, not all members can access the full set of features or calls. Each member has limited privileges enforced by modifiers. For instance, only the owner of the contract can add suppliers, and endpoint devices can start an auction, while fog service providers can only place their bids.

- *Decentralized Storage Systems:* The metadata of the fog service providers and endpoint devices need to be stored. However, a traditional centralized database is not compatible with the blockchain as it creates a weakness that the blockchain was used to overcome. Therefore, decentralized storage such as InerPlantetary File System (IPFS), Filecoin, and swarm is associated with the smart contract where all the information can be stored securely. These systems can be used to store not only text data but also images and videos in a decentralized manner.
- *Ethereum Name Service:* The ENS is a service that benefits the readability of addresses by giving humane readable names to meaningless Ethereum Addresses. ENS is a distributed service that registers Ethereum clients and associates names to them, similar to a hostname. These names help with knowing the user's identity, association, and general profile. These names are stored in a decentralized, immutable ledger.

Fig. 3 shows the members of our auctioning-based solution and the interactions between their representative classes. The *ReverseAuctioningSmartContract* manages all endpoint devices and fog service providers, and the auctions held between them. It orchestrates the entire auctioning and bidding process according to a predefined set of rules on the blockchain. Each bidding fog node provider and endpoint devices are linked with the deposited collateral in addition to a reputation score and credibility score, respectively, which are computed by the designated reputation score smart contract as described in [5]. An Auction is a passive entity that, unlike others, does not require a unique EA for itself. It is linked to the EA of the endpoint device that started the auction, and it comprises a data structure that holds the metadata of the auction in addition to information about the bidders and bids.

The sequence diagram that encompasses all of the interactions in an exemplary auction held by endpoint devices in our system is shown in Fig. 4. The diagram includes the typical calls and triggered events and some of the common error messages that occur. After the smart contract registers the set of involved endpoint devices and fog service providers, they can access their assigned functionalities of the smart contract. Before endpoint devices can start an auction, a deposit needs to be transferred to ensure honest conduct. Once an endpoint requests for auctioning, the floor becomes open for reverse bidding. Fog service providers start engaging in the bidding process by submitting their offers. Along with each offer, an amount equivalent to the reserve price is to be transferred as a monetary deposit. When a better offer is made, the smart contract returns the full deposit to the previous bidder and takes the new deposit. The endpoint device closes the auction when it is satisfied with a bid. The smart contract transfers the service fee to the fog service provider. After the fog node has provided the service to the endpoint device, the connection between the two entities is ended, and the remaining balance of their deposits is returned accordingly.

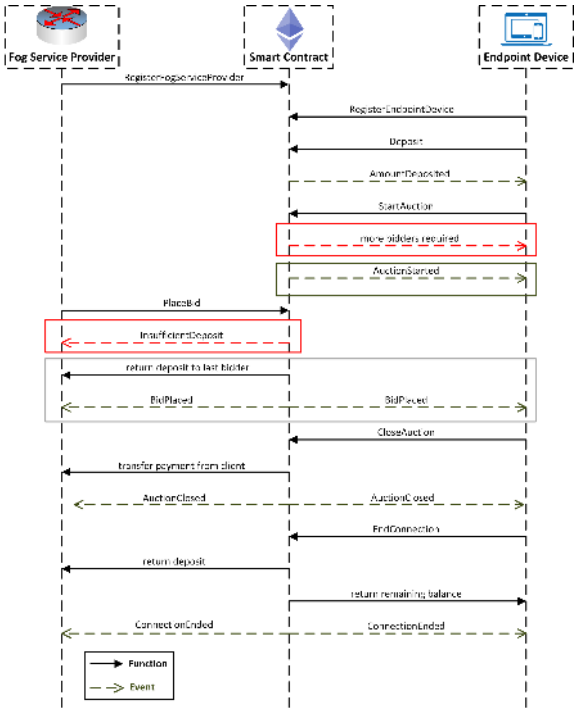

FIGURE 4. Interactions showing the function calls exhibiting reverse auctioning mechanism.

IV. IMPLEMENTATION

Our solution was implemented using Ethereum Remix in Solidity programming language, which provided us with an environment to test and simulate smart contracts and to validate and verify the business logic, check for errors, debug code, and test performance. Solidity provides a built-in mappings data structure that links an EA to an arbitrary number of variables. We have used mappings to save information about all running auctions and the fog service providers and endpoint devices. At first, the contract was deployed on the Ethereum network by the contract owner.

