This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3067284, IEEE Access

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier

A Scalable and Efficient Multi-agent Architecture for Malware Protection in Data Sharing over Mobile Cloud

ZAHID HUSSAIN QAISAR[1], SULTAN H. ALMOTIRI [2] MOHAMMED A. ALGHAMDI [2] ARFAN ALI NAGRA [3] GHULAM ALI [4] .

[1]School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, PR China
[2]Department of Computer Science, UMM Al-Qura University, Makkah City, Saudi Arabia
[3]Department of Computer Science, Lahore Garrison University, Lahore, Pakistan
[4]Department of Computer Science, Okara University, Okara, Pakistan

Corresponding author: Zahid Hussin Qaisar (E-Mail: zahidhussainqasiar@gmail.com).

ABSTRACT Sensitive Data requires encryption before uploading to a public cloud. Access control based on Attribute-Based Encryption (ABE) is an effective technique to ensure data shared security and privacy in the public cloud. Cipher-Text Policy of ABE may suffer from scalability and performance issues as they do not permit for addition or removal of computing nodes at run time. Furthermore, another problem is existing approaches suffer from single-point-of-failure (SPoF). Therefore, we introduce a scalable multi-agent system architecture based on CP-ABE to ensure data sharing on public cloud storage and reliability in our proposed work. We proposed a cloud host as an inter-mediator between the user and the authorized agents without violating the system's privacy and security. We have also proposed a novel methodology to protect the cloud from malware by exploiting the state's efficient power of the art Gemini approach. Gemini is an efficient methodology for binary code-based graph embedding similarity detection. Our proposed study overcomes the deficiencies of scalability and efficiency along with providing the mechanism for malware detection in the cloud. It covers three aspects: scalability, the efficiency with multi-agents, and malware detection capability. Our contributed work is scalable, efficient for cloud data sharing, and protects from malware. Results reveal that our work provides better performance with preserving the security, privacy, and fine granularity features of CP-ABE and malware screening using regress analysis by the graph embedding technique.

INDEX TERMS Scalable Cloud Architecture, Multi-Agent Architecture, Malware Detection for Mobile Cloud, Graph Similarity Based Malware Detection, Graph Embedding Based Analysis.

I. INTRODUCTION

DUE to the inherent benefits of *Mobile Cloud Computing* (MCC)), organizations tend to migrate their data to the public cloud. It enables data can be accessed by their employees anytime, anywhere using their *Mobile Devices* (MDs). By accessing data using smart devices increases productivity [1]. However, moving confidential data to a public cloud and accessing the data using MDs encounter many security and privacy threats due to third-party service providers' involvement and the nature of MDs [2] [3]. Therefore, data must be encrypted before it can be uploaded to public cloud storage.

There are some critical challenges such as data confidentiality, the privacy of the user's identity that must be addressed in order to make effective use of cloud-based services in case of security attacks [4]. Researchers have proposed several strategies based on ABE in the literature to leverage cloud computing benefits while ensuring data privacy and security in the MCC environment. One of the prominent technique is the *Key-Policy ABE* (KP-ABE) [5] [6]. However, in the KP-ABE mechanism, the data owner cannot decide who can decrypt the cipher-text; instead, this control is with the central authority. In quest of tackling this issue, Cipher text-Policy *Ciphertext-Policy ABE* (CP-ABE) [4] has been proposed where the access structure is specified within the cipher-text; therefore, the data owner has the power to specify who can or cannot decrypt the data. In addition, the attribute authorities generate the partial secret keys for users against their static attributes. Unki *et al.*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3067284, IEEE Access

[5] conducted a comparative analysis of KP-ABE and CP-ABE mechanisms. The authors conclude that the CP-ABE mechanism achieves better performance and takes less time to encrypt or decrypt a message and generate keys while achieving better security than KP-ABE. Therefore, in this study, we adopt the CP-ABE technique for access control.

Several schemes have been introduced to enforce the security and privacy of users and their data using CP-ABE. Broadly, they can be classified into two main categories:(1) *Single-Authority CP-ABE* (SA-CP-ABE), and (2) *multi-authority CP-ABE* (MA-CP-ABE). In SA-CP-ABE, one single authority is responsible for verifying the user's attributes and providing corresponding encryption and decryption credentials, making it a security and privacy threat if the authority is corrupted. For resolving this issue, MA-CP-ABE schemes have been proposed where multiple AAs participate in the key-generation process [7]. Each AA is responsible for verifying a disjoint set of attributes and providing partial keys to users based on the users' provided attributes [8]. These individual AAs are independent, and there is no coordination among them during the key generation process. This preserves the privacy and security as the individual attribute authority cannot identify the users' identity and their associated attributes.

The existing strategies may suffer from scalability and performance issues as they mainly focus on the data's security aspects. Furthermore, most of the existing strategies suffer from SPoF. Therefore, we put forward a scalable multi-agent CP-ABE architecture to tackle the issues above in this work. [9]. Contrary to the existing approaches where each computing host provides a single authority, our strategy has two unique factors. The first unique factor is that every authority (CA/AA) has at least two copies (called agents) in the system. The second factor is at least two authority agents will run on every computing host; these factors result in better system utilization, and throughput [10]. Moreover, no two types of the same kind can run on the same host, making it resilient against the SPoF. The public cloud may suffer security issues like uploaders may upload malicious apps or data. Some authorization and authentication mechanism in the private cloud eliminates chances of loading malware [11]. Malware can be ransomware, Trojan, and botnet [12] [13].Our proposed work provides security from malware as we have proposed malware protection. Our work has an effective mechanism of malware detection based on regress analysis [14]. In short, our main contributions are: (1) We introduce multi-agent CP-ABE architecture that is dynamically scalable and efficient in terms of performance; (2) We solve SPoF issue by running multiple agents of the same authority on different hosts; (3) our work based on regress analysis provides malware protection (4) We preserve the security, privacy, and fine-granularity features of CP-ABE [15] ; (5) Comparative analysis against current state-of-the-art strategies is shown using simulation results.

Approaches in the literature have issues of scalability, efficiency and have no mechanism for detecting and reporting the malicious material. The performance issue is also the foremost concerning factor in the quest to secure the data. Most of the existing approaches encounter the problem of SPoF. Therefore our proposed work contribution is to present scalable multi-agent CP-ABE to resolve the single point of failure problem. Proposed MA-CP-ABE covers three factors: scalability to cater to SPoF provides efficiency and achieves performance compared to other state of the art approaches, and has the strategy to detect malware in the cloud. The malware detection approach is based on the graph embedding mechanism that exploits the art Gemini approach's latest state. Compared to the traditional techniques available in the literature, our approach has a unique factor of authority (CA/AA) with at least two copies considered agents. The other different factor our work has is the two authority agents run on any computing hosts. These different aspects of our work have resulted in effective utilization and throughput.

We have compared our work with other existing approaches and found our approach is more effective and efficient. In terms of the security aspect, we have compared our work with other cloud-based malware detection approaches, and comparison results show our approach is much better in terms of evaluation parameters defined in the proposed approach section. We compared Xiao, the Trustav, and Dey techniques for malware and compared other approaches for encryption and perfromanace. The comparison revealed our proposed work is advance in performance, scalability, and security.

The rest of the paper is structures as; Section II briefly presents current state-of-the-art approaches related to CP-ABE access control for MCC. Section III highlights the background knowledge; Section IV elaborates the leading entities in the proposed architecture. Section V explains the internal working of the proposed CP-ABE system. Section VI presents the contributed approach for malware detection. Section VII provides detail for malware detection datasets used for malware detection VIII. It describes malware detection results, and Comparison IX simulation results demonstrate the effectiveness of our proposed architecture against existing systems. In the end, Section X concludes the paper and describes future extensions in the proposed work.

