TRUST LEVEL EVALUATION BASED ASYMMETRIC CRYPTOGRAPHY PROTOCOL FOR FLEXIBLE ACCESS CONTROL IN FOG COMPUTING

C. Nagarani[1] and R. Kousalya[2]

[1]Assistant Professor, PSG College of Arts and Science, Coimbatore, Tamil Nadu, India
[2]Professor and Head, Department of Computer Applications,
Dr. N.G.P Arts and Science College, Coimbatore, Tamil Nadu, India

ABSTRACT

The foremost problems in the fog-enabled cloud computing model are security guarantees and data Access Control (AC) because of the imitation of data by invaders. To enhance the security of this system, an Extended Communication Latency-based Authentication Scheme (ECLAS) that solves the mobility and similar locality legitimate login failures via applying two-factor authentication and a keystroke dynamics computation with obfuscated Round Trip Latency (RTL) of each users. But, the data need to accessed by other user should fulfill an be expected authentication and defend against dishonest access or login. So, data AC at cloud or fog nodes is greatly essential in many applications of fog-enabled cloud systems. Therefore in this article, a Flexible AC (FAC) protocol is introduced with the ECLAS for controlling the data access in fog-enabled cloud systems according to the trust estimated by the user in the cloud and reputations created by the amount of fog nodes in a flexible way via applying the Elliptic Curve Cryptography (ECC) and Proxy Re-Encryption (PRE). In this scheme, multi-dimensional controls are proposed on cloud and fog data access according to the strategies set by the user. The user encrypts its information with asymmetric secret key and this key is split into many segments for supporting different control policies. So, the user encrypts various segments of secret key with different encryption keys which are accordingly handled by the user and an amount of fog nodes regarding various reputation characteristics in different scenarios. Then, the user or fog nodes manage the data access using data encryption by the user. Finally, the experimental results exhibit the effectiveness of the proposed FAC as compared to the state-of-the-art AC schemes.

KEYWORDS

Fog computing, Cloud computing, Extended CLAS, Access Control, Trust, Reputation, Elliptic curve encryption.

1. INTRODUCTION

Fog computing is typically a decentralized paradigm to process and accumulate the data between the origin and a cloud structure. Based on this paradigm, the necessity of processing and accumulating a vast amount of unwanted information is prevented. As a result, the data transfer overhead is minimized and the efficiency is maximized. Chiefly, this is motivated via the significant development of Internet-of-Things (IoT) systems. But, there are many issues in scalability and consistency because of high overload in the data server while a common client-server model is accounted. Such issues are tackled by this novel framework which offers the accessible decentralized result. It is realized via the novel hierarchically shared fog paradigm between the cloud and end-user systems [1-3].

Normally, a fog device has the minimum computing resources for data storage and a huge facility for data processing. Because of fewer necessities of resources, it has a high difficulty for executing the entire group of security solutions to identify and defend the attacks. But, there is no perfect security guarantees and estimations for the fog-enabled cloud paradigm. Likewise, verification and agreement solutions are not applicable because fog systems are functioning at the network edges. The configuration of fog systems may practice with many threats which are not accessible in the cloud.

In general, fog systems consist of different categories of connectivity to the secured cloud server to allocate the verification information and gather audit logs. But, it is estimated in specific configurations like smart grid. Perhaps, a verification scheme like secluded verification dial in client service or lightweight file AC over this connectivity is not recognized [4]. Also, trustworthiness on cloud central verification servers is uncertain as verification should extend for client systems locally while secluded verification server transfers are missed. A criterion for ensuring the fundamental access is essential although it defines via transferring normal AC, however with an audit test. Most of the successful attacks employ verification recommendations. Secret codes are mostly not trustworthy; however it is the key verification recommendation for network facilities. Conversely, invaders regularly revolutionize methods for negotiating the secret codes. The issues in those secret codes are overcome via multifactor verification approaches [5]. It needs typically other verification ciphertexts with standard secret code to login. But, it bears from many limitations and novel vulnerabilities. The invader can simply present the client with a fake unreliable monitor to input his/her additional ciphertexts which the assailant used are synchronized to imitate the legitimate clients. So, CLAS was suggested [6] to improve the safety of standard multifactor verification approaches via leveraging the RTL between users and verifiers. It accounts RTL including the standard authorizations of users and applies them for defending against secret code negotiation. Besides, it limits the login to profiled localities when stimulating additional information for non-profiled ones which highly minimizes the attacks even the legitimate recommendations were negotiated. But, it needs a combination of extra profiling characteristics for achieving high robustness against secret code compromise.

For this reason, an ECLAS was developed to taken into consider the mobility and identical locality issues in the CLAS. First, a 2-factor verification method was combined with the CLAS for controlling mobility and legitimate login failures [7]. In ECLAS, two kinds of mobility patterns: selective and arbitrary were taken into account. In selective mobility scenario, an individual profile for every locality was created by the CLAS to permit access via evaluating his/her real-time login profile with any of the accumulated reference profiles. In an arbitrary mobility scenario, 2-factor verification was combined with the CLAS. Besides, utility metric-based locality anonymization and obfuscation of RTL mechanisms were suggested for enhancing the security against mimic attacks. According to these mechanisms, the client's localities were anonymized and the RTL values were obfuscated that preserves the client compromise events from imitating the RTL via receiving closer to the client locality. Moreover, keystroke dynamics estimation was applied with obfuscated RTL for effectively defending similar locality attacks. Though it achieves better anonymity, AC to information at cloud or fog nodes is highly vital in several scenarios of fog-enabled cloud systems. Since the data need to be accessed by other entities to satisfy an expected service e.g., an e-Healthcare system and defended against distrusted or malicious access to data or login information.

