IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

A Lightweight Policy Update Scheme For Outsourced Personal Health Records Sharing

PG Scholar
Department of Computer Applications
Madanapalle Institute of Technology & Science, India

Guide name:Dr syed Siraj Ahmed N
Assistant Professor
Department of Computer Application

Abstract:

Several healthcare providers use electronic personal health records (PHRs) to enable individual patients to handle their own health data in a resilient and scalable environment, thanks to the high flexibility and accessibility of data outsourcing environments like cloud computing. PHRs, on the other hand, contain highly sensitive information about which security and privacy concerns are paramount. Furthermore, PHR owners should be able to create their own access policy for their outsourced data in a flexible and secure manner. Existing commercial cloud platforms typically include symmetric or public key encryption as an optional feature to enable data secrecy for its tenants, in addition to the basic authentication feature. However, due to the significant key management overhead of symmetric encryption and the high maintenance cost of maintaining multiple copies of ciphertext for public key encryption solutions, such traditional encryption algorithms are not ideal for data outsourcing environments. In this research, we design and construct a secure and fine-grained access control mechanism for outsourced PHRs that includes lightweight access policy updates. The ciphertext policy attribute-based encryption (CP-ABE) and proxy re-encryption are the foundations of our proposed approach (PRE). We also provide a policy versioning approach to support full policy change tracing. Finally, we conducted a performance evaluation to show that the proposed strategy is effective.

Keywords: PHRs, Access control, CP-ABE, Policy update, Proxy re-encryption, Policy versioning, Performance evaluation.

.

1. INTRODUCTION:

In a cloud storage system or other outsourced data sharing environment, the outsourced server must be available at all times to allow unrestricted access to shared data and services. Due to the cost savings and efficient resource management provided by cloud providers, many firms and individuals now prefer to store their valuable data on outsourced servers such as cloud storage. In terms of privacy and security, most data owners encrypt their information before sending it to a cloud server. Encrypting data is the most effective method of preventing unauthorised access to sensitive information. Nonetheless, encryption is insufficient to ensure strict security management. Another security perimeter that is commonly required is an access control system. Many works have used attribute-based encryption (ABE) [1] to overcome this issue. ABE's encryption approach is "one-to-many," with fine-grained access control. It also comes with encryption and access control features. Crypttext-policy attribute-based encryption (CP-ABE) [2] and key-policy attribute-based encryption (KPABE) are the two forms of ABE (KP-ABE). In the CP- The user's decryption key is built using attributes, and the data is encrypted using access policy. The user key is linked to the access policy in KP- ABE, and the encryption is handled by a set of attributes. CP-ABE is favoured in terms of security enforcement because the data owner can establish his or her own policy for encrypting the data. The benefits of CP-ABE include group key management [3]. Decoupling abstract qualities from actual keys is one of them. It lowers communication costs and allows for finer data access management. It also offers flexible one-to-many encryption rather than one-to-one encryption; it's seen as a promising solution for addressing the issue of secure fine-grained data sharing and decentralised access control. When an attribute is revoked or a policy is updated, CP-ABE adds expensive overheads like ciphertext re-encryption, key re-generation, and key re-distribution. Because the propagation effect on both the ciphertext and the user decryption key is high, these revocation and policy update procedures must be done with

caution. The calculation and transmission costs associated with key updating are very high when there are a large number of users. are exceptionally high. The data owner bears the cost of policy updates and data re-encryption, whereas the communication cost is determined by the amount of ciphertexts that must be downloaded and re-uploaded from and to the data outsourcing environment. Such overheads make real-world data sharing scenarios inefficient to implement. It's also possible that encryptors won't be available when the access policy needs to be updated. In this research, we show how to quickly update CP-ABE access controls without requiring the data owner to re-encrypt the data. In terms of PHR sharing, the data owner, such as a patient, can share their information with whoever they wish. We choose symmetric encryption for data encryption because it gives higher encryption performance, while the symmetric key is encrypted using the CP- ABE method to provide efficient encryption and improved data access and policy updating performance. Because we encrypt the symmetric key with the CP-ABE approach, the cost of updating the policy only affects the encrypted symmetric key. As a result, there is no need to re-encrypt all ciphertexts. This cuts the cost of computation on the proxy side dramatically. The following is a list of our contributions.