II. RELATED WORK

In this section, we highlight existing access control mechanisms based on ABE and Blockchain technologies. Related work is divided into three sections A, B, and C, to explain different aspects of work.

A. ABE BASED ACCESS CONTROL STRATEGIES

Different ABE-based mechanisms have been introduced to tackle both data security and privacy issue and the user's identity. The most promising scheme is CP-ABE. Recently, Unki et al. [5] conducted a comparative study of CP-ABE and KP-ABE schemes. According to the authors, the CP-ABE performs better in efficiency, privacy, and access control instead of KP-ABE. Therefore, in this work, we adopt the

CP-ABE strategy. A significant quantity of literature exists that uses CP-ABE to facilitate data sharing in the MCC environment. Studies in [16], [17], [18], define single authority to verify user's attributes and assign him/her credentials for encrypting and decrypting data. However, in this mechanism, the single administrative authority can decipher the data as it has all the attributes and associated credentials [19] [20]. So the security and privacy of the data cannot be guaranteed if the authority is compromised. Moreover, this mechanism suffers from SPoF as a single entity monitors all user attributes at the server side.

In order to solve the security and privacy issue, Chase [21] proposed a system where multiple attribute authorities have been introduced. The main goal is to divide the set of attributes into more than one disjoint set of attributes. Each disjoint set of attributes is then assigned to a single AA. In this mechanism, individual attribute authority is not able to decipher the cipher-text. Moreover, the identity of the user is kept private using anonymous key issuing protocol [22]. The attribute authorities are independent and do not communicate with each other. However, these mechanisms suffer from user collusion problems where more than one user can pool their partial private credentials to obtain the full credentials [23].

To improve the security, an improved version of multiple attribute authority schemes are proposed in [24] [25] [26] where a *Certification Authority* (CA) is responsible to verify users using their static attributes and issue a certificate. The user can then collect partial keys from each AA using the provided static attributes. However, these mechanisms ignore user devices' dynamic attributes for the encryption and decryption of the data, hence less suitable for the mobile cloud environment. The MA-CP-ABE based mechanism called *RAAC* (Robust and Auditable Access Control) is presented in [27] [28] to overcome the SPoF and low performance issue where CA works as an administrator. The CA can generate private keys for the users' requests based on verifying the AAs' attributes. However, this scheme assumes that CA is a trusted authority. So if CA is compromised, the user's privacy is also compromised in addition to the privacy of the data.

The use of dynamic attributes such as location, and unlock failures has been realized in [29], [30], [31], [32]. Strategies presented in [8], [29] uses only location of the user to encrypt and decrypt ciphertext. However, there is only a single authority, resulting in a user's privacy breach when the authority is compromised. More advanced and pioneering work is presented in [20] where the authors considered other dynamic attributes and the user's location. Moreover, a semi-trusted authority is introduced between the user device and the authorized agents to overcome unreliable mobile data networks. A lightweight data sharing scheme is proposed [19], where the authors introduced the semi-trusted cloud node for encryption and decryption services. The user can offload the encryption and decryption to the cloud server. The authors used symmetric key cryptography for data encryption on the client side. However, the symmetric key is encrypted on the cloud using the data owners' obtained credentials and their dynamic attributes. However, the system does not guarantee privacy as the semi-trusted cloud server can be compromised as the symmetric key is sent to the cloud server for encryption.

A few studies in literature focused mainly on the performance and reliability of the CP-ABE system. Agrawal et al. [1] presented a multi-authority CP-ABE system where they introduced a pair of agents to solve the disconnection or weak connection problem with the cloud storage server. A client-side and Server-side agent pair is responsible for handling the connection at the client side and the server side, respectively. However, this approach also suffers from SPoF in case if CA or any of the AA fails. Recent work in [24] extends [1] and solves the SPoF problem by introducing agent-based authorities where an agent can be a CA or AA. Moreover, every agent is responsible for keeping a backup of some other agent and can start the backup agent from its last updated state in case of failure. However, the proposed approach results in resource usage imbalance when the backup node runs a failed agent while other computing nodes run only a single agent. Consequently, the performance of the whole system will substantially degrade.

After a comprehensive study of the existing literature, we conclude that both SA-ABE and MA-ABE schemes suffer from performance and scalability issues. According to [6], the user might face huge communication delays during the key generation process as it has to communicate with several AAs independently. Therefore, in this work, we present a novel agent-based MA-CP-ABE system with a cloud node as a resource manager to solve the aforementioned issues.

B. BLOCKCHAIN-BASED ACCESS CONTROL

The block chain based mechanisms use encryption and ensure data confidentiality by allowing only authorized users to obtain the decryption keys for the cipher-text [9]. There are mainly two categories of blockchains: (1) Public Ledgers and (2) Permissioned Ledgers. In a public ledger based mechanism, an entity can freely join and leave the system. However, the permission based ledger does not allow the participating entities to enter or leave a system freely. Any entity that wants to be part of the access control system must obtain an identity first so that other entities can recognize it [29].

Existing access control mechanisms are not compatible with the block chain technology as they depend on a central trusted node for managing and enforcing access control policies [29]. Blockchain allows tackling this issue by allowing distributed participants to keep a local copy of the ledger. Changes to the ledger are accepted or rejected after running a consensus protocol. This kind of access control is resilient and does not suffer from availability issues [3].

C. MALWARE DETECTION IN MOBILE CLOUD

Cited approaches for data migration to the cloud have different malware scanning effectively. The reviewed approach is for Trojan detection, like malware game detection for mobile

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3067284, IEEE Access

devices [33]. Another approach is for mobile cloud data fusion based that uses machine learning [34] [35]. Other many approaches for privacy preservation for malware analysis [36] however non of approach is effective and productive for evolving android malware and is not integrated with the proposed architecture for mobile cloud. In nut shell, although we exploit CP-ABE based to ensure security and reliability while sharing organizations' data on public clouds. The architecture we propose in this paper can be easily extended, and block chain technology can easily be integrated to ensure data confidentiality. Our proposed approach covers three aspects: scalable, efficient, and more secure than other techniques in the cited literature. It has a robust mechanism for malware detection.

III. DESCRIPTION OF PROPOSED APPROACH

This section presents a brief background of the proposed approach and background information related to the CP-ABE system.

A. BI-LINEAR MAPS

Let $\mathbb{G}$, and $\mathbb{G}_T$ define two multiplicative cyclic groups. Let p be the common prime order and $\mathcal{G}$ be a generator of $\mathbb{G}$. Let e: $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ be a map with the following properties:

- *Bilinearity*: $\forall\ w,\ x\ \in\ \mathbb{Z}_p$, we have $e(\mathcal{G}^w,\ \mathcal{G}^x) = e(\mathcal{G},\ \mathcal{G})^{\mathrm{wx}}$.
- *Non-Degeneracy*: There exist $\mathcal{G}_1,\ \mathcal{G}_2\ \in\ \mathbb{G}$, such that $e(\mathcal{G}_1,\ \mathcal{G}_2)\ \neq 1$.
- *Computability*: There is an computationally efficient algorithm which can compute $e(\mathcal{G}_1,\ \mathcal{G}_2)\ \forall\ \mathcal{G}_1, \mathcal{G}_2\ \in \mathbb{G}$

B. COMPLEXITY ASSUMPTIONS

The *Decisional-Bilinear Diffie–Hellman* (DBDH) assumption: Let $w, x, y, z, u\ \in\ Z_p\ and\ \mathcal{G}\ \in\ \mathbb{G}$ be a generator. It can be said that DBDH assumption [18] [6] holds in $\mathbb{G}$ if there is no probabilistic algorithm which can differentiate the tuples $T_1\ =\ [\mathcal{G}, \mathcal{G}^w, \mathcal{G}^x, \mathcal{G}^y, e(\mathcal{G}, \mathcal{G})^{wxy}]$ and $T_2\ =\ [\mathcal{G}, \mathcal{G}^w, \mathcal{G}^x, \mathcal{G}^y, e(\mathcal{G}^z]$ in polynomial time with non-negligible benefit.