The first step for starting the auctioning mechanism after the endpoint device is registered is to deposit an amount of Ethers as a guarantee. The amount deposited should be equal or more to double the maximum cap requested for the service required. This a pessimistic approach to guarantee that even if only one fog node placed a bid equal to the reserve price of the auction, the endpoint device still has a considerable amount deposited as collateral.

The endpoint device then requests the smart contract to start auctioning by giving the contract the starting price, a duration time for the auction, a priority vector, and a minimum reputation for fog nodes. The priority vector contains the preferences of the endpoint device in regard to the characteristics of the fog node. For example, some endpoint devices prefer faster response, and some prefer a higher storage space.

The minimum reputation score is given to maintain a level of confidence in the fog nodes. This introduces a trade-off between the reputation score and the cost of service. As the minimum reputation increases, the possibility of low bids offered by less reputed fog nodes shrinks, but higher QoS is expected. For a fair and successful auction, the auction cannot start until a minimum number of bidders have joined the auction. This should not cause an issue, given abundant clients available in a public place. At this point, the auction is open and accepting offers from bidders.

Algorithm 1 Placing Bids by Fog Service Providers

Input: endpointAddress, rate
%endpointAddress is an Ethereum Address (EA).

Modifier: onlyFogServiceProvider
if *fog service provider is registered* $\wedge$ *auctioning is open for bidding* $\wedge$ *ether transferred* $=$ *rate* $\wedge$ *fog node reputation*>=*minimum reputation* **then**
  if *Previous bid has been placed* **then**
    if *offered price* >= *previous Bid* **then**
      Revert.
    end
    Return deposit of the previous bidder.
    Reduce *placed_bids* attribute of previous bidder by 1.
    Reduce *deposit* attribute of the previous bidder the amount that was offered.
  else
    if *offered price* > *previous Bid* **then**
      Revert.
    end
  end
  Increment the total number of bidders.
  Increase *placed_bids* of current fog service provider.
  Record the deposit offered.
  Update the Auctioning data to the new bidder address and rate offered.
  Broadcast the new bid to all fog service providers.
else
  revert.
end

Algorithm 1 shows how bidders can submit their offers to the smart contract and compete with the other fog service providers. The algorithm first ensures the auction is open and available for bidding. The offered rate should be less than the previous offer to be considered unless it is the first bid made where the bidders can offer a price equal to the reserve price. Moreover, the supplier also needs to transfer an amount equal to his offer to serve as a deposit and meet the reputation requirements. If all the conditions are satisfied, the deposit is recorded and kept in the smart contract, and the record of the fog service provider is updated accordingly. The smart contract code described in the algorithm can be seen in Fig. 5.

When the auction time expires, the endpoint device calls the function to close the auction if sufficient bids have been placed, as seen in Algorithm 2. Once this call has been made,

```
function placeBid(address buyer, uint rate) payable onlySupplier public{
    require(clients[buyer].exists,
    "Entered address does not refer to a client"
    );
    require(Auctions[buyer].open,
    "This Auction is not open for bidding"
    );
    require(rate<Auctions[buyer].lowestBid,
    "Please place a lower bid"
    );
    require(msg.value==rate,
    "Insufficient Deposit"
    );
    require(fog_suppliers[msg.sender].reputation>=Auctions[msg.sender].minRep,
    "Minimum Reputation requirement not met"
    );
    fog_suppliers[getLowestBidder(buyer)].deposit-=getLowestBid(buyer);
    fog_suppliers[getLowestBidder(buyer)].placed_bids--;
    if(getLowestBidder(buyer)!=address(0)){
        getLowestBidder(buyer).transfer(getLowestBid(buyer));
    }
    fog_suppliers[msg.sender].placed_bids++;
    fog_suppliers[msg.sender].deposit+=msg.value;
    Auctions[buyer].lowestBid=rate;
    Auctions[buyer].lowestBidder=msg.sender;
    emit BidPlaced(msg.sender, buyer, rate);

}
```

FIGURE 5. Smart Contract code for placing bids by fog service providers.