Hence in this paper, a FAC protocol is proposed with the ECLAS to manage the data access in fog-enabled cloud environment. This protocol is based on the trust computed by the client in the cloud and reputations made by the number of fog nodes in a flexible way via applying the ECC and PRE. Normally, Attribute-Based Encryption (ABE) is broadly applied in FAC strategy, but it increases the computational cost linearly with the number of attributes in FAC model.

This indicates that the ABE is not appropriate for end - users with limited energy resources. Therefore, in this paper, ECC with PRE is utilized to reduce the computational cost with the number of attributes used in FAC. In this FAC protocol, multi-dimensional controls are applied on cloud and fog data access according to the strategies set by the client. The client encrypts its information using the asymmetric secret key which is partitioned into several segments to handle different control policies. As a result, the client encrypts various segments of secret key using different encryption keys which are accordingly handled by the user and the number of fog nodes regarding various reputation characteristics in different scenarios. Then, the user or fog nodes manage the data access using data encryption by the user. Thus, this protocol can help to enhance the AC in fog-enabled cloud environments with a high anonymity against different attacks efficiently and flexibly.

2. LITERATURE SURVEY

Ren et al. [8] suggested an approach called Fine-grained and Flexible Access Control (F2AC) for enhancing the file storage in mobile cloud computing. In F2AC, a novel AC scheme such as a directed tree with the connected leaf framework was applied for generating, incorporating and removing customers in a group. Also, the customers were authorized as a group leader who can verify privileges to other group customers iteratively. Moreover, different AC principles were described since customers require and isolate AC schemes to shorten customer knowledge and control flows in the cloud. But, the customer with verified privileges cannot able to include additional non-additive customers for current records.

Li et al. [9] developed an effective secure and revocable Multi-Authority AC System (MAACS) in the cloud data storage. In MAACS, a novel Multi-Authority Ciphertext-Policy ABE (MA-CP-ABE) mechanism with decryption outsourcing was presented. The decryption overhead for clients was removed via outsourcing the unwanted bilinear pairing functions to the cloud servers. Also, an efficient attribute-level client revocation method was developed. On the contrary, it has high computation cost during decryption.

Li et al. [10] recommended a FAC via applying CP-ABE on encrypted information for mobile clients in hybrid cloud computing. In this method, the actual decryption keys were partitioned into different keys: control, secret and a group of transformation keys. The secluded cloud is controlled by the group supervisor who updates the transformation keys via the control key. Also, the condition of FAC and attribute was changed. As well, the mobile client's single private key remains unaltered including the ciphertext even if the data client's attribute was removed. Moreover, the AC list was altered via inserting the attributes with respective control key and transformation keys. However, it has a high computation burden for keys and data management.

Ahuja & Mohanty [11] introduced a scalable attribute-based AC method with flexible delegation cum distribution of access privileges for cloud data storage. This method was based on the extension of the CP-ABE for achieving the flexible delegation of access privileges and distributed access privileges including the scalability and FAC through decentralizing the key issuing authority at various levels of hierarchy i.e., low-level clients acquire secret keys from the high-level clients in the hierarchy. Nonetheless, the encryption and decryption time was high while increasing the number of leaf nodes of access tree. An attribute-based AC method with controlled access delegation [12] was suggested to provision the multi-level access delegation with every delegating client. Also, it has the ability to manage additional delegations via the delegate. Further, this method was extended to limit the total amount of delegations which allowable for a given attribute to accommodate for unsuitable client attributes. By combining the proper trust model, the trustworthiness of access decision- making was enhanced. However, it has high computational complexity.

For secure and FAC, a fine-grained HER-AC method [13] was recommended in which the data owner generates ciphertexts prior to learning data and access policies. Also, the resultant ciphertext and access policies were obtained through the online learning. But, it is not able to simultaneously execute encryption, decryption and key generation through offline. ShareHealth was designed [14] to achieve the secure sharing of mHealth data streams. First, cryptographically-enforced-AC measures were considered and the transient existence of mHealth data streams was acknowledged. Also, few data streams were revoked to access them. But, it has high computational complexity.

Miao et al. [15] presented the Lightweight Fine-Grained ciphertexts Search (LFGS) method in fog computing via improving CP-ABE and Searchable Encryption (SE) for achieving FAC and keyword search simultaneously. In LFGS, the partial computational and storage overhead were shifted from end - users to elected fog nodes. Moreover, the fundamental LFGS was enhanced for maintaining the conjunctive keyword search and attribute update to evade returning inappropriate search outcomes and illicit accesses. But, it has high communication complexity.

Daoud et al. [16] developed a security framework according to the cooperation between IoT and fog. The primary task of this model was applying a distributed AC according to the security resource control model for fog-IoT systems and proactive security mechanisms under ultra-trustworthiness including low-latency limits. In this framework, an efficient AC task was integrated related to the forecasting method for ensuring secure communication between different resources and functional units. Also, a complete scheduling task and resource assignment method were proposed for enhancing the intended efficiency of the system. Conversely, it has high overhead due to the splitting of data into many blocks.