The *Decision-Linear* (DLinear) assumption: It can said that the DLinear assumption [18] holds in $\mathbb{G}$ if there is no probabilistic algorithm that can differentiate the tuples $T_1\ =\ [\mathcal{G}, \mathcal{G}^w, \mathcal{G}^x, \mathcal{G}^{wy}, \mathcal{G}^{xz}, \mathcal{G}^{y+z}]$ and $T_2\ =\ [\mathcal{G}, \mathcal{G}^w, \mathcal{G}^x, \mathcal{G}^{wy}, \mathcal{G}^{xz}, \mathcal{G}^u]$ in polynomial time with non-negligible benefit.

C. CP-ABE

The CP-ABE consists of four algorithms:

(1) Setup(λ, υ) $\rightarrow\ \varkappa$: The Setup algorithm takes λ and υ as input and generates κ and $\varkappa$ as an output, where λ is the security parameter, υ is the description of attributes universe, κ defines the public parameters, and $\varkappa$ is the master private key.

(2) Encrypt(κ, M, $\mathbb{AS}$) $\rightarrow$ **C**: This algorithm receives κ, M, $\mathbb{AS}$ as input and outputs **C**, where M defines the original message, $\mathbb{AS}$ is the access structure, and **C** is the cipher-text produced by the encrypt algorithm.

(3) KeyGen($\varkappa$,S) $\rightarrow\ SK$: The KeyGen takes $\varkappa$ and S (the attributes set) as input and outputs a secret key SK.

(4) Decrypt(κ, C, SK) $\rightarrow\ M$: The Decrypt function receives κ, **C**, and SK as input and generates the original message M.

IV. SYSTEM ARCHITECTURE AND SECURITY REQUIREMENTS

In this section, the system's architecture and security requirements are explained. The following subsections will describe security requirements and the model.

A. SYSTEM ARCHITECTURE

We present an agent-based MA-CP-ABE architecture as shown in Figure 1: system architecture. We define agents for the main entities (CA, AA, and CS). However, we also define some additional agents in order to achieve our stated goals such as scalability, availability, and efficiency. The agents defined in our system are described below:

- **Certificate Authority Agents**: Let $CA = \{CA_1, ..., CA_m\}$ be a set of all CAs indexed by i and m represents the total number of CAs; m is equal to 1 in our system. Each CA_i has more than one agents/instances denoted by $CAi = \{CA_{i1},\ CA_{i2},\ ...,\ CA_{in}\}$ distributed over multiple hosts.
- **Attribute Authority Agent (AA)**: Let $AA\ =\ \{AA_1,\ AA_2,\ ...,\ AA_k\}$ be a set of all AAs indexed by j where each AA_j manages a disjoint set of user attributes and their corresponding partial encryption/decryption credentials. Each AA_j has further ℓ number of agents denoted by $AA_j\ =\ \{AA_{j1},\ AA_{j2},\ ...,\ AA_{j\ell}\}$ where each agent that belongs to an AA_j maintains the same set of attributes and hence, the same encryption/decryption credentials. In case of failure of one or more hosts, the system shows graceful degradation as at least one other copy of a failed agent must be running in the system until the whole system goes down. Moreover, it improves a CP-ABE system's performance by distributing multiple incoming requests to two or more computing nodes.
- **High-Level Certificate Agent (HLCA)**: HLCA works as an intermediary between the user and the instances of CA in the system. It distributes the incoming request for certificates from end users in the round-robin fashion to the available CAs evenly to improve system utilization. Moreover, the HLCA is placed in a cloud node, making it accessible and reliable from the end user's point of view. HLCA is for reliability and availability in the cloud environment.
- **High-Level Attribute Agent (HLAA)**: Like HLCA, HLAA resides in a cloud node and uses round-robin scheduling to distribute the incoming requests for encryption or decryption credentials from end users to one of the corresponding instances of each AA. The main

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3067284, IEEE Access

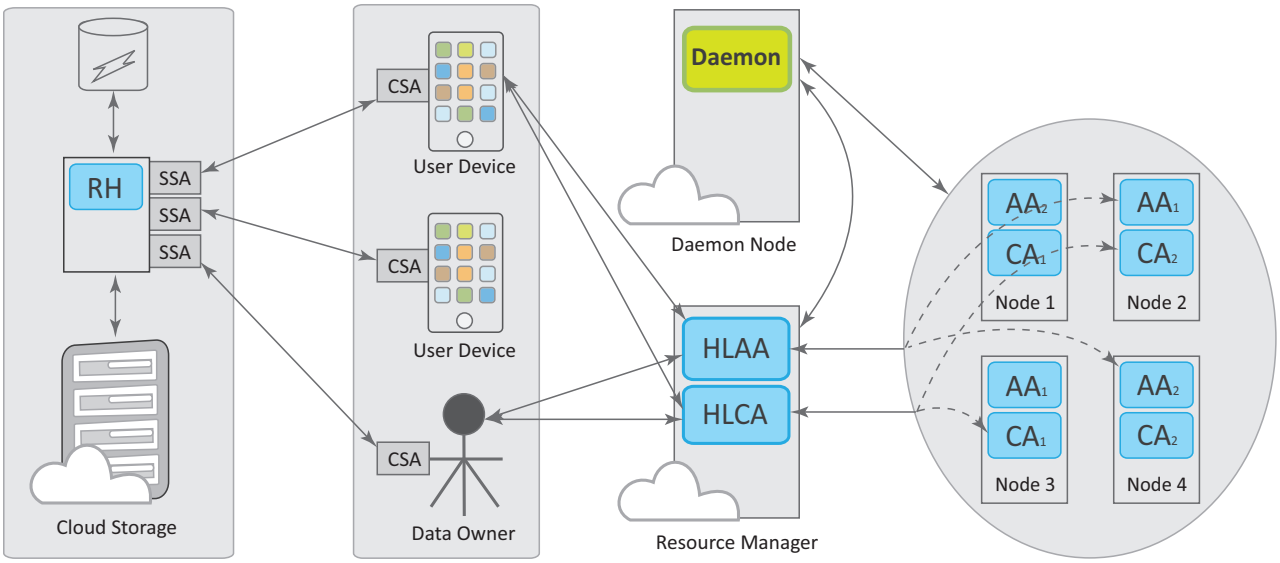


FIGURE 1: Architecture of the proposed system.

advantage of using HLAA is that the user does not need to communicate with each AA separately. Instead, the HLAA takes care of most of the communication with AAs. We will discuss the privacy and security aspects of this step in Section VI.

- **Daemon Agent (DA)**: The DA is kept in a separate cloud node. This agent works as an eye-keeper and is responsible for keeping track of all the agents of CAs, AAs, HLCA, and HLAA. In case of failure of any agent, the Eye-Keeper asks another agent of the same parent (CA, AA, HLCA, HLAA) to clone and migrate to some other node by calling the clone() and the migrate() functions, respectively.
- **Pair of Client-Side and Server-Side Agents**: To tackle the weak or unreliable connection problem of mobile networks, we adopt the technique presented in [1], [24] where a pair of Client-Side Agent (CSA) and Server-Side Agent (SSA) has been introduced on the client and cloud server-side respectively for each end-user connection. These agents use different techniques, such as caching and compression, to tackle unreliable network connections.