Algorithm 2 Closing an Open Auction and Payment Settlement

Modifier: onlyEndpointDevice
if *auction is open for bidding* $\wedge$ *the closing time has passed* $\wedge$ *at least one bidding offer has been made* **then**
    Transfer service fee to the fog service provider.
    Decrement *placed_bids* attribute of fog node.
    Set auction status to *closed*.
    Reduce endpoint's deposit according to the service fee.
    Broadcast Auction Closed Event. `/* After connection termination */`
    Transfer the remaining balance of the deposit to the endpoint device.
    Return deposit of fog service provider.
else
    revert.
end

no further bids are accepted by the smart contract, and the winning bid has been selected. The bidder with the best offer receives the service fee that has been agreed on, and the fog-endpoint device interaction commences. This is also reflected in the data of the endpoint device saved in the smart contract where the deposit and other information are updated. After the connection ends between the fog node and the endpoint device, the smart contract calculates the fees, and both parties get back their shared amount from the deposited amounts.

In addition, a fog service provider can abandon the auctioning; however, the fog node should not have any bids placed. If the fog node has no pending bids, the smart contract returns any remaining deposit recorded and decrements the total number of bidders. Endpoint devices as well can request a refund for their deposit. The smart contract confirms whether there are any open auctions linked to the EA of the endpoint device. The endpoint device is also not allowed to quit if it has any active connections or unsettled payment with a fog node.

V. TESTING AND EVALUATION

We implemented our proposed solution using Ethereum Remix. We tested the code and validated the functionality. Remix allows developers to access a virtual blockchain with multiple virtual Ethereum Accounts (e.g., EAs and EoAs), for testing purposes. Multiple plugins are also available for debugging, code analytics, and other advanced features. Remix also has a console to explore transactions, outputs of functions calls, and system variables.

A. VALIDATION

The smart contract was first deployed on the blockchain, which gave the smart contract a unique EA. For testing the business logic, the owner registers two fog service providers in the smart contract. Two endpoint devices also register themselves in the system. After that, each endpoint device starts auctioning with different attributes. The fog service providers then compete by providing bids to the auction. Multiple scenarios were demonstrated to confirm the compliance of the code to the system requirements. For demonstration, some of the cases are highlighted.

An endpoint device sends a request to start auctioning for a service with a reserve price of 25 Wei and a minimum reputation of 80 to the blockchain and specifies the duration of the auction. We selected the reputation score (80) and the reverse price (25 Wei) just for testing purposes — a true market price will depend on the provided services and market conditions. We presented the details of the reputation calculation scheme in our previous work [5], whereby the reputation scores represented the aggregated feedback provided by endpoint devices. In addition, the smart contract considers the credibility of endpoint devices as well. This approach ensures trustworthiness on the integrated reputation system as well as it ensures that the endpoint devices to behave honestly and responsibly. However, the level of credibility of clients does not affect the initial reputation scores of fog service providers who are still new to the system. The reputation score given to a fog service provider is calculated, as shown in Eq. 1.

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

fn represents the address of the fog node being evaluated.

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

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

$Cr(n)$ is the credibility of endpoint device 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.

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

The smart contract records this request and confirms if the endpoint has enough deposit or have transferred enough money. If the endpoint has sufficient balance, the smart contract approves the auction and broadcasts the event announcing the opening of the auction with its EA and reserve price, as shown in Fig. 6. The fog service providers are now allowed to place bids, and they are using EoA of one of the endpoint devices, we placed a bid with a rate of 20 Wei for that service. The auction of that endpoint device is shown in Fig. 7, and as it can be seen, the lowest bid and bidder address are shown along with the timestamps of start and end of the auctioning period. For such a bid, the fog service provider had to transfer 20 Wei to the smart contract that is now held as collateral until the end of the auction.


```
"from": "0xec5bee2dbb67da8757091ad3d9526ba3ed2e2137",
"topic": "0x44c53be110c6aa83aa83cd02e351ed172359268272ee1b5d31c0fe48db35c6c7",
"event": "AuctionStarted",
"args": {
        "0": "1581509312",
        "1": "40",
        "2": "90",
        "closingTime": "1581509312",
        "startPrice": "40",
        "MinRep": "90",
        "length": 3
}
```


FIGURE 6. Event showing the started auction.