3. PROPOSED METHODOLOGY

In this section, the proposed FAC protocol is described in detail. Consider a system involving three categories of units: client, server and Stealthy Relay (SR). At first, the CLAS is performed to authenticate each client's login details. If the client's login information is authenticated, then this FAC protocol is executed during data access i.e., if the client requests to access the information of another client, then FAC protocol is performed to handle the accessibility of data that provides an expected service to the client. Every client must handle their data access for enhancing the flexibility of the fog-cloud system.

To attain trusted data access and prevent possible threats in fog-cloud storage services, this FAC protocol must accomplish different objectives:

- Safety measure: The information with number of attributes accumulated in the server can only be accessed via appropriate system units that are sufficiently honest; the control of information access is based on trust and reputation with the minimum threats.
- Heterogeneity: The FAC protocol can help different data access requirements. It can manage data access directly via the client and one or more SRs indirectly while the client is not available or cannot make an access decision.
- Flexibility: The FAC protocol can be flexibly used for satisfying various access control situations, policies and strategies.
- Lightweight: It manages cloud data access with the least computation and communication overhead.

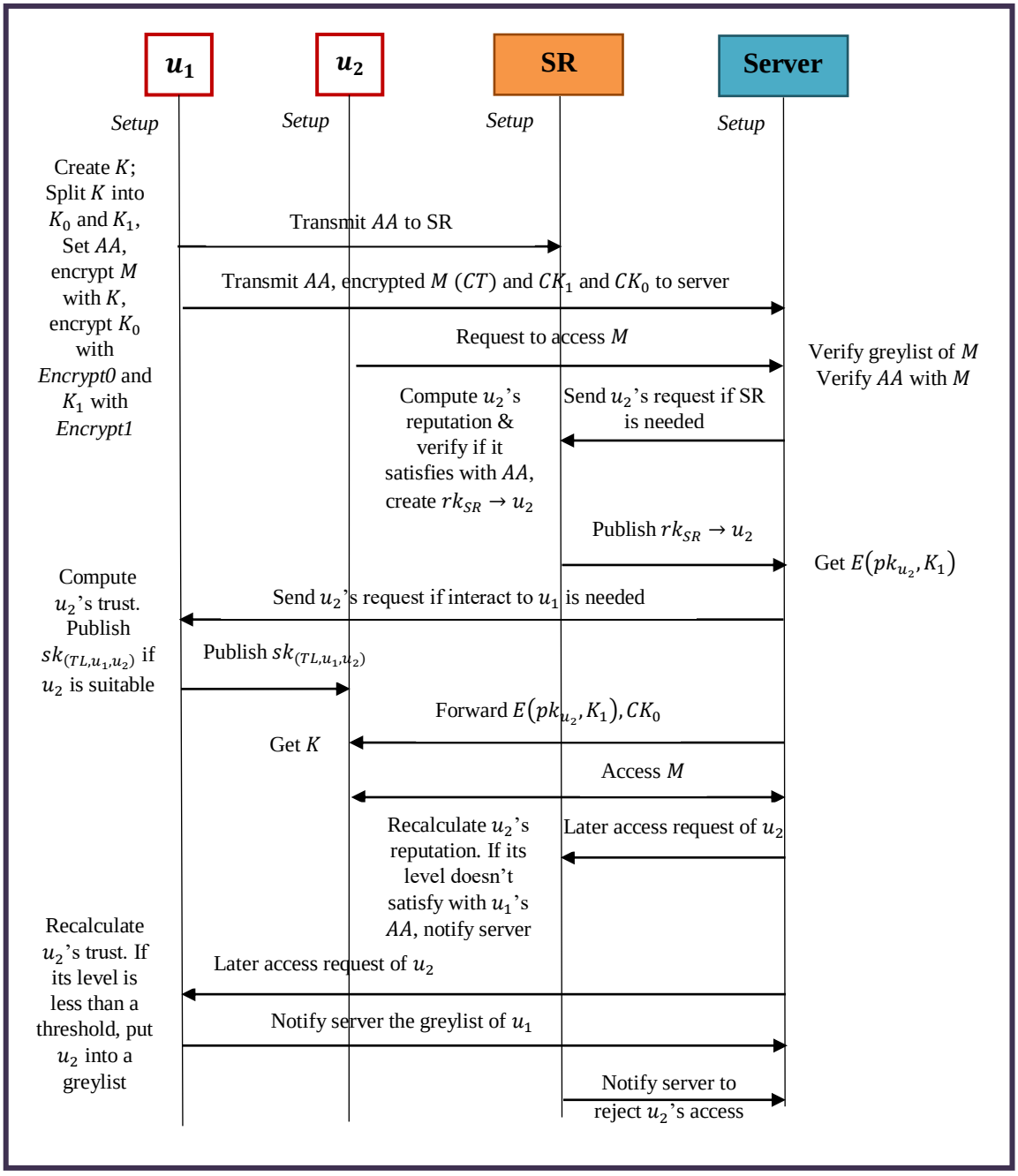

Figure 1. Process of Proposed FAC Protocol

The FAC protocol encompasses an amount of basic processes depicted in Figure 1 are explained below.

