A Blockchain-Encryption-Based Approach to Protect Fog Federations from Rogue Nodes

Mohammed Alshehri, Brajendra Panda

Department of Computer Science and Computer Engineering University of Arkansas Fayetteville, USA msalsheh,bpanda@uark.edu

Abstract—People have used cloud computing approach to store their data remotely. As auspicious as this approach is, it brings forth many challenges: from data security to time latency issues with data computation as well as delivery to end users. Fog computing has emerged as an extension for cloud computing to bring data processing and storage close to end-users; however, it minimizes the time latency issue but still suffers from data security challenges. For instance, when a fog node providing services to end users is compromised, the users' data security can be violated. Thus, this paper proposes a secure and fine-grained data access control scheme by integrating the Ciphertext Policy Attribute-Based Encryption (CP-ABE) algorithm and blockchain concept to prevent fog nodes from violating end users' data security in a situation where a compromised fog node is being ousted. We also classify the fog nodes into fog federations, based on their attributes such as services and locations, to minimize the time latency and communication overhead between fog nodes and cloud server. Further, the exploitation and integration of the blockchain concept and the CP-ABE algorithm enables fog nodes in the same fog federation to perform the authorization process in a distributed manner. In addition, to solve time latency and communication overhead problems, we equip every fog node with an off-chain database to store most frequently accessed data files for specific time, and with an on-chain access control policies table (On-chain Files Tracking Table) which must be protected from being tampered by malicious (rogue) fog nodes. Therefore, blockchain plays a vital role here as it is tamper-proof by nature. We demonstrate our scheme's efficiency and feasibility by designing algorithms and conducting a security analysis. The provided analysis shows that the proposed scheme is efficient and feasible in ousting malicious (rogue) fog nodes.

Index Terms—Fog Computing, blockchain, , rogue node, finegrained access control

I. INTRODUCTION

Cloud computing is a thriving paradigm due to the enormous on-demand services it provides to end users over the internet. Cloud computing has provided customers with innovative features such as availability, scalability, and economy that help to satisfy the substantial demand for storage and computation resources [10]. End users outsource their data, to the core network on the cloud, for processing and storage. However, there are many obstacles facing data owners. First, the response time between users and the cloud is high because the data is stored in far from the data owners. Second, endusers' data security and privacy are susceptible to violation because of the semi-trusted third party controls the cloud. The research community has studied the issues of data security and privacy in cloud computing by adopting and applying advanced cryptographic techniques. However, it is still demanding to invent a new technology to resolve the cloud latency issue [9][16].

From here, fog computing was introduced in 2012 [6] to reduce the time latency between the cloud and end users' devices [9][16] and to provide new services and applications to end users. Fog computing paradigm is as small clouds close to end-user's devices, which allows customers to utilize the computing and storage services at the edge of the network [6]. More specifically, fog computing comprises of multiple fog nodes which provide services to end users [27]. The fog computing paradigm has supported end devices with distinguishing features such as mobility, location awareness, and low time latency [23]. Because fog computing is deemed as an extension of the cloud computing paradigm; therefore, it inherits some of the security and privacy obstacles in cloud computing[24]. In particular, a fog computing paradigm is susceptible to different threats, such as malicious attacks and technical issues. In this context, the end users' data traveling through fog computing nodes to the cloud is vulnerable to different violations. Therefore, it is necessary to design a scheme to protect end-users' data confidentiality, availability, and integrity by ousting the rogue (malicious) fog nodes. One reason for these vulnerabilities is that end users outsource sensitive data to the nearby fog node for processing and then to the cloud for further processing and storage. Because this fog node is seen as a connector between end users and the cloud, it is challenging to protect other fog nodes and the cloud from malicious attacks. End users' data security would be difficult to defend if the fog node providing services in terms of storage and computation is compromised or goes rogue. We have been motivated by those issues to protect customers' private data from being compromised. Also, a method to secure communication among fog nodes is required to exchange encrypted data for reducing the time latency to retrieve it from the far cloud.

In general, threats in the context of fog computing takes two forms. 1) data modification: if an adversary gets hands-on end users' private data, he/she might violate its confidentiality and integrity. Therefore, introducing a security scheme is necessary to prevent data confidentiality and integrity violation among fog nodes; and between fog nodes and the cloud. 2) unauthorized access: when an adversary compromises a fog node, he/she can get unauthorized access to the end users' private data. Besides, data availability is in danger of malicious violation in this way. Therefore, it is essential to introduce a security scheme to protect against rogue (malicious) fog nodes and to oust them off the network while maintaining low time latency feature fog computing provides. Additionally, it is vital to enabling the fog nodes to carry on a distributed authorization process and to communicate in a trust-less environment.

Attribute-Based Encryption (ABE) is a cryptographic primitive that enables one-to-many encryption. It comprises two types: ciphertext policy attribute-based encryption (CP-ABE) [3] and key policy attribute-based encryption (KP-ABE) [11]. CP-ABE was introduced by Bethencourt at el. [3] which is deemed as one of the most significant algorithms to provide fine-grained data access control.