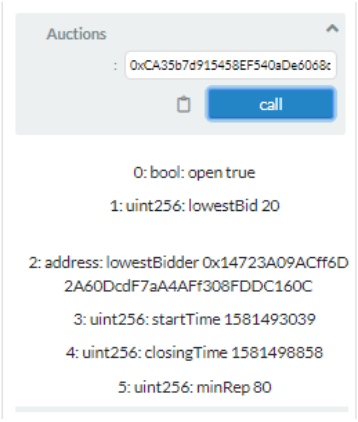

FIGURE 7. Auctioning details after submitting a bid by a fog node.


```
"from": "0xdc84977a2078c8ffdf086d618d1f961b6c546222",
"topic": "0x7787e67b4b8529b1aa2c1a59dbf0f0e766d63860956b86f33f5d870088235501",
"event": "AuctionClosed",
"args": {
        "0": "1580585908",
        "1": "0xCA35b7d915458EF540aDe6068dFe2F44E8fa733c",
        "2": "0x14723A09ACff6D2A60DcdF7aA4AFf308FDDC160C",
        "3": "20",
        "closingTime": "1580585908",
        "clientAddress": "0xCA35b7d915458EF540aDe6068dFe2F44E8fa733c",
        "lowestBidder": "0x14723A09ACff6D2A60DcdF7aA4AFf308FDDC160C",
        "lowestBid": "20",
        "length": 4
}
```


FIGURE 8. Event showing closing an auction.

When the endpoint device wishes to close the auction, it sends a request to the smart contract. The smart contract checks the duration of the auction and, if the ending time has passed, it marks the auction as closed, and bidders are not allowed to place any more bids. Fig. 8 shows the triggered event when an auction is closed. The address of the endpoint device who hosted the auction, as well as the winning bid and fog service provider, are all mentioned in the event. After the service delivery, the endpoint device requests the connection termination, and each deposit is returned to the respective fog node.

If a fog node wants to quit the auction, the smart contract offers a method to release the fog node. The smart contract also refunds whatever remaining balance is held from that fog node. However, a fog node cannot quit and get a refund if it has placed a bid that is still the best offer made to any endpoint device. The only way this fog node can be released is if another fog service provider presents a better offer for the endpoint. Otherwise, the fog node is obliged to service the auctioning endpoint device in order to retain its deposit. Fig. 9 shows the error message shown in the console when a fog node tries to quit the auction with a pending bid. The endpoint device also can request a refund if it is not currently connected to a fog node that is providing it with services.

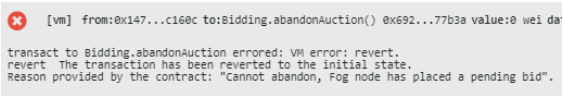

FIGURE 9. Error message showing the fog node cannot leave the auction.

TABLE 1. Gas cost of Ethereum functions in USD.

Method name	Transaction cost	Execution cost	Cost (USD)
addClient	48031	26759	0.0073
addBidder	70352	47672	0.01301
startAuction	130964	108796	0.0297
placeBid	142263	119391	0.0326
closeAuction	29118	36964	0.01008
endConnection	24971	28670	0.00783
refundSupplierDeposit	28368	7096	0.00193
abandonAuction	22436	23599	0.00644

B. COST ANALYSIS

This subsection shows a brief cost analysis of the Ethereum smart contract code and the function calls. The cost of the execution of the smart contract code comes from the gas cost of execution. When a transaction is executed on the Ethereum blockchain, it costs gas to send it to the Ethereum blockchain and to actually execute that command. Remix offers a highly useful feature that approximates the execution and transaction gas costs. Upon performing a transaction, the console outputs the estimated gas cost for that transaction. The cost depends on the complexity of the function called, inputs and the state of the smart contract code. While the gas cost is not fixed compared to fiat currency, the approximate cost of each transaction at the time of this writing is presented in Table 1 and cost of using the smart cost functions with transaction frequencies is depicted in Fig 10 whereby we list each function with its cost considering its gas cost and its corresponding cost in US dollars measured on the 30th of January, 2020. The price on the ETH Gas Station [35]

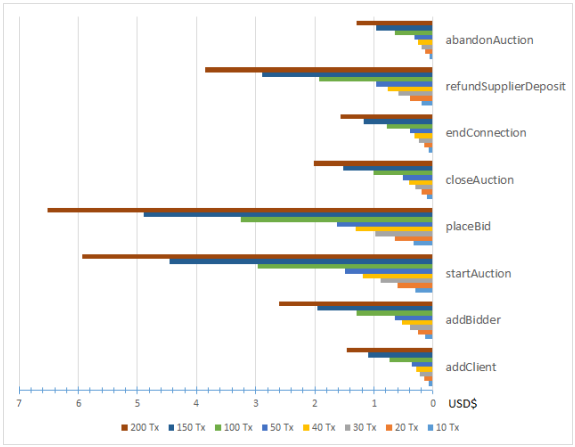

FIGURE 10. Cost vs. transaction volume of using smart contract functions.

is assumed to be 1.5 Gwei, and 1 Ether costs 183.22 USD. All function calls cost less than $0.03, which is considered a low price.