Setup*:* It takes the implicit safety factor 1^kas input. It selects a bilinear multiplicative set $\mathbb{S}$ of prime order p with generator g and a pairing $\alpha: \mathbb{S} \times \mathbb{S} \rightarrow \mathbb{S}_T$. Then, it picks a random point $\mathcal{P} \in \mathbb{S}$and a random exponent $\gamma \in \mathbb{Z}_p$. After, it results the public key $PK = \{\mathbb{S}, \mathbb{S}_T, e, g, \mathcal{P}, e(g,g)^{\gamma}\}$ and the master key $MK = g^{\gamma}$. This task is carried out at the client system or a trusted client agent. In the meantime, every fog node (SR) creates its public and private keys pk_{SR} and sk_{SR} for the determination of PRE.

ECCClientKeyGeneration(PK, MK, u)***:*** It considers the public key PK, the master key MK and a unique client identity u. It selects a random secret $mk_u \in \mathbb{Z}_p$and results a public client key $pk_u = g^{mk_u}$ that is utilized for distributing secret attribute keys for u and a secret client key$mk_u = MK \cdot \mathcal{P}^{mk_u} = g^{\gamma} \cdot \mathcal{P}^{mk_u}$ used to decrypt the ciphertexts. As well, it selects a hash function $\mathcal{H}_{sk_u}: \{0,1\}^* \rightarrow \mathbb{Z}_p$ equally and randomly from a finite set of hash functions. This task is performed at the client system or an honest client agent.

PREClientKeyGeneration(u)***:*** It creates public key $pk_u$and private key sk_u for PRE.

$$pk_u = (Z^{a_1}, g'^{a_2}), \qquad sk_u = (a_1, a_2)$$

The system parameters are random generators $g' \in \mathbb{S}, Z = e(g, g) \in \mathbb{S}_T$ and $a_1, a_2 \in \mathbb{Z}_p \cdot pk_u$ is applied to create the re-encryption key at SR for u. This task is performed at the client system or a trusted client agent.

GenerateEncryptionKey(): It creates a asymmetric key K for data encryption and is executed by the client. In this experiment, ECC is used.

SplitKey(K, n)***:*** It splits the input K into $n + 1$ segments where $n \geq 0$.

MergeKey$(K_0, \dots, K_n)$*:* It aggregates partial keys $(K_0, \dots, K_n)$ for obtaining the complete key K.

GenerateIndividualTrustPK(PK, TL, sk_u)*:* It is carried out by the client system whenever u would request to manage the access of its information according to the trust computation. It verifies the Trust Level (TL) related strategies. If this is the situation, this task results a public attribute key of the TL for u represented as $pk_{(TL,u)}$ which has 2 elements:

$$pk_{(TL_i,u)} =< pk_{(TL_i,u)'} = g^{\mathcal{H}_{sk_u}(TL_i)},$$
$$pk_{(TL_i,u)''} = e(g, g)^{\gamma \mathcal{H}_{sk_u}(TL_i)} >$$

Or else, it results NULL. Observe that $pk_{(TL,u)} = \{(pk_{(TL_i,u)})\}, (i \in [0, TL_{max}])$ where TL_{max} denotes the highest level of TL.

DistributeIndividualTrustSK$(PK, TL, sk_u, pk_{u'})$***:*** It is carried out at the client system via verifying the admissibility ofu'. It verifies whether u' with public key $pk_{u'}$ is appropriate of the attribute TL i.e., it authenticates in which trust level u' is positioned. If u' is situated in the trust level $\mathcal{V}_{TL}$ which is an integer and$\mathcal{V}_{TL} \in [0, TL_{max}]$. Note that u' is appropriate for attribute $TL_i, (i \leq \mathcal{V}_{TL})$. After, it results a secret TL key $sk_{(TL,u,u')}$ for client u'.

$$sk_{(TL_i,u,u')} = pk_{u'\mathcal{H}_{sk_u}(TL_i)} = g^{mk_{u'}\mathcal{H}_{sk_u}(TL_i)}, (i = \mathcal{V}_{TL})$$

Or else, it gives NULL.

Encrypt$\left(PK, K_0, AA, pk_{(TL,u)}\right)$*:* It considers the partial key K_0 and the public keys $pk_{(TL,u)}$ as input related to the individual trust occurring in the data access strategy AA of u. It encrypts K_0 with the policy AA and results the cipher-keyCK_0.The access strategy AA is represented as $AA = \mathcal{V}_{j=1}^{m} TL_j$ where m is the amount of chosen TL. Here, TL_j is an individual trust level set by u for controlling the access and $TL_j = TL_i$ refers that u provides the clients with TL_i the ability for decrypting the cipher-key. The *Encrypt0* task iterates over all $j = 1, \dots, m$. It creates a random value$R_i \in \mathbb{Z}_p$, for every needed TL level TL_i in the policy and constructs CK_0^i as:

$$CK_0^i = \left\langle \begin{array}{c} E_i = K_0 \cdot pk_{(TL_i,u)''R_i} \\ E_i' = P^{R_i}, \\ E_i'' = pk_{(TL_i,u)'R_i} \end{array} \right\rangle$$

This task is carried at the client system. The client distributes the result of this task including its encrypted information to the server.