Blockchain has attracted the attention of researchers since it was introduced in late 2008 [19]. Blockchain is a shared distributed ledger, which is secured, immutable, and transparent, that helps in processing and tracking resources without depending on a centralized trusted third party [26]. With this promising technology, peer-to-peer nodes can communicate, and exchange resources where the decision is carried on in a distributed manner by the majority of the network's nodes rather than a single centralized trusted third party.

This paper proposes a scheme to oust rogue (malicious) fog nodes and to minimize the communication overhead between the cloud service provider (CSP) and fog nodes (FNs) by integrating the CP-ABE algorithm and blockchain technology. Blockchain is adopted as a medium to store on-chain tracking table to verify the identity of each fog node using the smart contract before they could access the encrypted data on the CSP. The blockchain immutability feature prevents fog nodes from maliciously changing the on-chain tracking table, if such a change detected, the FN which issues the request is reported as a rogue fog node.

The rest of this paper is organized as: Section (II) presents the related work. Section (III) focuses on the motivation of this paper. Section (IV) explains the proposed scheme, and the scheme description is broadly explained in section (V). Finally, the security analysis and the conclusion consecutively are in sections (VI) and (VII).

II. RELATED WORK

The main objectives of this paper are to propose a new scheme to allow fog nodes (FNs) to communicate in trustless environment, to maintain end users' data security, and to reduce the time latency and communication overhead between the cloud service provider (CSP) and the FNs by adopting and integrating the CP-ABE and blockchain. Therefore, we summarize the overview of any related state-of-the-art works in this section, a brief literature review about access control, rogue fog nodes, and fog nodes communication.

A. Access Control

Sahai et al. [20] proposed an attributes-based encryption (ABE) scheme that uses the identity-based encryption algorithm [5]. Its two variants are the key-policy ABE (KP-ABE) and the ciphertext-policy ABE (CP-ABE) [11][3]. ABE encryption and decryption processes are based on users' attributes, so the scheme was first adopted in cloud computing to help data owners overcome obstacles such as data security and data access control when outsourcing data to the cloud.

The concept of access control has been well researched in cloud-computing, but more investigations are still needed into fog computing. Recently, the research community has studied access control issues in this area and started to adopt ABE in the environment with the object of providing fine-grained data access control and guaranteeing data security.

ABE has been applied to IoT devices to resolve data access control issues. Yu et al. [27] presented the fine-grained data access control issues arising in a wireless server network, proposing an FDAC scheme in which each sensor node was assigned a set of attributes and each user was assigned an access structure to specify access rights.

Huang et al. [14] proposed a CP-ABE scheme to support data security and fine-grained data access control, using features such as ciphertext updating and computation outsourcing for fog computing with IoT devices. As Zuo et al. [28] also found, their scheme's main problem is that it was only suitable with an AND-gate encryption predicate. Xiao et al.[25] proposed a hybrid fine-grained search and access authorization scheme for fog computing, based on a CP-ABE cryptographic primitive. Their scheme was hybrid as it supported both index encryption and data encryption. This too, however, supported only AND-gate predicates. Mao et al. [17] proposed the construction of a CPA-secure and RCCA-secure scheme; it was based on an ABE model with the possibility of outsourced encryption verification. Dsouza et al. [21]proposed a scheme to support secure data sharing and communication in fog computing. This scheme is no more than a policy management framework as it lacks details on building the policy repository and users' identities, as well as on making decisions and protecting users' identities and data privacy. Stojmenovic et al. [23]studied authorization and authentication issues among fog devices and between fog and cloud. They based their study on the ABE algorithm and argued that end users could still be authenticated and authorized to fog devices even in the presence of a vulnerable connection between the fog and the cloud. Li et al.[15] proposed a model to collect smart devices' attributes as dynamic attributes, incorporating them with the ABE algorithm to verify the access authority in real time. Mollah et al.[18] proposed a lightweight cryptographic model to provide access control and data-sharing; in this model, all security operations are offloaded to nearby fog servers.

B. Rogue Fog Nodes

A rogue fog node is a fog node that pretends to be legitimate and deceiving end devices, other fog nodes, or the cloud into communicating with it. Its malicious hidden intent is to violate owners' data security and privacy. Having compromised a fog node, an attacker can violate the security and privacy of end users' data in transit to the cloud through the fog nodes. The research community has not broadly addressed this problem.

Stojmenovic et al. [22] proposed a scheme to show the feasibility of a man-in-the-middle attack in which the gateway is either being compromised or substituted by a fake. Han et al. [12][12] proposed a measurement-based scheme to help users avoid connecting to fake access points. Their scheme discovers the rogue AP by calculating the round-trip time between end users and the DNS server. Mohammed et al.[2] proposed an encryption-based approach to protect fog computing from rogue fog nodes based on the CP-ABE algorithm. In this scheme, fog nodes cannot exchange data unless they fully trust each other. So, to let the fog nodes to carry on a distributed authorization process is an outstanding idea.