B. SECURITY ASSUMPTIONS

In the proposed model, the following assumptions are made related to the security of the system. The cloud server is managed by a third-party service provider and is always online. The cloud service is always honest in carrying out the tasks assigned; however, it is curious to mine secret information from the stored data. Moreover, it is also assumed that the HLCA and HLAA cannot be trusted and can be compromised as they are also run on third party cloud infrastructure. The users may collude with other users to get the private keys for decryption or even collude with a compromised AA to obtain higher illegal privileges. Finally, the data owners define access policies for the uploaded data.

subsectionSecurity Requirements In order to guarantee security and privacy in public clouds, the system must fulfill the following key security aspects:

- *Confidentiality:* An unauthorized user must not be able to access sensitive data.
- *Collusion Resistance:* The system must prevent malicious users from combining their attributes to calculate decrypted credentials if they cannot decrypt the data alone.
- *Identity theft resistance:* The system must prevent AAs from knowing the identity of the users during their communication with the AAs.ge M.

In this section, we highlight the main entities that constitute our proposed system model. Moreover, We also describe the security requirements and assumptions related to access control for shared data in the MCC environment. However, before going into the detailed description, we define the main entities of the proposed MA-CP-ABE model as follows:

- **Certificate Authority (CA)**: This entity is responsible for verifying the user and issuing a certificate.
- **Attribute Authority (AA)**: The user uses the certificate issued by the CA to communicate with AAs to get the partial keys (encryption/decryption). The AA is responsible for maintaining a disjoint set of user attributes and generating partial credentials for encryption and decryption based on the user's provided attributes. There are multiple AAs, however, which run on different hosts.
- **Mobile Device (MD)**: Mobile host defines the end user interacting with the certificate or attribute authority. They can be mobile devices, laptops, etc.
- **Cloud Server (CS)**: The CS is responsible for storing the organizations' data and allow the members of the organization to access the shared data from anywhere, anytime.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3067284, IEEE Access

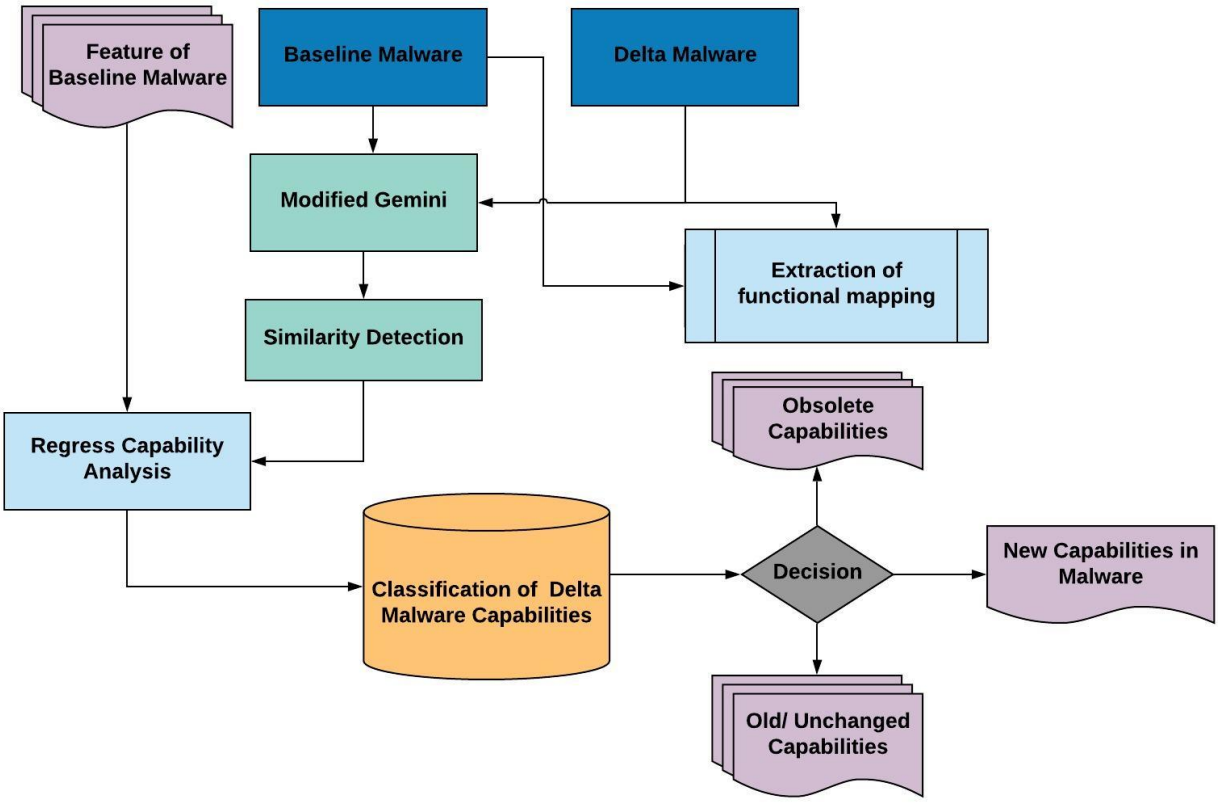


FIGURE 2: Proposed Regress Analysis for Graph Embedding Binary Code

In our proposed system, we define agents for each entity. An agent is an instance of a particular entity (CA, AA, MD, or CS) and provides the respective services. Every agent goes through several states during its lifetime. We define the following states of an agent adapted from [1] with slight modifications:

- *Created*: The *create()* function is used to instantiate a particular agent in the system.
- *Initialized*: After the creation of an agent, the *init()* function is called to initialize the state of the agent.
- *Activated*: An agent is activated and starts providing services after the *activate()* function is called.
- *Deactivated*: An agent can be deactivated to pause the service provisioning. To resume the services, the *activate()* function must be called so that the agent starts providing its services again.
- *Migrated*: An agent is migrated from host A (source) to host B (destination) by calling the *migrate()* function.
- *Cloned*: A *clone()* function is used to create a copy of an agent with Deactivated state.
- *Updated*: The state of the agent is updated using the *update()* function.
- *Disposed*: Lastly, to stop an agent and free up the resources it is occupying, the *dispose()* function must be called.

V. PROPOSED MA-CP-ABE SYSTEM

In this section, we will discuss the proposed CP-ABE system.

A. OVERVIEW

To ensure a scalable, efficient, and reliable CP-ABE based access control for MCC, we introduce an agent-based MA-CP-ABE architecture. In the proposed system, we introduce intermediary agents residing in a cloud node that handles most of the communication between User Device (UD) and authority agents (CA/AA). There are multiple agents of CA, and each AA running on different hosts. Two agents having the same services cannot reside in the same machine as the host machine's failure will cause both instances (agents) of the same authority to be unavailable, leading to unreliable service.

The first step in the proposed agent-based system is verifying the user's legitimacy by the CA. The CA issues a certificate to the user if successfully verified and encrypts the certificate using its private key. The user then requests each AA for the corresponding partial credentials (encryption or decryption). However, instead of communicating with individual AA or CA, the UD communicates with the HLAA or HLCA. This mechanism allows distributing the user requests to multiple hosts, consequently improving system performance. So, the user communicates with the HLAA in order to obtain the encryption/decryption key. The HLAA

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3067284, IEEE Access

Algorithm 1: Data_Upload_With_Malware_Detection

Output: Secure Data Upload to Public Cloud
initialization
while *Similarity Check Based on Graph Embed ()* **do**
instructions
if *Similarity with Malicious Existing Malware instances* $\geq$ *Existing Malware Instances* **then**
Determine Class of Malware()
else
Transfer to MA-CP-ABE ()

communicates with each AA on behalf of the UD and gets partial keys from each AA. The HLAA combines the partial keys and sends them back to the UD. In this way, the user needs one time connection with the HLAA to get the partial keys. The proposed system follows the Anonymous Key-Issuing protocol, which ensures the privacy of the user.

After Receiving the credentials, the data owner and the data accessor can combine the partial credentials and create a full encryption/decryption key. The data owner uses the encryption key and his/her dynamic attributes to encrypt the data and saves it on the cloud. Similarly, the data accessor downloads the encrypted data from the cloud and uses the decryption key and his/her dynamic attributes to decrypt the downloaded data. All this is achieved in four steps in our proposed system: system initialization, key generation, encryption, and decryption. We elaborate on each of these steps in the following subsection.