C. QUALITATIVE COMPARISON

The qualitative comparison with existing state-of-art systems (as summarized in Table 2) shows that our proposed system offers superior support to endpoint devices and fog service providers. Existing systems either centralized [31], lack in support to reputation mechanisms [10], [30], [31], or they do not enable support to smart contracts [10], however, our proposed system offers fully decentralized public blockchain-based smart contracts to maintain location-aware reverse bidding. Moreover, our proposed system ensures trust in fog service providers by employing smart contract-based reputation mechanisms, which result in more involved and responsible participation of fog service providers.

TABLE 2. Comparison with state-of-the-art related work.

Features	*[10]*	*[30]*	*[31]*	*Our Work*
Decentralized	✓	✓	✗	✓
Blockchain-based	✓	✓	✓	✓
Smart Contract	✗	✓	✓	✓
Reputation	✗	✗	✗	✓
Endpoint Credibility	✗	✗	✗	✓
Reverse-bidding	✗	✓	✗	✓
Location-aware	N/A	✗	NA	✓
Computation-aware	✓	✗	NA	✗

D. SECURITY ANALYSIS

The main objective is to design secure smart contracts that can withstand the known security attacks. In order to ensure the bug-free secure smart contract code, We analyzed the proposed smart contract using two reliable security tools, namely Oyente and Security. The analysis proves that our smart contract raises 'false' flags using a *low-level call* method, which generates alerts in case of any run-time exception or malicious code execution in the EVM. In addition, we used *pull-based external calls* in order to ensure the safe execution of smart contract code on underlying EVMs on the Ethereum blockchain network. Finally, we *explicitly* labeled functions and state variables in order to ensure the well-defined accessibility of data modifiers on the blockchain network.

A major risk of calling external contracts is their ability to control the execution of smart contract code and perform unexpected manipulation in transactional data. These types of bugs in a smart contract code could exist in multiple forms. The attackers can execute the malicious code and reenter in the smart contract by recursively calling the value transfer function and performing multiple transactions. Considering the *reentry attack*, we used *send()* function instead of *call.value()* therefore our smart contracts prevent execution of any external code. The *cross-function race conditions* are another form of attack where a potential attacker can replicate the attack from two or more different functions having shared state variables. The severity of attack could be disastrous when cross-function race conditions span over multiple contracts. Therefore, we avoided cross-function and cross-contract race conditions by minimizing external calls, and these calls are made only when the functions completely traversed through the internal state models of the functions. Using *mutual exclusion*, strategies could be another possible solution which we will explore in our future research works.

In the case of verifying that the fog node is the right node that the endpoint device should authenticate itself to, the blockchain name service (BNS) like ENS [36] could be used. ENS works in the same manner as DNS by replacing the long hashed addresses (like EA) to human-readable names in a secure and decentralized way. ENS does not suffer from the security issues the DNS has as it is built on smart contracts on the Ethereum blockchain [36].

Moreover, we performed byte-code level debugging analysis of proposed smart contracts using the semantic framework proposed in [37]. We analyzed the smart contracts in terms of *call integrity* and *atomicity*. Ethereum has a history of DAO bug, which resulted in the loss of 60 million US dollars from Ethereum's blockchain network. The DAO bug allowed the *reentry of attackers* on the blockchain who redirected payments to specific EAs. The semantic analysis shows that our smart contracts comply with call integrity property by not allowing any intermediate calls during mining and verification and atomicity by ensuring complete transaction execution and value transfer before processing the next transaction.

Listed below are the key security features accounted for in our solution:

- *Availability:* The blockchain is a multi-node decentralized network; hence it creates resilience with multiple points-of-failure and multiple points-of-resumption to ensure high availability of network resources. Therefore, smart contracts uploaded to the blockchain network are available on all participating nodes of the blockchain. This eliminates the single-point-of-failure vulnerability and limits downtime.
- *Authorization:* As previously mentioned, the smart contract uses modifiers to ensure each participant has access

to specific parts of the system. Some functionality can only be accessed by the owner of the contract. Most other functions can either be accessed by buyers or suppliers, not both. Similarly, information about an endpoint device such as its deposit can only be accessed by that endpoint device itself. No other member has access to such relatively sensitive information.