Decrypt$\left(PK, AA, CK_0, sk_{u'}, sk_{(TL,u,u')}\right)$*:* It considers the cipher-key generated by the *Encrypt0* task and a key ring $sk_{u'}, sk_{(TL,u,u')}$ as input foru'. It decrypts the cipher-key CK_0 and results the respective plain key K_0 if the attribute is adequate to fulfill the strategy AA applied for encryption.

$$K_0 = E_i \cdot \frac{e\left(E_i', sk_{(TL_i,u,u')}\right)}{e\left(E_i'', sk_{u'}\right)}$$

Or else, it results NULL. It is simple for authentication that the decryption is accurate. Consider $a_i : \mathcal{H}_{sk_u}(TL_i)$. After, $E_i = K_0 \cdot e(g,g)^{\gamma a_i R_i}, E_i'' = g^{a_i R_i}$ and

$$E_i \cdot \frac{e\left(E_i', sk_{(TL_i,u,u')}\right)}{e\left(E_i'', sk_{u'}\right)} = K_0 \cdot e(g,g)^{\gamma a_i R_i} \cdot \frac{e\left(P^{R_i}, g^{mk_{u'} a_i}\right)}{e\left(g^{a_i R_i}, g^{\gamma \cdot P^{mk_{u'}}}\right)}$$
$$= K_0 \cdot e(g,g)^{\gamma a_i R_i} \cdot \frac{e(P,g)^{R_i mk_{u'} a_i}}{e(P,g)^{R_i mk_{u'} a_i} \cdot e(g,g)^{\gamma a_i R_i}} = K_0$$

ReencryptionKeyCreation$(pk_{SR}, sk_{SR}, pk_{u'})$*:* It results $rk_{SR} \to u' = g'^{b_2 a_1} = pk_{u'}^{a_1}$ where a_1 is segment of sk_{SR} and b_2 is segment of $sk_{u'}$. On input pk_{SR}, sk_{SR} and $pk_{u'}$, it creates the re-encryption key $rk_{SR} \to u'$ for u' if it fulfills the access strategy of the client according to the new reputation computation at SR. After, the SR transmits $rk_{SR} \to u'$ to the server.

Encrypt1(pk_{SR}, K_n)*:* The client encrypts its partial secret key $K_n (n \geq 1)$ via the public key of SR for acquiring the encrypted K_n by pk_{SR} represented as:

$$E(pk_{SR}, K_n) = (g'^x, K_n Z^{a_1 x})$$

Here, Z^{a_1} is segment of pk_{SR} and $x \in \mathbb{Z}_p$. The client distributes $E(pk_{SR}, K_n)$ including its encrypted information to the server.

RE$\left(rk_{SR} \rightarrow u', E(pk_{SR}, K_n)\right)$*:* If u' is permitted to access the information, the server executes

$$RE\left(rk_{SR} \rightarrow u', E(pk_{SR}, K_n)\right) = E(pk_{u'}, K_n) = (Z^{b_2 a_1 x}, K_n Z^{a_1 x}) = CK_n$$

Also, it offers it tou'. Client u' decrypts $E(pk_{u'}, K_n)$ via its private key $sk_{u'}$ for acquiringK_n. In this FAC protocol, server performs as the proxy in terms of PRE. It shares the partial secret key K_n indirectly to the trusted i.e., certified clients without any knowledge about these secrets.

Decrypt$\left(sk_{u'}, E(pk_{u'}, K_n)\right)$*:* It considers the cipher-key generated by the *RE* task and $sk_{u'}$ as input. As well, it decrypts the cipher-key $E(pk_{u'}, K_n)$,

$$K_n = \frac{K_n Z^{a_1 x}}{(Z^{b_2 a_1 x})^{\frac{1}{b_2}}}$$

Encrypt(K, M)***:*** It considers K and information $M$with number of attributes as input for obtaining the encrypted information CT. The client distributes CT to the server. In this experiment, ECC is used.

Decrypt(CT, K)***:*** It considers a cipher-text CT generated by the *Encrypt* task and the full encryption key K as input to get the plaintext M.

FAC Protocol:
Assume that client 1 (u_1) accumulated its confidential private information at server when client 2 (u_2) wants to access it with the verification of u_1 and one SR.

Begin

Initialize the system via requesting *Setup;*

Create an encryption key *K*by u_1;

Splits$K$ into$K_0$and K_1;

Encrypts information *M*with number of attributes using the secret key *K* for obtaining CT;

Creates the data access strategy AA related to individual trust level threshold, public reputation threshold to access *M*;

Upload the encrypted information CT, strategy AA and encrypted $K_1(CK_1)$ by u_1 via executing *Encrypt1* and encrypted $K_0(CK_0)$ via applying *Encrypt0* to server;

Transmits AA to SR byu_1;

$u_2$desiresto access u_1's information via requesting server;

The server authenticates the validity of its ID and the set of encrypted *K* to choose if transmitting this request to u_1 and/or SR if it is not in the greylist;

The server confirms whether to interact u_1 and/or SR according to the data in AA;

$\boldsymbol{if}(SR\ or\ u_1\ is\ interacted)$

SR computes u_2's reputation and verifies if it fulfills with M's access strategy AA;

if $(access\ is\ permitted)$

SR creates and distributes $rk_{SR} \rightarrow u_2$ to the server that re-encrypts the CK_1 to obtain $E(pk_{u_2}, K_1)$;

else

u_1 verifies the admissibility of u_2 for creating the adapted secret key $sk_{(TL,u_1,u_2)}$ for u_2 to decrypt CK_0;

u_1 distributes $sk_{(TL,u_1,u_2)}$ to u_2;

end if

end if

The server permits u_2 for accessing the requested information via corresponding encrypted information CT and encrypted keys (CK_1 and CK_0) to u_2;

u_2 decrypts CK_1 and CK_0 with the distributed secret keys from u_1 and its private key sk_{u_2};

u_2 obtains the complete K via merging K_1 and K_0 to decrypt CT and obtain M;

u_1 recalculates the trust according to the earlier and newly stored knowledges with regard to the information access configuration;

If u_2 has been distributed the secret keys and is not appropriate currently, then u_1 will put them into its actual information access greylist and notify the server;

SR recalculates the reputation of various users according to the newly gathered information;

If SR represents that u_2 does not fulfil with access strategy AA, SR will notify the server to block u_2' access to u_1's information;

End

Observe that the greylist is data-oriented because various information access may request several trust levels. Its information is updated in a dynamic manner according to the timely trust and reputation computation.