C. Communication among Fog Nodes

Arwa et al. [1] proposed an attribute-based encryption scheme to maintain fog communications security. Designed to provide authentic, confidential communications among a group of fog nodes, it made use of a key-exchange protocol based on the CP-ABE algorithm. The researchers have not widely addressed this issue.

Recently, researchers have started using blockchain to develop secure applications, benefiting customers in various fields. Christidis et al. [8] summarized the blockchain use cases built into the IoT. They reviewed the integration of a blockchain smart contract into the IoT. Kamanashis et al.[4] proposed a multi-layered security scheme to create a secure communication platform in smart cities by integrating blockchain with smart devices. Researchers use blockchain as a distributed database to store heterogeneous data related to the smart city, such as traffic and location. These data need to be shared among smart cities' components. The main issues dealt with by this scheme are scalability and reliability in smart cities. In[13], authors proposed a blockchain-based scheme, consisting of distributed data storage, to enable data sharing in IoT environments. The aim of using blockchain was to enable data access control and data storage. In their scheme, the authors separated data store from data management, then used blockchain to verify the separation and to decentralize the access control.

The fog computing layer occupies the middle ground between cloud servers and end users' devices and is, therefore, more susceptible to attack than other layers. Data owners can find their data damaged or leaked by a rogue fog node. Since one of the primary fog computing features is to increase network reliability, fog nodes need protection method against malicious attacks. Protecting the nodes will defend the data owners' security and privacy.

D. Our Contributions

To the best of our knowledge, no previous work has adopted and integrated the blockchain technology with the CP-ABE algorithm to form a fog federation (FF) and address the problem of rogue fog nodes in fog computing. Based on [26] and [3], we propose a secure scheme in the context of fog computing with the following features: 1) to detect and prevent rogue fog nodes from violating end users' data confidentiality and besides to minimize the communication overhead between the cloud server and the fog nodes. 2) to enable the fog nodes to communicate in a trust-less environment. 3) to enable the fog nodes to perform the authorization process in a distributed manner. 4) it supports end users' data availability. However, we focus our work on the communication among fog nodes and between the end users' devices and the fog nodes is secured for now.

III. MOTIVATION

Fog computing plays critical roles in providing multiple services to end users such as music, adults' entertainment, kids' entertainment, emergency vehicle response, and health data collection at low latency time [6]. Even the innovative features fog computing paradigm provides, it introduces many security issues. However, customers cannot benefit from those services fog computing provides if the fog node, which provides the intended services, is not functioning thoughtfully. Therefore, costumers care about their data security and availability in different geographic location. This scheme is motivated by the idea of protecting customers' data security from rogue fog nodes. For example, in the health care sector, health care provider outsources patients' sensitive data to the close FN for processing and then for uploading to the cloud for permeant storage. In the future, suppose the FN responsible for these data is compromised, this FN could breach end users' data to attackers. Moreover, end users would not be able to access his/her original data as the FN in charge of providing this service is comprised, so users need to be forwarded to another trusted fog node for data retrieving from the CSP. This paper focuses on preventing the security threats that rogue fog nodes cause the system to breach end users' encrypted data and the time latency to delivering the data from the cloud to the fog node. Moreover, we leverage the blockchain technology to enable fog nodes to provide authorization in a distributed manner. So, we argue that this would help in recognizing rogue fog nodes. Specifically, an adversary can either take over the fog node or intercept data flown from the fog nodes to the cloud and vice versa. Based on the literature review, no previous scheme has addressed the issue of protecting the fog nodes from rogue nodes by combining a cryptographic primitives, such as CP-ABE, with blockchain. Hence, there is a need for an efficient and secure scheme that considers these security challenges and time latency.

IV. PROPOSED SCHEME

As Fig. 1 illustrates, our model comprises of the following entities: a cloud service provider (CSP), fog nodes (FNs), and a Cryptographic Materials Issuer(CMI). Fog nodes (FNs) with same attributes form a fog federation (FF) which is considered as a private blockchain. FNs in the FF function



Fig. 1. The system model.

as miners to validate the requests generated by FN in same FF. Through this paper, we use fog federation and private blockchain interchangeably.

Data delivery from cloud storage to end users' devices is hindered by time latency and communication overhead. Fog computing has emerged to address these issues. However, FNs are exposed to malicious attacks; consequently, the end users' data security may be violated. To maintain confidentiality, integrity and availability for end users' data travelling through FNs to the CSP, we exploit and integrate blockchain with the CP-ABE algorithm, an advanced cryptographic primitive that enforces fine-grained data access control to secure communication between FNs and the CSP. Using blockchain, FNs can perform the authorization process in a distributed manner.