- *Non-repudiation:* The blockchain is a permanent immutable ledger that saves all transactions on its network. Once added, a block cannot be tampered with and cannot change. This means that fog nodes and endpoint devices cannot deny having executed a transaction. This is specifically important because suppliers should not be able to deny making an offer at a specific rate. If a fog service provider submits an offer, it is unchangeable, and the fog node is obligated to commit to delivering the service as per the agreement.

VI. CONCLUSION

In this paper, we presented a reverse auctioning solution for bidding for services provided by public fog nodes developed using blockchain and smart contracts. The proposed solution automates the process of bidding and payment in a completely decentralized manner without the involvement of a trusted third party. The bidding process involves having the endpoint mobile and IoT devices advertise to adjacent public fog nodes for needed services. Smart contracts are used to govern all the interactions among participating fog nodes depending on a predefined set of rules. The Ethereum smart contract code has been publicly made available in a Github repository.

Our blockchain-based solution addresses misbehaving endpoint devices and fog service providers in which dishonest behavior is penalized by reducing the return of deposited funds. The solution incorporates a reputation system that is designed to penalize the credibility and trustworthiness of misbehaving nodes. Our solution was implemented using Ethereum Remix for code deployment, testing, and analysis. All key functionalities of the reverse auctioning mechanism were shown to work as expected. The gas cost for invoking all blockchain transactions were shown to not exceed 0.03 USD at the time of testing. The low cost supports the idea of deploying our blockchain-based solution for real-world use cases involving bidding. Finally, security analysis was presented to discuss the robustness of the blockchain-based approach. To further enhance this solution, we want to perform an end-to-end system implementation and testing. We are looking to deploy our smart contract in a real environment along with decentralized applications (DApps) on Ethereum blockchain.

REFERENCES

[1] *Cisco Global Cloud Index: Forecast and Methodology, 2016–2021 White Paper*, Cisco, San Jose, CA, USA, 2018. Accessed: Jan. 25, 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.html

[2] S. P. Singh, A. Nayyar, R. Kumar, and A. Sharma, "Fog computing: From architecture to edge computing and big data processing," *J. Supercomput.*, vol. 75, no. 4, pp. 2070–2105, Apr. 2019, doi: 10.1007/s11227-018-2701-2.

[3] M. Mukherjee, L. Shu, and D. Wang, "Survey of fog computing: Fundamental, network applications, and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 1826–1857, 3rd Quart., 2018, doi: 10.1109/comst.2018.2814571.

[4] P. Bellavista, J. Berrocal, A. Corradi, S. K. Das, L. Foschini, and A. Zanni, "A survey on fog computing for the Internet of Things," *Pervas. Mobile Comput.*, vol. 52, pp. 71–99, Jan. 2019, doi: 10.1016/j.pmcj.2018.12.007.

[5] 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.

[6] M. Al-Khafajiy, T. Baker, H. Al-Libawy, Z. Maamar, M. Aloqaily, and Y. Jararweh, "Improving fog computing performance via fog-2-fog collaboration," *Future Gener. Comput. Syst.*, vol. 100, pp. 266–280, Nov. 2019.

[7] M. H. U. Rehman, A. Batool, and K. Salah, "The rise of proximal mobile edge servers," *IT Prof.*, vol. 21, no. 3, pp. 26–32, May 2019, doi: 10.1109/MITP.2019.2898185.

[8] Z. Ning, J. Huang, and X. Wang, "Vehicular fog computing: Enabling real-time traffic management for smart cities," *IEEE Wireless Commun.*, vol. 26, no. 1, pp. 87–93, Feb. 2019, doi: 10.1109/mwc.2019.1700441.

[9] Z. Xiong, S. Feng, W. Wang, D. Niyato, P. Wang, and Z. Han, "Cloud/fog computing resource management and pricing for blockchain networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4585–4600, Jun. 2019.

[10] Y. Jiao, P. Wang, D. Niyato, and K. Suankaewmanee, "Auction mechanisms in cloud/fog computing resource allocation for public blockchain networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 9, pp. 1975–1989, Sep. 2019.