B. PROPOSED REGRESS ANALYSIS OF MALWARE BASED ON GRAPH CONVOLUTION

Private cloud is trustworthy; however, in the public cloud, there are more security concerns. In the case of private cloud, just authentication can not be considered reliable. Some time even uploading authority may be unaware of malicious apps from unreliable sources. Before uploading, there is a need for a second check other than authentication to ensure the uploader is uploading the trustworthy contents. Our proposed approach uses graph convolution based approach for malware. Our approach uses the preexisting knowledge base of malicious app patterns.

When a new app is considered for evaluation, it uses that knowledge base to check the current app's similarity under evaluation with the existing malicious patterns based on a graph similarity measure. Mentioned in Figure 2 Proposed Regress Analysis for Graph Embedding for Binary Code, Shows the proposed arrangement. The regression testing technique inspires our work. The regression testing approach is effective for testing the evolving apps. It helps to reduce the efforts and time [14]. In case the graph similarity is more than the threshold, it considered it similar and categorized it malicious with the existing category.

Illustration of proposed malware detection strategy is described in Fig. 2: Algorithm for malware detection in cloud.

Laber for Dataset	Number of Instances	Description
EMBER2017 [38]	1.1 Million files	PE files
Kharon Dataset [39]	1250	Around 49 families.
CICAAGM [40]	1860	Three kinds of app

TABLE 1: Datasets used for Malware Detection System

The threshold of similarity is defined. In case the data being uploaded has the similarity equal to or greater than the threshold. Such malware is categorized as malware based on this similarity indicator. On the contrary, if the similarity is less than the threshold, data is transferred to MA-CP-ABE. Proposed MA-CP-ABE will apply encryption in a fast way. On the contrary threshold, the similarity is less than a threshold; it will not be classified to the existing knowledge. We have exploited the Gemini approach for the graph similarity [37]. It is an efficient and less resource consuming approach.

C. DATA SET FOR PROPOSED MALWARE DETECTION APPROACH

Following data sets are used used to evaluate our proposed approach for malware detection in the mobile cloud. Our presented methodology for malware detection in the cloud environment has exploited the data set for malware mentioned in Table 1. These three different malware datasets are enriched and diversified with various attributes. Ember dataset is published in 2017 with 1.1 million files [38]. The second data set with label Kharon has 49 different files with around twelve hundred and fifty instances [39]. In comparison, the third dataset with the label CICAAGM has around eighteen hundred instances with three different kinds of apps: benign, adware and generic [40]. To evaluate fairly and maintain transparency in the assessing mechanism, we have used these three datasets to compare all the techniques used.

D. RESULTS OF REGRESS ANALYSIS FOR MALWARE

In order to assess the presented work for malware prevention, we have evaluated the approach for the following evaluation parameters.

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{1}$$

equation 1 represents the accuracy parameter. we have used the accuracy parameter to evaluate and compare our proposed approach with other approaches. Here TP means True positives, and TN represents the True Negatives. Similarly, FP shows the false positive, and FN depicts the false negative.

$$Precision = \frac{TP}{(TP + FP)}) \tag{2}$$

Here, Precision is another is based on the true positive and false positive ratio as depicted in the above equation of Precision.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3067284, IEEE Access

Approach	Accuracy	Precision	Recall	F-Measure
MA-CP-ABE	98	97	96	96
Xiao2017	95	93	92	91
Trustav2020	96	94	88	90
Dey2019	94	92	80	89

TABLE 2: Comparison of Proposed Approach

$$Recall = \frac{TP}{(TP + FN)} \tag{3}$$

Similarly, the recall measure is used for calculating the F-measure. Evaluation to assess the recall is shown in the above equation.

$$F - measure = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)} \tag{4}$$

F-measure require the precision and recall to get the results. Equation 4 shows the F-measure calculation.

$$MAE = \frac{1}{n}\left(\sum_{i=1}^{n}\left|\frac{Predicted - Actual}{Actual}\right|\right) \tag{5}$$

The MAE is based on the ratio of predicted and actual. The ratio is used to measure the presented approach and compare it with other approaches. The figure 3: Comparison of Proposed Technique shows results of evaluation parameters of compared techniques. However, the figure 4 shows the comparison of the proposed approach with other state of the art approaches in the literature. Following 3 shows the comparison of the proposed approach with other state of the art approaches in the literature. We have compared our approach with the latest and closely related to our proposed work for malware detection. Comparison of presented work with other three approaches proposed by Xia, the Trustav, and Dey. All these approaches are evaluated on the same datasets to evaluate proposed approaches fairly.

Comparison of presented work with state of the art technique shows that our proposed work has provided better results in all four evaluation factors. Figure 3: Comparison of Proposed Technique shows Xiao's approach obtained accuracy 95 percent, precision 93, recall 92, and F-measure 91 [33].Statistical illustration of comparison with proposed malware detection approach is illustrated in Table 2. Evaluation parameters like precision, accuracy, recall, and F-measure are used for comparison. Similarly, the trustav approach obtained 96 percent accuracy, 94 precision, 88 recall, and 90 F-measure [36]. The third approach compared with the presented approach was posed by deyannis. This approach gives accuracy 94, precision 92, recall 80, and F-measure 89. However, our approach beats these three techniques in all evaluation parameters as it obtains 98 percent accuracy, precision 96, recall 94, and F-measure 93. So results show that our approach is more precise, accurate, and reliable with less false-positive rates. We will compare our proposed study in terms of performance and scalability to resolve the SPoF issue in upcoming sections. State of the art approaches are compared with the proposed approach incoming section to measure the performance.

E. DETAILS FOR THE PROPOSED SYSTEM'S INTERNAL WORKING

The current section elaborates on the internal working of the proposed system.

1) System Initialization

two tasks are performed during the system initialization process. Firstly, the CA selects $\mathbb{G}$ and $\mathbb{G}_t$ as two multiplicative cyclic groups having the common p. The CA then defines $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_t$ on $\mathbb{G}$ as binary map. Let $MSK = w,\ x,\ y,\ z$ be the master key where $w,\ x,\ y,\ and\ z\ \in\ \mathbb{Z}_p$ and are randomly chosen by the CA. Moreover, CA randomly creates public keys for each attribute A_i.

The second task during system initialization is the registration of users and AAs with the CA, where CA first generates private and public key pair, which is used to sign and verify. During the system initialization, each AA requests for registration to CA. CA assigns $AA_{id}\ \in\ \mathbb{Z}_p$ (a unique id) and chooses $K_{\mathrm{AA_{id}}}\ \in\ \mathbb{Z}_p$, and then the corresponding $PK_{\mathrm{AA_{id}}}$ is calculated. Lastly, a certificate is generated and sent to the AA with id AA_{id}. Similarly, a user requests CA for registration, where a unique U_{id} is assigned. CA then chooses $K_{\mathrm{U_{id}}}$ and generates $Cert_{\mathrm{U_{id}}}$ including the public key and is sent to the user.

2) Hidden Keys Generation Process

The HLAA obtains encrypted partial keys from each AA on behalf of the user. The process works as follows: The user sends the request for encryption/decryption key to the HLAA along with the certificate issued by CA and the required static attributes. The HLAA requests each AA for the partial key and forwards them the static attributes and the user's certificate. Each AA first checks if the certificate provided is valid using the public key of the CA. Lastly, after verifying the user attributes, AA sends the partial key against the provided attributes. The HLAA, being a cloud node, cannot be trusted, and therefore, the partial keys are encrypted by the AAs using the user's public key. So the HLAA combines and sends the encrypted partial keys to the end user (data owner/accessor).

3) Encryption

The data owner first obtains the encryption credentials from the HLAA, encrypts the data using the received credentials and the dynamic attributes, and uploads the encrypted data to the cloud server.