To solve the problem of high time latency and communication overhead, we equip every FN with an access control policies table (on-chain files tracking table), as shown in Table-IV, which must be protected from tampering conducted by malicious (rogue) FNs. Blockchain plays a vital role here, as it is tamper-proof by nature. This scheme equips each FN with a set of attributes, namely location and services. Each FN in the same FF maintains an on-chain files tracking table to provide fine-grained access control by using a smart contract. This allows other FNs in the same FF to access the encrypted data on the CSP when their attributes satisfy the stored predicates in the on-chain files' tracking table. When a FN sends a request to other FNs in the same FF to access a certain file, they check whether any FN in the FF has the required file stored in its off-chain database (DB). If so, they refer the requestor to get the file from the FN currently in possession of the file instead of sending the request to the CSP. This feature is a benefit of our scheme, as it minimizes the time latency and communication overhead between the CSP and the FF. In addition, the CSP maintains a verification list (VL) of FFs along with their Fog Federation Public Key (FFPK) and Fog Federation Attributes (FF_{att}) , as shown in Table-I.This table allows the CSP to verify the received request from FNs. It also stores each encrypted file (EF) along with a verification file (VF), as shown in Table-II. The CSP uses this table to match each EF to the corresponding VF in order to verify the requests arriving from the FNs.

In fog computing, FNs collect data from end devices. They process those data and upload the results to the cloud for further processing and storage. We equip every FN with two databases (DBs), an off-chain and an on-chain DB. The offchain DB stores the encrypted data most frequently accessed by end users, which reduces time latency when retrieving the encrypted data from the CSP. FN can retrieve data encrypted by FNs in the same FF. This feature helps maintaining data availability when ousting an FN from the FF near end users' devices. The on-chain database is considered as digital ledger and stored on the blockchain. It stores an on-chain file tracking table as shown in Table-IV to verify FF's members identifications. To protect cryptographic shares from breaching by malicious FN, it is stored off-chain by every FN, Table-III. For the sake of clarity, we will consider the following example:

Suppose FN_1 encrypts file F_1 using a random secret key (SK). Then it generates a VF. Consequently, it uploads the EF into the CSP along with the VF and generates a new row in the on-chain file tracking table.

FN₁ divides the SK and the VF into n shares where n is the number of FNs in a given FF (private blockchain). It sends $[SK_{share}, VF_{share}, EF_{ID}]$ to each FN in the FF to be stored as off-chain cryptographic shares, Table-III. All FNs in the same FF maintain an on-chain file tracking table, as shown in Table-IV. This table is used to verify the data requestor (FN_{ID}) by verifying the FN_{sign-att}. This table is protected from being tampered by malicious FNs through the nature of blockchain services. Suppose FN₂ in the FF seeks to retrieve an EF from the CSP. There are three scenarios, which are as follows.

In the first scenario, FN₂ needs to retrieve EF₄ from the

TABLE I VERIFICATION LIST

FF _{ID}	FFPK	\mathbf{FF}_{att}
FF_1	Rtufn@10	(Health OR Education) and Atlanta
FF_n	1039gNF	Education and Atlanta

TABLE II ENCRYPTED FILES AND VERIFICATION FILES

EF _{ID}	\mathbf{VF}_{ID}
F_1	VF_1
F _n	VF _n

CSP, and no FN has the file in its off-chain DB. FN(requestor) needs to send a request through the FF (private blockchain). Then, the FNs in the same FF would verify the requestor's attributes signature. If it is verified by the majority of the FNs in the FF, they would share the [SK_{share} , VF_{share} , EF_{ID}] they have related with the EF_4 with the requestor. When the requestor collects at least 51% of the VF's shares, it sends a request to retrieve the EF_4 from the CSP as [VF_{shares} , EF_4 ,signed (FN_{att})]. Accordingly, the CSP verifies the VF_{shares} ; if they match the VF attached with the EF_4 , it sends the file to the FN to decrypt. For more details, read the scheme's description.

In the second scenario, FN_2 needs to retrieve EF_2 from the CSP, and FN_1 has the file in its off-chain DB. It sends a request through the private blockchain with the $[EF_2, signed (FN_{att})]$. Accordingly, FNs verify the FN_2 's identity by verifying its signed attributes using FFPK. If verified, they check which of the off-chain DBs has the file and then send the FNs_{ID} that have the EF_2 to the requestor. In this case, the file requestor, FN_2 , does not have to contact the far CSP to retrieve the file. It could contact FN_1 to get EF_2 . This feature gives our scheme a credit in minimizing the time latency and communication overhead between the CSP and the FFs. This is possible with the fog federation idea. For more details, read the scheme's description.

In the third scenario, if FN (the requestor) is not verified, FNs in the FF will report the FN as a rogue FN and delete it from the on-chain tracking table. It then cannot access any encrypted data.