[11] D. Bermbach, S. Maghsudi, J. Hasenburg, and T. Pfandzelter, "Towards auction-based function placement in serverless fog platforms," 2019, *arXiv:1912.06096*. [Online]. Available: http://arxiv.org/abs/1912.06096

[12] M. Liwang, S. Dai, Z. Gao, Y. Tang, and H. Dai, "A truthful reverse-auction mechanism for computation offloading in cloud-enabled vehicular network," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4214–4227, Jun. 2019, doi: 10.1109/jiot.2018.2875507.

[13] 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.

[14] S. Yakut, Ö. eker, E. Batur, and G. Dalkılıç, "Blockchain platform for Internet of Things," in *Proc. Innov. Intell. Syst. Appl. Conf. (ASYU)*, Oct. 2019, pp. 1–6, doi: 10.1109/asyu48272.2019.8946403.

[15] 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.

[16] Q. Zhou, H. Huang, Z. Zheng, and J. Bian, "Solutions to scalability of blockchain: A survey," *IEEE Access*, vol. 8, pp. 16440–16455, 2020, doi: 10.1109/ACCESS.2020.2967218.

[17] S. Wan, M. Li, G. Liu, and C. Wang, "Recent advances in consensus protocols for blockchain: A survey," *Wireless Netw.*, pp. 1–15, Nov. 2019, doi: 10.1007/s11276-019-02195-0.

[18] S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han, and F.-Y. Wang, "Blockchain-enabled smart contracts: Architecture, applications, and future trends," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 49, no. 11, pp. 2266–2277, Nov. 2019.

[19] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2016, pp. 254–269.

[20] H. Wang, H. Qin, M. Zhao, X. Wei, H. Shen, and W. Susilo, "Blockchain-based fair payment smart contract for public cloud storage auditing," *Inf. Sci.*, vol. 519, pp. 348–362, May 2020, doi: 10.1016/j.ins.2020.01.051.

[21] D. Kumar, G. Baranwal, Z. Raza, and D. P. Vidyarthi, "Fair mechanisms for combinatorial reverse auction-based cloud market," in *Information and Communication Technology for Intelligent Systems*. Singapore: Springer, 2018, pp. 267–277.

[22] X. Peng, K. Ota, and M. Dong, "Multiattribute-based double auction toward resource allocation in vehicular fog computing," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3094–3103, Apr. 2020, doi: 10.1109/JIOT.2020.2965009.

[23] R. Besharati and M. H. Rezvani, "A prototype auction-based mechanism for computation offloading in fog-cloud environments," in *Proc. 5th Conf. Knowl. Based Eng. Innov. (KBEI)*, Feb. 2019, pp. 542–547.