4. SIMULATION RESULTS

In this section, the FAC-ECC protocol is implemented in JAVA in which the client identities are accumulated in the Microsoft Access Database. Also, the SR functions are executed via configuring and deploying the PC [6]. In this experiment, 175 clients are taken into consideration for creating a client profile with a number of attributes. For example, consider the Electronic Health Record (EHR) files in the field of healthcare systems. In this system, the clients include doctors, nurses, pharmacists, therapist, etc., whereas the attributes include the role of a doctor, the time of creation of an EHR, origin(locality) of an EHR, EHR sensitivity, etc. An EHR with the highest sensitivity level can only be accessed by the doctor who generated the file and at the same locality as the origin of the EHR. The FAC protocol is implemented while u_2 (e.g., junior doctor) requests to access the confidential data stored at the server by u_1. This process authenticates whether it controls the data accessibility and confidentiality or not. Moreover, the performance is compared with the FAC-CP-ABE [10] in terms of time costs needed for encryption, decryption of ECC keys including the executing time of PRE functions.

4.1. Encryption Time

It is the time cost to process encryption functions using ECC keys with regard to various individual TLs.

Figure 2 depicts the encryption time (in milliseconds) of FAC-CP-ABE and FAC-ECC protocols with a varied amount of attributes. The FAC-ECC achieves 6.61% less encryption time compared to the FAC-CP-ABE protocol while using 25 attributes. This analysis indicates that the FAC-ECC protocol achieves less time to encrypt the plaintext for security during data access as compared to the FAC-CP-ABE. The encryption time is varied for various individual TLs requested to access the data and handle the number of attributes or reputation characteristics in different situations.

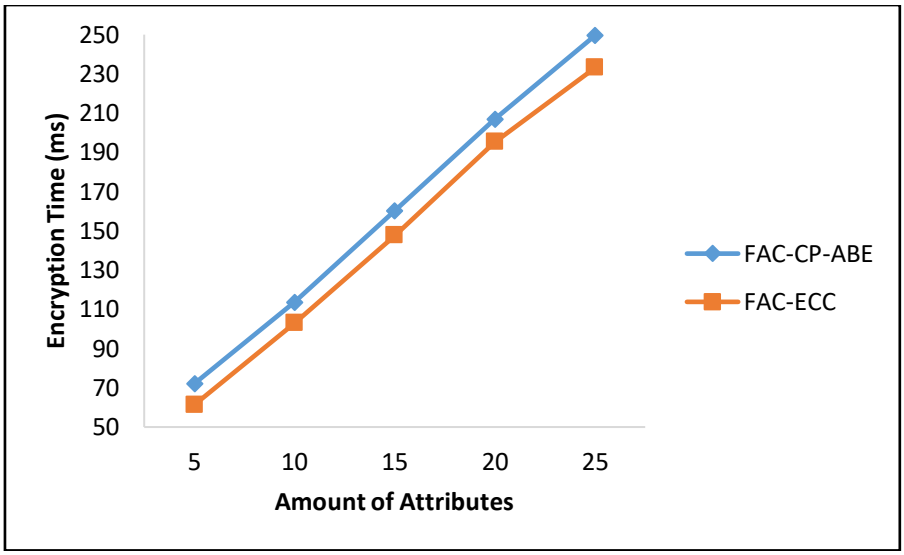

Figure 2. Encryption Time vs. Amount of Attributes

4.2. Decryption Time

It is the time needed to process decryption functions regarding to various authenticated individual TLs.