1. In a multi-authority data outsourcing scenario, we present an access control paradigm for PHRs with lightweight policy updates. When the policy is modified, the re-encryption operation is offloaded to the proxy, while the data owner deals with minimal computations, thanks to our cryptographic construction and the PRE approach. Two-step encryption is used to reduce costs for both the data owner and the proxy.
2. We offer a policy versioning mechanism that allows all update events to be thoroughly recorded and older versions of any policy to be reconstructed at any moment for full study.
3. All crypto operations in the PRE system are parallelized using parallel programming. When the policy is revised in our model, the system efficiently re-encrypts all ciphertexts affected by the new policy.
4. We conduct a security and performance analysis to demonstrate that our suggested method is both secure and efficient in practise.

The remainder of this work is laid out as follows. The background and related work are discussed in Section II. The background of CP-ABE is described in Section III. Our proposed system is shown in Section IV. The policy versioning technique is discussed in Section IV. The security analysis is presented in Section VI. The evaluation and experiment are covered in Section VII. The paper is finished with Section VIII.

2. RELATED WORKS:

One of the biggest overheads of CP-ABE is policy update, which degrades scalability and efficiency. A data owner updates the policy by adding, changing, or cancelling the attribute or logical gates (AND, OR, or M of N) in the policies. The data owner must first retrieve the policies that are normally kept on site and then make the update. All ciphertexts encrypted by the impacted policy must be re-encrypted after the policy is modified. The ciphertexts will be uploaded to the cloud after they have been re-encrypted. Such activities are often carried out by data owners and result in processing and communication overhead on their end. In the CP-ABE setting, there are two techniques for implementing policy update: ciphertext updating and proxy re-encryption (PRE).

Update on Ciphertext

Data owners must create update keys or tokens and upload them to the outsourced server where the cyphertexts are kept using this approach. The computation of affected characteristics yields update keys, which will be used to update the associated ciphertext components. Most approaches that use this mechanism rely on the linear secret sharing scheme (LSSS), in which policy changes directly affect the access matrix and mapping functions [5,12,13,16].

Belguith et al. [4] suggested an efficient access policy update approach using ciphertext of short size. However, because this scheme is based on KP-ABE, data owners' capacity to define their own access policy is limited. Furthermore, the data owner is responsible for generating the entire updated ciphertext. This increases the computational cost on the data owner.

Li et al. presented an efficient policy and file updating in the CP-ABE setting in [5. The ciphertext components are modified in response to the data owner's key update. It lowers the client's storage and transmission costs. Furthermore, the suggested system is shown to be secure when the decision q-parallel bilinear Diffie-Hellman exponent is assumed. However, in addition to the update key generation based on the LSSS principle, the data owner must keep the encrypted components of the existing ciphertext. Kan Yang et al. suggested a ciphertext updating approach to handle cloud server policy modifications in [12,13]. They looked at the cost of policy updates and developed linear secret sharing schemes for adding and removing characteristics in the AND, OR, and threshold gates of ABE policies (LSSS). Data owners must build update keys based on generic order groups in this technique, and the complexity increases linearly with the number of policy characteristics. As a result, resource-constrained devices are unable to perform bilinear operations. Yuan introduced a policy update method based on the matrix update algorithm and integrated it with the fundamental CP-ABE scheme's encryption algorithm in [16]. The scheme necessitates that the data owner deal with a ciphertext update algorithm that compares the number of attributes in the old and new policies. The cost of computation and communication is determined by the policy size. Varri et al. [17] recently presented a multi-keyword search with dynamic policy update over CP-ABE. This approach updates the policy without selecting a new secret value by using the encryption information from the previous policy. Then, to change the access policy, the UpdateKeyGen technique is used. Even though this scheme is efficient for ciphertext updates caused by policy updates, the cost of the update key computation, which includes transforming the LSSS matrix and mapping function and comparing an old and new policy at the data owner side, is high if the policy contains a large number of attributes.