An FN leaves the system for whatsoever reason, or it could go rogue after being compromised by a malicious attack. To ensure users' data security, namely data confidentiality, and availability, there has been a demand to take precautions against rogue FNs. Our scheme efficiently acts as a protection layer against rogue FNs to prevent them from accessing the encrypted data stored on the CSP.

All FNs in one FF maintain the same on-chain file tracking table. Malicious modification of the table by a malicious FN would be detected, as any update must be verified by the majority of the FNs, which is why we utilize the blockchain as FFs. If an FN issues an update and it is not passed by most of the FNs, the issuer would be considered as a rogue FN and then excluded from the FF. When a rogue FN is detected in the FF, it is reported to the CSP. Then, the CSP will not respond to any requests from this FN. Accordingly, this feature enhances the scheme's ability to protect against rogue FNs.

TABLE III		
CRYPTOGRAPHIC SHARES		

EF _{ID}	VF _{share}	SK _{share}
EF ₁	MNGONMDF	SK_1
EF_n	MNGONMDF	SK _n

TABLE IV ON-CHAIN TRACKING TABLE

FN _{ID}	\mathbf{EF}_{ID}	$FN_{sign-att}$	attributes set	off-chain DB
FN ₁	EF_1	fNJk3u@	Movies and CL	EF_2
FN ₂	EF ₃	KLJk3J@	Edu and AT	0
		•••		
FN_n	EF ₇	@3EFJJL@	Health and AR	0

Algorithm 1 Setup(λ)

6:

- 1: Choose bilinear cyclic group G_1 of prime order p with generator g_1 ;
- 2: Randomly generates exponents $\alpha, \beta \in G_p^*$;
- Random Oracle H: USAT → {0,1}* // to map each att in USAT into random element ∈ G₁;
- 4: compute SMK and SPK as:
- 5: $SMK = \{\beta, g_1^{\alpha}\}$
 - $SPK = \{G_1, g_1, g_1^{\beta}, e(g_1, g_1)^{\alpha}\};$
- 7: The SPK is public to all FNs, but SMK is private for CMI

V. SCHEME DESCRIPTION

In this section, we present the description of our scheme based on the integration of the CP-ABE algorithm and blockchain. We exploit the access tree model presented in [3] as an access structure A, as shown in Fig-2. To satisfy the access tree conditions, we follow the same methods in [3][5]. For more information about the access tree model, read through [3][5]. Next we provide algorithms needed for our model.

Algorithm 1, which is based on [3], is executed by CMI. It receives a security parameter as an input, then it outputs a system master key (SMK) and a system public key (SPK). It builds a universal set of attributes (USAT), which FF classification is based on in our scheme; USAT = FN's location, FN's services.

Algorithm-2 is executed by an FN that intends to join the system to provide a service for end users. It contacts the CMI, then the CMI checks the FFL. Then, it assigns the FN to the corresponding FF if its attributes are verified. If the FN is verified but no FF_{att} matches its attributes, the CMI creates a



Fig. 2. Example of an access structure.

Algorithm 2 Join (FN_{att})

- 1: New FN contacts CMI to join the system
- 2: CMI verifies the FN (requestor)
- 3: if not verified then
- 4: Declined
- 5: else
- 6: Check the FFL
- 7: **if** $FF_{att} = new FN_{att}$ **then**
- 8: Assign new-FN_{att} to the corresponding FF_{ID}
- 9: **end if**
- 10: Assign new FN_{ID} as: FF_{ID} . FN_{ID}
- 11: Send new FN_{ID} to all FN in the same FF
- 12: end if
- 13: for every $FN \in FF$ do
- 14: share on-chain ledgers (on-chain tracking table) with the new FN
- 15: **end for**
- 16: if New FN receives at most 49% of un-identical on-chain identical ledgers then
- 17: report a problem to CMI
- 18: else
- 19: Return (on-chain ledgers are authenticated)
- 20: end if

Algorithm 3 Generating FNSK (SPK, SMK, S, FN_{ID})

- 1: CMI gets request from new FN to join the system
- 2: Choose bilinear cyclic group G_1 of prime order p with generator g_1 ;
- 3: Randomly generates exponents $\alpha_1, \beta_1 \in G_p^*$;
- 4: Random Oracle H: USAT → {0,1}* // to map each att in USAT into random element ∈ G₁;

5: compute FFPK and FFMK as:

6: $FFPK = \{\beta_1, g_1^{\alpha_1}\}$ 7: $FFMK = \{G_1, q_1, g_1^{\beta_1}\}$

$$FFMK = \{G_1, g_1, g_1^{\beta_1}, e(g_1, g_1)^{\alpha_1}\}$$

- 8: generate random $r \in Z_p^*$
- 9: for every $att_i \in S$ do
- 10: generate random $r_i \in Z_P^*$
- 11: **end for**
- 12: for each new $FN_i \in FF$ do
- 13: compute FNSK
- 14: send FNSK to FN_i
- 15: **end for**

new FF and adds the new FN to it. If the FN is not verified, the connection will be refused. However, as each FF is considered a private blockchain, the FNs already part of the FF have to share their on-chain ledger with the new FN. If the new FN receives 51% of identical ledgers, it saves them in the on-chain DB. Otherwise, it reports an issue to the CMI. The details of how to verify and authenticate the FN's attributes are out of our scope for now.