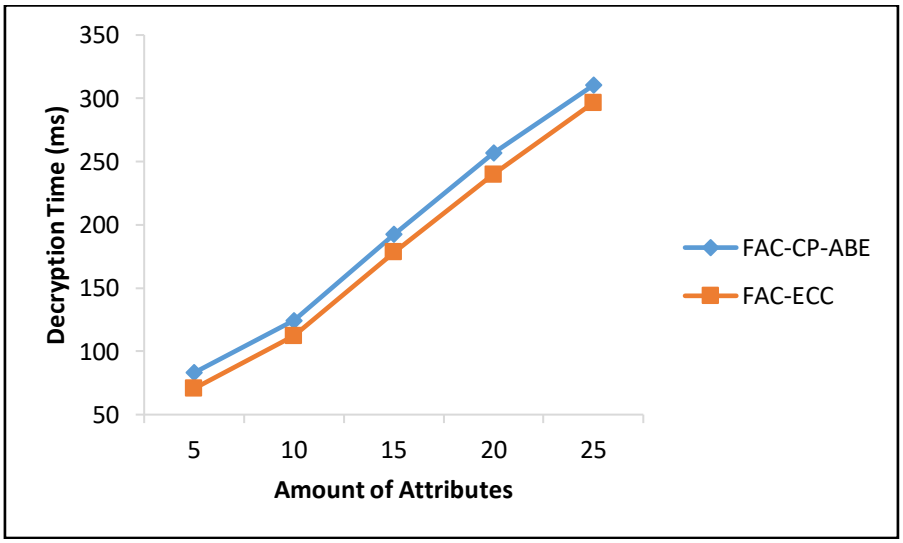

Figure 3. Decryption Time vs. Amount of Attributes

The decryption time (in milliseconds) of FAC-CP-ABE and FAC-ECC protocols under different amounts of attributes is portrayed in Figure 3. The FAC-ECC achieves a 4.45% less the decryption time compared to the FAC-CP-ABE protocol while using 25 attributes. This scrutiny observes that the FAC-ECC protocol minimizes the cost during decryption process i.e., FAC-ECC reduces the time to decrypt the ciphertext with the help of the amount of attributes or reputations in various scenarios while different individual TLs needed to access the information as compared to the FAC-CP-ABE.

4.3. Execution Time

It is total time cost for all the tasks including encryption, decryption and re-encryption for accessing data based on different authenticated TLs.

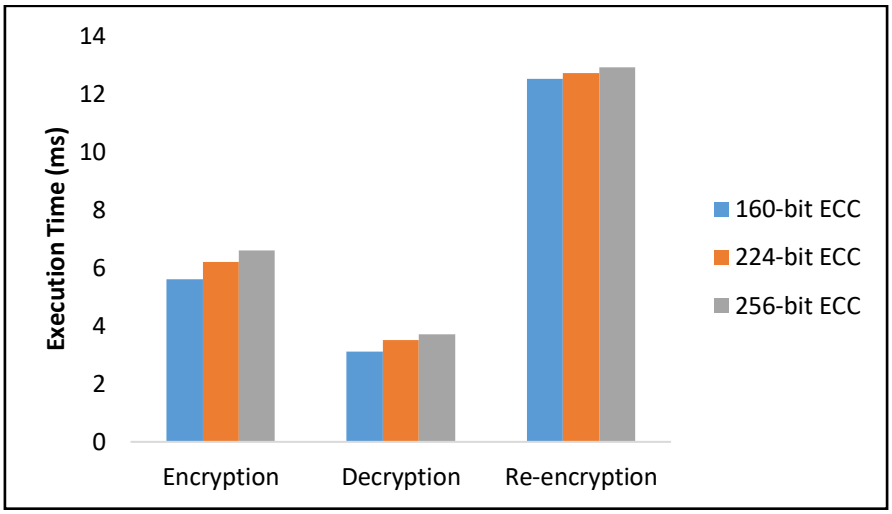

Figure 4. Execution Time of PRE Functions

Figure 4 exhibits the execution time (in milliseconds) of PRE functions including encryption, decryption and re-encryption. The FAC-ECC achieves a 6.6ms of encryption time, 3.7ms of decryption time and 12.9ms of re-encryption time when applying 256-bit ECC compared to 160 and 224-bit ECCs. It indicates that the PRE functions are not affected by the different sizes of ECC keys. It might help the client for deciding the appropriate sized asymmetric key for fulfilling its security demands. Since a client may access cloud services or information from other clients several times in a similar scenario, neglecting the re-encryption key creation may highly enhance the efficiency and feasibility of the F-ECC protocol.

4.4. Security Level

It refers to the level of security attained by the FAC-ECC and FAC-CP-ABE protocols against mimic and same location attacks.

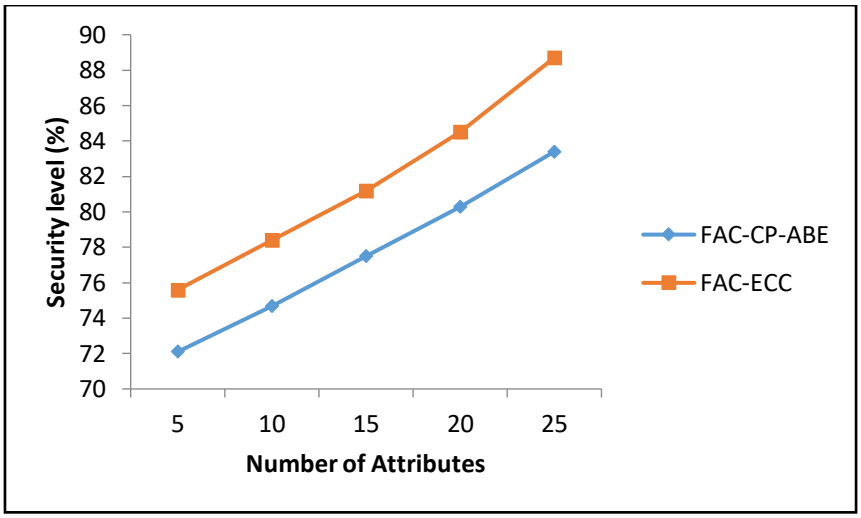

Figure 5. Security Level vs. Amount of Attributes

Figure 5 depicts the security level (in %) of FAC-CP-ABE and FAC-ECC protocols with a varied amount of attributes. The FAC-ECC achieves a 6.35% higher security compared to the FAC-CP-ABE protocol while using 25 attributes. This analysis indicates that the FAC-ECC protocol achieves a higher security against mimic and same location attacks compared to the FAC-CP-ABE by selecting the proper asymmetric key which satisfies the security requirements for accessing the data.