[24] L. Fawcett, M. Broadbent, and N. Race, "Combinatorial auction-based resource allocation in the fog," in *Proc. 5th Eur. Workshop Softw.-Defined Netw. (EWSDN)*, Oct. 2016, pp. 62–67.
[25] F. Dewanta and M. Mambo, "Bidding price-based transaction: Trust establishment for vehicular fog computing service in rural area," in *Proc. IEEE Int. Conf. Pervas. Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2019, pp. 882–887.
[26] A. Tasiopoulos, O. Ascigil, I. Psaras, S. Toumpis, and G. Pavlou, "FogSpot: Spot pricing for application provisioning in edge/fog computing," *IEEE Trans. Services Comput.*, early access, Jan. 24, 2019, doi: 10.1109/TSC.2019.2895037.
[27] J. Wang, A. Liu, T. Yan, and Z. Zeng, "A resource allocation model based on double-sided combinational auctions for transparent computing," *Peer-Peer Netw. Appl.*, vol. 11, no. 4, pp. 679–696, Jul. 2018.
[28] Y. Chen, Z. Li, B. Yang, K. Nai, and K. Li, "A Stackelberg game approach to multiple resources allocation and pricing in mobile edge computing," *Future Gener. Comput. Syst.*, vol. 108, pp. 273–287, Jul. 2020, doi: 10.1016/j.future.2020.02.045.
[29] J. Chen. (2019). Reverse Auction. Investopedia. Accessed: Jan. 20, 2020. [Online]. Available: https://www.investopedia.com/terms/r/reverse-auction.asp
[30] M. F. Franco, E. J. Scheid, L. Z. Granville, and B. Stiller, "BRAIN: Blockchain-based reverse auction for infrastructure supply in virtual network functions-as-a-service," in *Proc. IFIP Netw. Conf. (IFIP Netw.)*, May 2019, pp. 1–9, doi: 10.23919/ifipnetworking.2019.8816843.
[31] N. Afraz and M. Ruffini, "A distributed bilateral resource market mechanism for future telecommunications networks," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Waikoloa, HI, USA, Dec. 2019, pp. 9–13.
[32] M. Debe, K. Salah, M. H. U. Rehman, and D. Svetinovic, "Monetization of services provided by public fog nodes using blockchain and smart contracts," *IEEE Access*, vol. 8, pp. 20118–20128, 2020, doi: 10.1109/access.2020.2968573.
[33] Z. Zheng, S. Xie, H.-N. Dai, W. Chen, X. Chen, J. Weng, and M. Imran, "An overview on smart contracts: Challenges, advances and platforms," *Future Gener. Comput. Syst.*, vol. 105, pp. 475–491, Apr. 2020.
[34] M. Al-Khafajiy, T. Baker, A. Waraich, D. Al-Jumeily, and A. Hussain, "IoT-fog optimal workload via fog offloading," in *Proc. IEEE/ACM Int. Conf. Utility Cloud Comput. Companion (UCC Companion)*, Dec. 2018, pp. 359–364.
[35] Ethereum. *Eth Gas Station*. Accessed: Jan. 29, 2020. [Online]. Available: https://ethgasstation.info/
[36] T.ENS. (2019). *Ethereumnameservice*. Accessed: Feb. 20, 2019. [Online]. Available: https://ens.domains/
[37] I. Grishchenko, M. Maffei, and C. Schneidewind, "A semantic framework for the security analysis of Ethereum smart contracts," in *Proc. Int. Conf. Princ. Secur. Trust*. Cham, Switzerland: Springer, 2018, pp. 243–269.

MAZIN DEBE received the B.S. degree in computer engineering from the Khalifa University of Science and Technology, Abu Dhabi, UAE, where he is currently pursuing the degree with the Department of Electrical Engineering and Computer Science. He is also associated with the researchers of the Center for Cyber-Physical Systems, Khalifa University of Science and Technology. He has published three research articles in highly ranked IEEE conferences and journals. 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, 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 joined the Khalifa University of Science and Technology, UAE, 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 the Khalifa University of Science and Technology, he worked at the Department of Information and Computer Science, King Fahd University of Petroleum and Minerals (KFUPM), Saudi Arabia, for ten years. He is currently a Full Professor with the Department of Electrical and Computer Engineering, Khalifa University of Science and Technology. He has over 190 publications, holds three patents, and has been giving a number of international keynote speeches, invited talks, tutorials, and research seminars in 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 was the Track Chair of the IEEE GLOBECOM 2018 on cloud computing. He serves on the Editorial Board of many WoS-listed journals, including *IET Communications*, *IET Networks*, JNCA (Elsevier), SCN (Wiley), IJNM (Wiley), J.UCS, and AJSE. He is an Associate Editor of the IEEE BLOCKCHAIN NEWSLETTER.

MUHAMMAD HABIB UR REHMAN (Member, IEEE) received the bachelor's and master's degrees 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 of Science and Technology, UAE, as a Postdoctoral Research Fellow. He has been an Alumnus of DAAD's Postdoctoral network, since September 2019. He is currently working on trustworthy blockchain technologies for intelligent cyber-physical systems. Overall, he has authored or coauthored 40 international publications, including journal articles, conference proceedings, book chapters, and magazine articles, whereby his four articles are categorized as highly cited publications by the Web of Science. His research interests include blockchain technologies, cyber-physical systems, secure key management, big data, edge computing, the industrial IoT, and the research and development of trust models for decentralized and trustworthy artificial intelligence applications for cyber-physical systems. He is a bright spark fellow. He received gold medals and 100% fee-waiver scholarships from COMSATS University Islamabad.

DAVOR SVETINOVIC (Senior Member, IEEE) received the Ph.D. degree in computer science from the University of Waterloo, Canada, in 2006. He is currently an Associate Professor of computer science with the Khalifa University of Science and Technology, UAE. Previously, he worked as a Visiting Professor and a Research Affiliate 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 working experience on complex multidisciplinary research projects. He has published over 80 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 the ACM.