4) Decryption

Firstly, the user is allowed to download any file free from the cloud. Then data accessor obtains all the partial keys and then combines them to get the full private key. Then he/she uses the private key together with the dynamic attributes to decrypt the file downloaded from the cloud server.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3067284, IEEE Access

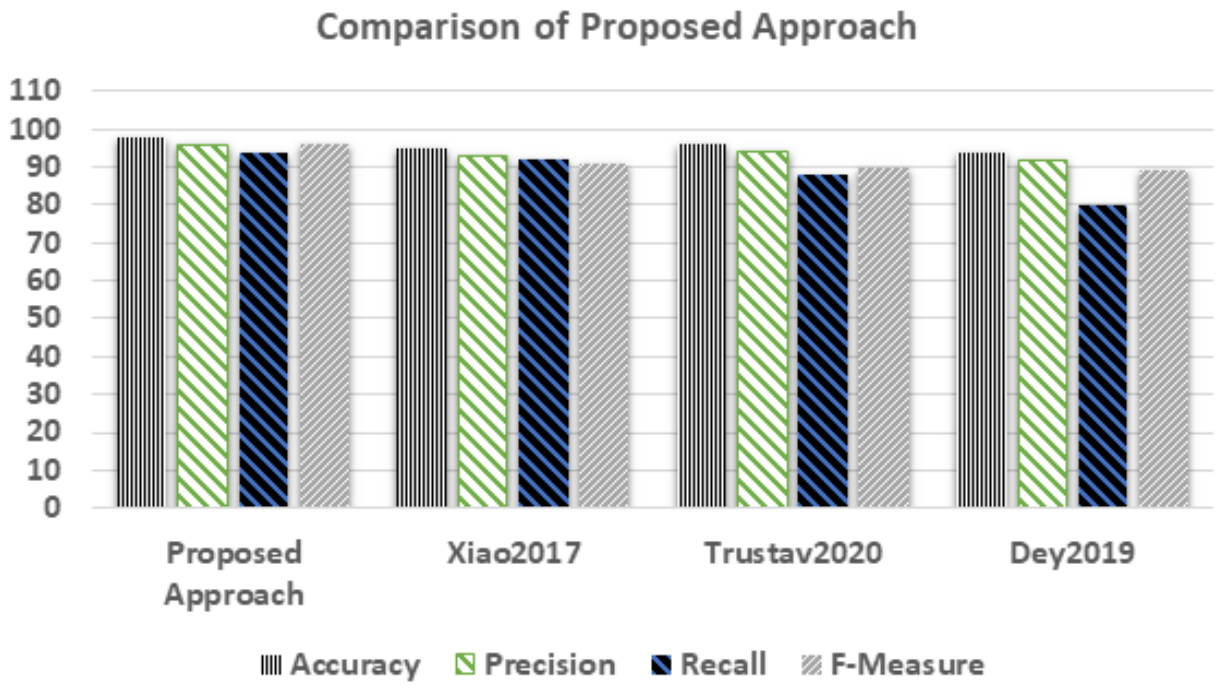


FIGURE 3: Comparison of Proposed Technique

VI. PERFORMANCE EVALUATION

Before going into the detailed discussion of the experimental results, we briefly describe the simulation environment and the existing strategies [1], [24] with which we compare our proposed scheme. We implement our proposed strategy in java using the CP-ABE library realized by [10], which uses pairing based cryptography presented in [9]. We conduct simulations on Core i5 2.2 GHz machine having 16 GB of RAM and Windows 10 installed. The system is developed as a client-server application using "Socket APIs" in java. In our experiments, we have only one CA and four AAs. We analyze the proposed methodology in terms of computation overhead for authorization. Moreover, we also discuss how our proposed system ensures availability when one or more CA or AA agents fail.

A. SYSTEM AVAILABILITY IN CASE OF FAILURE

When the failure of one of the certificate authorities or attribute authorities occurs, the system proposed in [1] cannot provide its services as there is only one CA, and each AA maintains a disjoint set of attributes. While the scheme presented by Jamal et al. [24] handles this single-point-of-failure problem by keeping a backup of every agent on some other node and runs it when the agent fails. However, while starting a backup agent from the backup state, the system will not provide its services. So during this period, the system will be unavailable to process incoming user requests. In contrast, our proposed scheme runs multiple instances of each authority in advance, and hence, the system is available until and unless all the host machines fail. Moreover, while other instances providing their services, the Daemon creates copies of the failed agents and assigns them to run hosts.

B. COMPUTATION OVERHEAD FOR AUTHORIZATION

n the current section, we will describe the computation overhead for authorization

C. DURING NORMAL OPERATION

We calculate the time taken for the authorization process. It includes two steps: first, the user devices start communicating with the authorities for credentials; second, the user gets the response from all the authorities. Fig. 4 depicts milliseconds for the authentication process in normal conditions when all the host machines are working normally. In the graph, the x-axis represents the number of attributes used for authorization, while the y-axis shows the time taken in milliseconds. The graph clearly shows that, compared to its competitors, the proposed system takes less time for authentication of users due to the involvement of cloud nodes, which reduces the communication overhead between the user and the attribute authorities.

1) When one of the host machines fails

Similarly, we also studied the situation where one of the running hosts fails. This time, the system proposed in Ref. [1] is unavailable as the CA or AA running on that host fails. However, in the case of [24], the system starts a backup agent on one of the available hosts. As a result of such a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3067284, IEEE Access

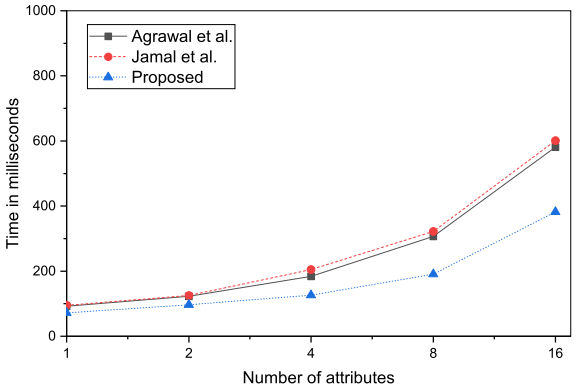


FIGURE 4: The computational overhead in millisecond for authorization process without any failure

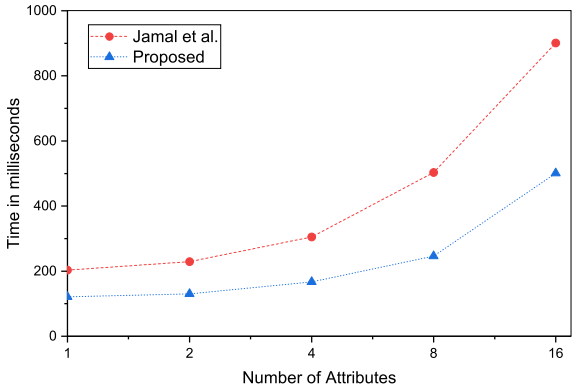


FIGURE 5: The computational overhead in millisecond for authorization process after one of the AA agent fails

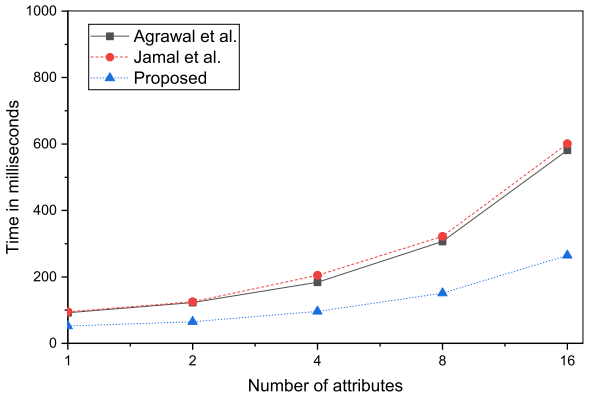
backup procedure, there is an imbalance of agent distribution over the available hosts. One of the hosts would run two attribute authorities while others run only one. In contrast, the proposed strategy (the Daemon agent) creates multiple instances of each authority (AA and CA) and distributes them evenly on the available host machines. To achieve this, the Daemon agent may create new instances or dispose of existing ones. Moreover, the incoming user requests are handled by an intermediary cloud node (HLCA/HLAA). The HLCA and HLAA use the round-robin algorithm to distribute the incoming requests among an agent's multiple running instances. This mechanism makes the proposed scheme scalable as one can add or remove host machines from the system at run time, and the Daemon agent will recalculate the number of instances to be created or disposed of. Fig. 5 shows the time taken for the authentication process when one of the hosts fails in the system. It can be observed that the proposed scheme performs better as compared to the strategy presented by Jamal et al. [24].