Algorithm 3 is executed by the CMI. It generates a Fog Federation Public Key (FFPK) and a Fog Federation Master Key (FFMK). The FFPK is public to all FNs in the FF, whereas the FFMK is private to the CMI. It also generates

Algorithm 4 Data Encryption (M, T, SPK, FFPK)

- 1: SK = gen.DEC-Key
- 2: verification-file = generate.VF
- 3: $VF_{share} = VF/n // n$ is the number of FNs in FF
- 4: divide SK key to shares as: SK/n
- 5: Encrypted-File (EF) = Encrypt (M, SK) //M is the message to encrypt
- 6: A is set of atts represented by monotone access structure tree T
- 7: for each node x in T do
- 8: set a polynomial
- 9: **if** node x is root node in T **then**
- 10: set $q_{(0)R} = s // s$ is random value $\in Z_P^*$
- 11: **else**
- 12: $q(0)_x = q_{parent(x)}(index(x))$
- 13: then choose $d_x = k_x 1$ as polynomial degree to interpolate with q_x
- 14: end if
- 15: end for
- 16: for each $y \in Y$ do //Y is the set of leaf nodes in T
- 17: $C_y = g^{q_y(0)}$

18:
$$C'_{y} = H(att(y))^{q_{y(0)}}$$

- 19: end for
- 20: Compute: $C = g^{\beta s}$
- 21: for each $SK_i \in SK$ do
- 22: Compute $C_{ver_i} = SK_i \cdot e(g, g)^{\alpha s}$
- 23: **end for**
- 24: Upload [EF, EF_{ID} , VF, FN_{att_sig}] to CSP
- 25: for each $FN \in FF$ do
- 26: send [EF_{ID} , VF_i , C_{ver_i}]

27: end for

a distinct Fog Node Secret Key (FNSK) for each FN in the FF. Then, the CMI securely shares the FFPK and FNSK with the new FN.

Algorithm 4 is executed by authorized FNs in a given FF. The FN intending to encrypt a file generates a secret key (SK) and uses it to encrypt the file. It also generates a verification file (VF) which is used for further requestors verification by the CSP. Then, it divides the SK and the VF into n shares as

Algorithm 5 on-chain track	ng table smart contract
----------------------------	-------------------------

- 1: sig = sign(FN_{att}, FNSK) // for future authentication
- 2: propagate the transaction through FF [EF_{1D}, Current-Hash, Prev.Hash, Timestamp, FN_{att-sig}]
- 3: if transaction is verified and $FN_{att.sig.verify}$ is true then
- File_{owner} (FN) generates a new row in on-chain file tracking table as:[FN_{ID}, EF_{ID}, attributes-set, off-chain DB]
- 5: FNs \in FF update the on-chain tracking table

6: **else**

- 7: decline and Report FN as rogue FN to all FNs in FF by its ID
- 8: **end if**

Algorithm 6 File Retrieving Smart Contract

1: Part 1: FN (requestor) ID verification 2: FN send a request through FF as $[FN_{ID}, EF_{ID}]$ FN_{att-sig}] 3: for every $FN_i \in FF$ do if $FN_{att-sig} ==$ true then 4: send $[SK_{share}, VF_{share}, EF_{ID}]$ 5: else 6: 7: decline end if 8: 9: end for 10: if FN (requestor) recieves 51% of (SK,VF) shares then send (SK,VF) shares to CSP 11: 12: end if Part 2: CSP checks the VF_{shares} 13: 14: if VF_{shares} match at least 51% of VF then send EF to the FN (requestor) 15: 16: else decline the request and report FN to CMI as rogue node 17: 18: end if

n is the number of FNs in the FF. After that, it encrypts the $SK_{i,shares}$ using the CP-ABE algorithm to enable every FN in the same FF to decrypt the encrypted_{SK}. Finally, the data owner (FN) uploads the EF along with the VF to the CSP. We assume that the number of FNs in an FF is fixed for now.

Algorithm 5 is an on-chain smart contract which is triggered among authorized FNs in the FF. The basic function of this smart contract is to keep tracking which FNs have which files in the off-chain DB. This means less time latency to retrieve EFs from the CSP and less communication overhead between the CSP and the FFs. No FN can retrieve the EF or perform any operation unless it is authorized by the majority of the FNs in the FF. This is attributed to the blockchain tamperproof feature. When a FN submits a rquest to retrieve an EF from the cloud it has to sign its attributes using its own FNSK, then the others FF's members will authenticate it by checking its signature using the FFPK. This feature is a credit to our scheme, as it prevents FNs in the FF from falsifying information about files' tracking table or accessing the EF in the CSP.