5. Conclusions and Future Work

In this paper, a FAC protocol is suggested including the ECLAS to handle the data access in fog-enabled cloud systems on the basis of trust computed by the clients in the cloud system and reputations produced by the number of fog nodes in a flexible manner by applying the ECC and PRE. In this FAC, multi-dimensional controls are utilized on cloud and fog data access based on the policies assigned by the clients. The client encrypts its data using the asymmetric a secret keys which is partitioned into several segments to support various control strategies. Therefore, the client encrypts different segments of secret key with several encryption keys which are accordingly controlled by the client and a number of fog nodes with regard to different reputation characteristics in various situations. Moreover, the client or fog nodes control the data access via data encryption by the client. To end, the experimental results proved that the proposed FAC protocol achieves better efficiency in terms of time required for encryption, decryption and re-encryption as compared to the existing AC schemes. Some of the future work directions include: to prevent sensitive information in the cloud storage when the permitted users in the cloud have the behavior of dynamic change.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] Kunal, S., Saha, A., & Amin, R. (2019). An overview of cloud fog computing: architectures, applications with security challenges. Security and Privacy, 2(4), 1-14.

[2] Zhang, P., Zhou, M., & Fortino, G. (2018). Security and trust issues in fog computing: a survey. Future Generation Computer Systems, 88, 16-27.

[3] Naha, R. K., Garg, S., Georgakopoulos, D., Jayaraman, P. P., Gao, L., Xiang, Y., & Ranjan, R. (2018). Fog computing: survey of trends, architectures, requirements, and research directions. IEEE Access, 6, 47980-48009.
[4] Khan, S., Parkinson, S., & Qin, Y. (2017). Fog computing security: a review of current applications and security solutions. Journal of Cloud Computing, 6(1), 1-22.
[5] Sun, H. M., Chen, Y. H., & Lin, Y. H. (2011). oPass: a user authentication protocol resistant to password stealing and password reuse attacks. IEEE Transactions on Information Forensics and Security, 7(2), 651-663.
[6] Dou, Z., Khalil, I.,& Khreishah, A. (2017). CLAS: a novel communications latency based authentication scheme. Security and Communication Networks, 2017, 1-20.
[7] Nagarani, C., & Kousalya, R. (2019). Round trip latency based authentication scheme in fog-enabled cloud computing system. International Journal of Recent Technology and Engineering, 8(2), 1270-1278.
[8] Ren, W., Zeng, L., Liu, R., & Cheng, C. (2016). F2AC: a lightweight, fine-grained, and flexible access control scheme for file storage in mobile cloud computing. Mobile Information Systems, 2016, 1-22.
[9] Li, Q., Ma, J., Li, R., Liu, X., Xiong, J., & Chen, D. (2016). Secure, efficient and revocable multi-authority access control system in cloud storage. Computers & Security, 59, 45-59.
[10] Li, W. M., Li, X. L., Wen, Q. Y., Zhang, S., & Zhang, H. (2017). Flexible CP-ABE based access control on encrypted data for mobile users in hybrid cloud system. Journal of Computer Science and Technology, 32(5), 974-990.
[11] Ahuja, R., &Mohanty, S. K. (2017). A scalable attribute-based access control scheme with flexible delegation cum sharing of access privileges for cloud storage. IEEE Transactions on Cloud Computing, 14(8), 1-14.
[12] Pussewalage, H. S. G., & Oleshchuk, V. A. (2017). Attribute based access control scheme with controlled access delegation for collaborative E-health environments. Journal of Information Security and Applications, 37, 50-64.
[13] Liu, Y., Zhang, Y., Ling, J., & Liu, Z. (2018). Secure and fine-grained access control on e-healthcare records in mobile cloud computing. Future Generation Computer Systems, 78, 1020-1026.
[14] Greene, E., Proctor, P., & Kotz, D. (2018). Secure sharing of mHealth data streams through cryptographically-enforced access control. Smart Health, 12, 49-65.
[15] Miao, Y., Ma, J., Liu, X., Weng, J., Li, H., & Li, H. (2018). Lightweight fine-grained search over encrypted data in fog computing. IEEE Transactions on Services Computing, 12(5), 772-785.
[16] Daoud, W. B., Obaidat, M. S., Meddeb-Makhlouf, A., Zarai, F., & Hsiao, K. F. (2019). TACRM: trust access control and resource management mechanism in fog computing. Human-centric Computing and Information Sciences, 9(1), 1-18.

AUTHORS

C. Nagarani received the M.Sc. Computer Science degree from Periyar University, India in the year 2004 and M.Phil degree in Computer Science from Periyar University, India in the year 2007 respectively. Currently, she is an Assistant Professor of Computer Science, PSG College of Arts and Science, Coimbatore, affiliated with Bharathiyar University. She has a total experience of over 15 years. She has published 4 papers in Journals. Her area of interest is network security.

Dr. R. Kousalya received the B.Sc. degree in Physics from P.S.G.R. Krishnammal College for Women, India in the year 1997, the MCA degree in Computer Applications from Bharathiar University in the year 2000, the M.Phil degree in Computer Science from Manonmaniam Sundaranar University in the year 2003 and the Ph.D degree in Computer Applications from Manonmaniam Sundaranar University in the year 2016. Currently, she is a Head of Department/Professor of Computer Application, Dr. N.G.P Arts and Science College, Coimbatore, affiliated to Bharathiyar University. She has a total experience of over 19 years. She has published 33 papers in Conference and 11 papers in Journal. Her area of interest includes data mining and web mining.