2) When new nodes are added

In order to analyze the scalability of our system, we add two new host machines and monitor the performance. The existing systems proposed in [1] and [24] do not allow the addition of new nodes. However, in the proposed system, the Daemon agent creates more copies of the authorized agents and dispatches them over the new host machines. Fig. 6 depicts the proposed system's effectiveness as opposed to its counterparts when two new nodes are added to the existing cluster. It is obvious that the proposed system significantly reduces the time taken during the authorization process when new nodes are added.

D. DISCUSSION OF RESULTS

results of our proposed work are illustrated in figures 3,4,5, and 6. Results are evaluated and compared in two aspects. The first aspect of result evaluation is for MA-CP-ABE



..

FIGURE 6: The computational overhead in millisecond for authorization process after new nodes are added

with other literature approaches like Agrawal and Jamal's approaches. However, for malware detection in the cloud, the regress malware detection approach in our work is compared with Xiao, Trustav, and Dey. Overall results in both aspects are better to compare to the latest cited approaches. Our proposed strategy has outperformed the malware detection and encryption methodology. Therefore our proposed work is preferable as compared to the others in the literature.

VII. CONCLUSION

In this work, we proposed a novel agent-based MA-CP-ABE system to improve system performance and avoid SPoF. Specifically, we create and run multiple agents of the same authority and distribute them over multiple hosts. Hence, the system can provide its services in case of failure of one or more agents. We introduced an intermediary cloud node for distributing the incoming user request for certificates and partial credentials to the CAs and AAs to improve

system performance. From the security point of view, the proposed system inherently ensures the privacy of both the users and their data and is resistant to collusion attacks. We also proposed a malware detection mechanism to protect the cloud environment. Our proposed mechanism for malware detection is efficient and effective, as results show it's comparatively better than the cited latest approaches.

Finally, simulation results demonstrate that the proposed system reduces the communication overhead between the user and the AAs, significantly improving the system's response time. Moreover, the parallel execution of multiple instances of the same authority eliminates the SPoF issue. It improves performance by reducing the resource usage imbalance in case of failure of existing hosts. The proposed system is scalable for future workload increase as one can add more host machines dynamically at run-time. Our proposed scheme provides the security mechanism for malware. It detects malware in a more accurate, precise, and efficient way. Our approach provided a scalable, efficient, and secure architecture for the cloud environment. So it's novel as it covers three aspects differently.

VIII. FUTURE WORK

In the future, we will extend our current work using transfer learning for malware detection in the cloud. Our future work will utilize the random forest approach for feature extraction and selection of relevant attributes. Our future will be effective in malware detection and will use lesser resources, and will be efficient. We will extend our current multi-agent architecture and will use encryption techniques using advance deep learning approaches. The current architecture will be extended using Generative Adversarial Networks. We have targets and research objectives to get better results in all three aspects of scalability, efficiency, and malware detection in future work.

We have a plan to extend our work to apply the latest transfer learning approaches in the future. We plan to integrate the random forest tree approach for feature extraction and then apply the transfer learning approach to learn effectively. A combination of a haphazard forest tree approach and transfer learning will be effective for malware detection and prevention. Similarly, we will extend our multi-agent architecture using more robust and efficient approaches as there is still scope for improvement in our proposed architecture.

ACKNOWLEDGMENT

We acknowledge the efforts of Mr. Rizwan Khan, Mr. Adeel Aslam for supporting and helping us in this work.

REFERENCES

[1] Neha Agrawal and Shashikala Tapaswi. A trustworthy agent-based encrypted access control method for mobile cloud computing environment. Pervasive and Mobile Computing, 52:13–28, 2019.

[2] Ming Li, Peng Lin, Gangyan Xu, and George Q Huang. Cloud-based ubiquitous object sharing platform for heterogeneous logistics system integration. Advanced Engineering Informatics, 38:343–356, 2018.

[3] Mehdi Sookhak, F Richard Yu, Muhammad Khurram Khan, Yang Xiang, and Rajkumar Buyya. Attribute-based data access control in mobile cloud computing: Taxonomy and open issues. Future Generation Computer Systems, 72:273–287, 2017.

[4] Yehia Kotb, Ismaeel Al Ridhawi, Moayad Aloqaily, Thar Baker, Yaser Jararweh, and Hissam Tawfik. Cloud-based multi-agent cooperation for iot devices using workflow-nets. Journal of Grid Computing, 17(4):625–650, 2019.

[5] Prakash H Unki, Suvarna L Kattimani, and BG Kirankumar. Cp-abe based mobile cloud computing applicatıon for secure data sharing. In International Conference on Intelligent Data Communication Technologies and Internet of Things, pages 561–568. Springer, 2019.

[6] Yinghui Zhang, Robert H Deng, Shengmin Xu, Jianfei Sun, Qi Li, and Dong Zheng. Attribute-based encryption for cloud computing access control: A survey. ACM Computing Surveys (CSUR), 53(4):1–41, 2020.

[7] Kshitij Bakliwal, Maharshi Harshadbhai Dhada, Adria Salvador Palau, Ajith Kumar Parlikad, and Bhupesh Kumar Lad. A multi agent system architecture to implement collaborative learning for social industrial assets. IFAC-PapersOnLine, 51(11):1237–1242, 2018.

[8] Iwailo Denisow, Sebastian Zickau, Felix Beierle, and Axel Küpper. Dynamic location information in attribute-based encryption schemes. In 2015 9th International Conference on Next Generation Mobile Applications, Services and Technologies, pages 240–247. IEEE, 2015.

[9] Praveen Kumar, PJA Alphonse, et al. Attribute based encryption in cloud computing: A survey, gap analysis, and future directions. Journal of Network and Computer Applications, 108:37–52, 2018.

[10] Jiguo Li, Ningyu Chen, and Yichen Zhang. Extended file hierarchy access control scheme with attribute based encryption in cloud computing. IEEE Transactions on Emerging Topics in Computing, 2019.

[11] Gianni D'Angelo, Massimo Ficco, and Francesco Palmieri. Malware detection in mobile environments based on autoencoders and api-images. Journal of Parallel and Distributed Computing, 137:26–33, 2020.

[12] Rahim Taheri, Reza Javidan, and Zahra Pooranian. Adversarial android malware detection for mobile multimedia applications in iot environments. Multimedia Tools and Applications, pages 1–17, 2020.

[13] Khaled Bakour and Halil Murat Ünver. Visdroid: Android malware classification based on local and global image features, bag of visual words and machine learning techniques. Neural Computing and Applications, pages 1–21, 2020.

[14] Zahid Hussain Qaisar and Shafiq Ur Rehman. A safe regression testing approach for safety critical systems. Advances in Engineering Software, 42(8):586–594, 2011.

[15] Qinlong Huang, Yixian Yang, and Mansuo Shen. Secure and efficient data collaboration with hierarchical attribute-based encryption in cloud computing. Future Generation Computer Systems, 72:239–249, 2017.