Algorithm 6 is a smart contract which two parts. First part is executed between the FN that intends to retrieve an EF from the CSP and FF's members. Second part is between the FN (requestor) and the CSP. FN first propagates a request to retrieve an EF from the CSP through the private blockchain. All FNs in the FF verify the requestor attributes by verifying the requestor attribute signature. When majority FNs verify the requestor (see algorithm 7), they send $[SK_{share}, VF_{share}]$ to the EF requestor. When the EF's requestor (FN) receives 51% of the shares, it send $[SK_{share}, VF_{share}, EF_{ID}]$ to the CSP. The CSP checks the shares against the pre-stored VF. If the shares match at least 51% of the VF, the CSP sends the EF to the FN. If the shares do not match the VF, the CSP will report the requestor as a rogue FN to the CMI. This feature

Algorithm 7 Consensus approach

- 1: assume 3f + 1 of FNs in FF // f is max number of FNs which may fail
- 2: if 2f +1 of FNs confirm the transaction then
- 3: FNs come to consensus
- 4: else
- 5: discard the transaction
- 6: end if

strengthens our scheme in ousting rogue FNs to protect end users' data confidentiality, availability and to authenticate the FF's members identity.

Algorithm 7 is based on Practical Byzantine fault tolerance approach (PBFT) [7]. We assume that the FF has 3f + 1 of FNs, where f is the maximum number of FNs that could fail. To confirm a specific transaction in our scheme, at least 2f + 1 of FNs must agree on it. In this method, the authorized FNs in the FF come to a consensus state.

VI. SECURITY ANALYSIS

A. Confidentiality

The EFs in the CSP is protected against rogue FNs. To retrieve an EF, the FN must sign its attributes using its FNSK, then send it with the request as a transaction through the private blockchain. The EF's requestor must accumulate at least 51% of $[SK_{shares}, VF_{shares}]$ to be able to retrieve the EF from the CSP. This will not happen unless the EF's requestor passed the authorization process (algorithm 6) through the private blockchain. If the EF's requestor is not authorized by the majority of FF's members, it is reported as a rogue FN and consequently cannot access the EFs in the CSP or in an off-chain DB of other FNs. This feature makes our scheme efficient in protecting end users' data confidentiality.

B. Availability

The proposed scheme guarantees the data availability: even the FN providing services to end-users is down. The idea of the FF enables data owner to retrieve their encrypted data through any FN \in same FF. The most frequently accessed EFs are stored in the off-chain DB of the FN that recently accessed it. This means the end users' data would be available near them for a period of time. The off-chain DB is flushed regularly as precaution against malicious FN.

C. Ousting Rogue FN

This scheme aims to oust rogue FNs from the system to keep end users' data secured from being breached or compromised. This feature is obvious through algorithms (5,6). In particular, if the FN (requestor) does not sign its attributes with the correct FNSK, FN authorization process will be denied by the private blockchain. Therefore, it would be ousted and protected from accessing the EFs in the CSP and in an offchain DB of other FNs in same FF.

VII. CONCLUSION

If the FNs providing services to end users are malfunctioning, end-users' data security can be violated. Also, it is possible that the FNs to go rogue which maliciously would violate end-users' data security. Besides, the time latency and communication overhead between the CSP and FNs are high when every time FN retrieves encrypted data from the CSP for processing. This paper proposed a novel scheme to protect end-users' data security from being breached by rogue FNs and to reduce the time latency and communication overhead between the FNs and the CSP. It also enables FNs to communicate in a trust-less environment, and to perform authorization processes in distributed manner to detect the rogue fog node. The proposed scheme exploited and integrated the CP-ABE algorithm and the blockchain concept. We classified the FNs into FFs (private blockchain) according to their attributes and equipped each FN with on-chain and offchain databases. The data requestor (FN) must sing its attribute using its own FNSK before propagating a request through the private blockchain to access EF stored in the CSP. In case the requestor FNatt-sig is not verified by the majority of the FNs \in FF, the FN would be reported as a rogue FN. We provided a security analysis to show the proposed scheme is secured against rogue FNs and is efficient in reducing time latency and communication overhead between the CSP and the FNs. Our future work will conduct a simulation stage to test our scheme performance.