[16] Hayden Wimmer, Victoria Y Yoon, and Vijayan Sugumaran. A multi-agent system to support evidence based medicine and clinical decision making via data sharing and data privacy. Decision Support Systems, 88:51–66, 2016.

[17] Long Li, Tianlong Gu, Liang Chang, Zhoubo Xu, Yining Liu, and Junyan Qian. A ciphertext-policy attribute-based encryption based on an ordered binary decision diagram. IEEE Access, 5:1137–1145, 2017.

[18] Yinghui Zhang, Xiaofeng Chen, Jin Li, Duncan S. Wong, Hui Li, and Ilsun You. Ensuring attribute privacy protection and fast decryption for outsourced data security in mobile cloud computing". Information Sciences, 379:42 – 61, 2017.

[19] Praveen Palanisamy. Multi-agent connected autonomous driving using deep reinforcement learning. In 2020 International Joint Conference on Neural Networks (IJCNN), pages 1–7. IEEE, 2020.

[20] Rui Wang, Miao Li, Limei Peng, Ying Hu, Mohammad Mehedi Hassan, and Abdulhameed Alelaiwi. Cognitive multi-agent empowering mobile edge computing for resource caching and collaboration. Future Generation Computer Systems, 102:66–74, 2020.

[21] Juan Du, Vijayan Sugumaran, and Bonan Gao. Rfid and multi-agent based architecture for information sharing in prefabricated component supply chain. IEEE Access, 5:4132–4139, 2017.

[22] Ishu Gupta and Ashutosh Kumar Singh. An integrated approach for data leaker detection in cloud environment. Journal of Information Science & Engineering, 36(5), 2020.

[23] Tadashi Ogino, Shinji Kitagami, Takuo Suganuma, and Norio Shiratori. A multi-agent based flexible iot edge computing architecture harmonizing its control with cloud computing. International Journal of Networking and Computing, 8(2):218–239, 2018.

[24] Fara Jamal, Mohd Taufik Abdullah, Zurina Mohd Hanapi, and Azizol Abdullah. Reliable access control for mobile cloud computing (mcc) with cache-aware scheduling. IEEE Access, 7:165155–165165, 2019.

[25] Wei-Chih Chen, Wen-Hui Chen, and Sheng-Yuan Yang. A big data and time series analysis technology-based multi-agent system for smart tourism. Applied Sciences, 8(6):947, 2018.

[26] Takuo Suganuma, Takuma Oide, Shinji Kitagami, Kenji Sugawara, and Norio Shiratori. Multiagent-based flexible edge computing architecture for iot. IEEE Network, 32(1):16–23, 2018.

[27] Jun Wu and Xin Xu. Decentralised grid scheduling approach based on multi-agent reinforcement learning and gossip mechanism. CAAI Transactions on Intelligence Technology, 3(1):8–17, 2018.

[28] Kaiping Xue, Yingjie Xue, Jianan Hong, Wei Li, Hao Yue, David SL Wei, and Peilin Hong. Raac: Robust and auditable access control with multiple attribute authorities for public cloud storage. IEEE Transactions on Information Forensics and Security, 12(4):953–967, 2017.

[29] Qi Li, Jianfeng Ma, Rui Li, Ximeng Liu, Jinbo Xiong, and Danwei Chen. Secure, efficient and revocable multi-authority access control system in cloud storage. Computers & Security, 59:45–59, 2016.

[30] Wei Li, Kaiping Xue, Yingjie Xue, and Jianan Hong. Tmacs: A robust and verifiable threshold multi-authority access control system in public cloud storage. IEEE Transactions on parallel and distributed systems, 27(5):1484–1496, 2015.

[31] Yaser Baseri, Abdelhakim Hafid, and Soumaya Cherkaoui. Privacy preserving fine-grained location-based access control for mobile cloud. Computers & Security, 73:249–265, 2018.

[32] Yogachandran Rahulamathavan, Suresh Veluru, Jinguang Han, Fei Li, Muttukrishnan Rajarajan, and Rongxing Lu. User collusion avoidance scheme for privacy-preserving decentralized key-policy attribute-based encryption. IEEE Transactions on Computers, 65(9):2939–2946, 2015.

[33] Liang Xiao, Yanda Li, Xueli Huang, and XiaoJiang Du. Cloud-based malware detection game for mobile devices with offloading. IEEE Transactions on Mobile Computing, 16(10):2742–2750, 2017.

[34] Saurabh Dey, Qiang Ye, and Srinivas Sampalli. A machine learning based intrusion detection scheme for data fusion in mobile clouds involving heterogeneous client networks. Information Fusion, 49:205–215, 2019.

[35] Ram Mahesh Yadav. Effective analysis of malware detection in cloud computing. Computers & Security, 83:14–21, 2019.

[36] Dimitris Deyannis, Eva Papadogiannaki, Giorgos Kalivianakis, Giorgos Vasiliadis, and Sotiris Ioannidis. Trustav: Practical and privacy preserving malware analysis in the cloud. In Proceedings of the Tenth ACM Conference on Data and Application Security and Privacy, pages 39–48, 2020.

[37] Xiaojun Xu, Chang Liu, Qian Feng, Heng Yin, Le Song, and Dawn Song. Neural network-based graph embedding for cross-platform binary code similarity detection. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, pages 363–376, 2017.

[38] H. S. Anderson and P. Roth. EMBER: An Open Dataset for Training Static PE Malware Machine Learning Models. ArXiv e-prints, April 2018.

[39] Nicolas Kiss, Jean-François Lalande, Mourad Leslous, and Valérie Viet Triem Tong. Kharon dataset: Android malware under a microscope. In The {LASER} Workshop: Learning from Authoritative Security Experiment Results ({LASER} 2016), pages 1–12, 2016.

[40] Arash Habibi Lashkari, Andi Fitriah A Kadir, Laya Taheri, and Ali A Ghorbani. Toward developing a systematic approach to generate benchmark android malware datasets and classification. In 2018 International Carnahan Conference on Security Technology (ICCST), pages 1–7. IEEE, 2018.

ZAHID HUSSAIN QAISAR Is a research scholar and an assistant professor in department of computer science in NFC Institute of Engineering and Technology Multan, Multan, and a research scholar in the department of computer sciences and information technology, Huazhong University of Science and Technology, Wuhan, China. He has diversified and excellent research and professional experiences. He has also worked as Lecturer in King Faisal University, Alhassa, Kindom of Saudi Arabia. He has published various papers in international and national reputed journals. He has worked on research projects and published articles in conferences and prestigious research forums.

SULTAN H. ALMOTIRI Sultan H. Almotiri received the B.Sc. degree (Hons.) in computer science from King Abdulaziz University, Saudi Arabia, in 2003, the M.Sc. degree in internet, computer, and system security from Bradford University, U.K., in 2006, and the Ph.D. degree in wireless security from Bradford University. He was the Chairman of the Computer Science Department, Umm Al-Qura University, Saudi Arabia, and the Vice Dean of eLearning and Distance Education with Umm Al-Qura University. He is currently the Chief Cyber Security Officer with Umm Al-Qura University and an Assistance Professor with the Computer Science Department, Faculty of Computer and Information Systems, Umm Al-Qura University. His research interests include cyber security, cryptography, AI, machine learning, eHealth, eLearning, the IoT, RFID and wireless Sensors, and image processing.

MOHAMMED A. ALGHAMDI received the B.Sc. degree in computer science from King Abdulaziz University, Saudi Arabia, in 2006, and the master's and Ph.D. degrees in computer science from the University of Warwick, U.K., in 2008 and 2012, respectively. He is currently an Assistant Professor at Umm Al-Qura University, Saudi Arabia. He has also published a number of good quality journal papers in the signal and image processing domain and various conference papers. His research interests include wireless networks and 4G/5G networks.