REFERENCES

- Arwa Alrawais, Abdulrahman Alhothaily, Chunqiang Hu, Xiaoshuang Xing, and Xiuzhen Cheng. An attribute-based encryption scheme to secure fog communications. *IEEE access*, 5:9131–9138, 2017.
- [2] Mohammed Alshehri and Brajendra Panda. An encryption-based approach to protect fog federations from rogue nodes. In *International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage*, pages 225–243. Springer, 2019.
- [3] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In 2007 IEEE symposium on security and privacy (SP'07), pages 321–334. IEEE, 2007.
- [4] Kamanashis Biswas and Vallipuram Muthukkumarasamy. Securing smart cities using blockchain technology. In 2016 IEEE 18th international conference on high performance computing and communications; IEEE 14th international conference on smart city; IEEE 2nd international conference on data science and systems (HPCC/SmartCity/DSS), pages 1392–1393. IEEE, 2016.
- [5] Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In *Annual international cryptology conference*, pages 213–229. Springer, 2001.
- [6] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM, 2012.
- [7] Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance and proactive recovery. ACM Transactions on Computer Systems (TOCS), 20(4):398–461, 2002.
- [8] Konstantinos Christidis and Michael Devetsikiotis. Blockchains and smart contracts for the internet of things. *Ieee Access*, 4:2292–2303, 2016.
- [9] Ruilong Deng, Rongxing Lu, Chengzhe Lai, and Tom H Luan. Towards power consumption-delay tradeoff by workload allocation in cloud-fog computing. In 2015 IEEE International Conference on Communications (ICC), pages 3909–3914. IEEE, 2015.

- [10] Armando Fox, Rean Griffith, Anthony Joseph, Randy Katz, Andrew Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, and Ion Stoica. Above the clouds: A berkeley view of cloud computing. *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep.* UCB/EECS, 28(13):2009, 2009.
- [11] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attributebased encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 89–98. Acm, 2006.
- [12] Hao Han, Bo Sheng, Chiu Chiang Tan, Qun Li, and Sanglu Lu. A measurement based rogue ap detection scheme. In *IEEE INFOCOM* 2009, pages 1593–1601. IEEE, 2009.
- [13] S. H. Hashemi, F. Faghri, P. Rausch, and R. H. Campbell. World of empowered iot users. In 2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI), pages 13–24, April 2016.
- [14] Qinlong Huang, Yixian Yang, and Licheng Wang. Secure data access control with ciphertext update and computation outsourcing in fog computing for internet of things. *IEEE Access*, 5:12941–12950, 2017.
- [15] Fei Li, Yogachandran Rahulamathavan, Mauro Conti, and Muttukrishnan Rajarajan. Robust access control framework for mobile cloud computing network. *Computer Communications*, 68:61–72, 2015.
- [16] Jianhua Li, Jiong Jin, Dong Yuan, Marimuthu Palaniswami, and Klaus Moessner. Ehopes: Data-centered fog platform for smart living. In 2015 International Telecommunication Networks and Applications Conference (ITNAC), pages 308–313. IEEE, 2015.
- [17] X. Mao, J. Lai, Q. Mei, K. Chen, and J. Weng. Generic and efficient constructions of attribute-based encryption with verifiable outsourced decryption. *IEEE Transactions on Dependable and Secure Computing*, 13(5):533–546, Sep. 2016.
- [18] Muhammad Baqer Mollah, Md Abul Kalam Azad, and Athanasios Vasilakos. Secure data sharing and searching at the edge of cloudassisted internet of things. *IEEE Cloud Computing*, 4(1):34–42, 2017.
- [19] Satoshi Nakamoto et al. Bitcoin: A peer-to-peer electronic cash system. 2008.
- [20] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pages 457–473. Springer, 2005.
- [21] Stavros Salonikias, Ioannis Mavridis, and Dimitris Gritzalis. Access control issues in utilizing fog computing for transport infrastructure. In *International Conference on Critical Information Infrastructures Security*, pages 15–26. Springer, 2015.
- [22] Ivan Stojmenovic and Sheng Wen. The fog computing paradigm: Scenarios and security issues. In 2014 Federated Conference on Computer Science and Information Systems, pages 1–8. IEEE, 2014.
- [23] Ivan Stojmenovic, Sheng Wen, Xinyi Huang, and Hao Luan. An overview of fog computing and its security issues. *Concurrency and Computation: Practice and Experience*, 28(10):2991–3005, 2016.
- [24] Hassan Takabi, James BD Joshi, and Gail-Joon Ahn. Security and privacy challenges in cloud computing environments. *IEEE Security* & *Privacy*, 8(6):24–31, 2010.
- [25] Min Xiao, Jing Zhou, Xuejiao Liu, and Mingda Jiang. A hybrid scheme for fine-grained search and access authorization in fog computing environment. *Sensors*, 17(6):1423, 2017.
- [26] Xiwei Xu, Ingo Weber, Mark Staples, Liming Zhu, Jan Bosch, Len Bass, Cesare Pautasso, and Paul Rimba. A taxonomy of blockchain-based systems for architecture design. In 2017 IEEE International Conference on Software Architecture (ICSA), pages 243–252. IEEE, 2017.
- [27] Shanhe Yi, Cheng Li, and Qun Li. A survey of fog computing: concepts, applications and issues. In *Proceedings of the 2015 workshop on mobile big data*, pages 37–42. ACM, 2015.
- [28] Cong Zuo, Jun Shao, Guiyi Wei, Mande Xie, and Min Ji. Cca-secure abe with outsourced decryption for fog computing. *Future Generation Computer Systems*, 78:730–738, 2018.