Re-encryption of proxies (PRE)

Proxy re-encryption was first introduced by Mambo and Okamoto [6]. (PRE). To conduct re-encryption of the ciphertext sent by the originator, the suggested technique uses a delegator notion. The delegator does not learn the decryption keys or the original data under this scheme.

Many works [7-10,19] have used this notion since it offloads the high cryptographic operations expenses. be carried out by the proxy In some designs, the PRE server is isolated to do solely the re-encryption work and connects with a cloud server to compute the secret component for user revocation support [18].

The authors proposed the PRE technique to support key update in [11]. The user must interact with the proxy in order to convey the value of the encrypted ciphertext along with the symmetric key to the proxy, allowing the proxy to retrieve a portion of the ciphertext. The proxy then sends the user another chunk of the ciphertext to decrypt. Even if the decryption processing is partially outsourced to the proxy, consumers must still deal with two inconvenient processes: decryption and communication costs. If there are multiple decryption events, the overhead between the user and the proxy is critical.

We presented an approach termed VL-PRE in [14] to facilitate policy updates in the cloud that are both efficient and computationally cost-effective. PRE and re-encryption key generation optimization are at the heart of the approach. The process on ciphertext re-encryption at the cloud side is still based on the CP-ABE, even though this technique does not require data owners to deal with significant cryptographic operations. Users that need to access the revised ciphertext may have performance issues if there is a high amount of re-encryption jobs Nonetheless, all of the previous studies that rely on both ciphertext updating and proxy re-encryption have not concentrated on the practicality aspect, as three examples show. shortfalls. First, the re-encryption cost is based on a number of updated policy attributes that are incompatible with mobile devices used by PHR owners, especially when the policy is large and updates are frequent. Second, no work has explicitly discussed the confidence of a delegated system such as an outsourced server or a delegated proxy server. Existing works neglect the trustworthiness of key updates or any other secret component conveyed from the data owner to the delegator. Finally, there are no approaches that focus on the complete traceability of policy updates.

The practical and efficient application of secure outsourcing CP-ABE policy updating with complete traceability is the focus of this article.. of policy revision

The practical and efficient application of secure outsourcing CP-ABE policy updating with complete traceability is the focus of this article. To facilitate lightweight policy update, we combine CP-ABE and a proxy re-encryption technique. To enable efficient data re-encryption, the suggested system uses a parallel processing technique. The data owner can alter the policies contained in the outsourced data storage according to our proposed method. The cryptographic details of CP-ABE are also apparent to users, allowing the tool to be used. We introduce the policy versioning technique as another essential feature of our proposed access control strategy to provide strong accountability of policy modification history.

3. BACKGROUND:

The formal idea of CP-ABE and related concepts employed in our proposed system are described in this section.

Encryption based on Ciphertext Policy Attributes (CP-ABE) The construction of ABE is mostly based on bilinear maps. Below is a description of the formal definition of bilinear maps.

Bilinear Diagrams

Let G0 and G1 be two prime-order multiplicative cyclic groups, and e be a bilinear map: G0G0G1. Let g be a G0 generator. Let H: 0*G0 be the hash function used by the security model in random oracle. 1. Bilinearity: e(ua, vb) = e(u, v)ab for all u, v G1 and a, b Zp.

e(g, g) 1. Non-degeneracy:

Allow a set of P1,P2,...,Pn to be specified. An assortment

A two! "!,"#,...," percent $ is monotone if B, C if B A and B C C A.

A 2"!,"#,..., access structure is a monotone collection A of non-empty subsets of P1,P2,...,Pn "a percentage

Access Tree (definition 2) Let T [2] be a tree that represents an access structure. Each non-leaf node in the tree represents a threshold gate, which is described by its children. value. 0 kx numx if numx is the number of children of node x and kx is the threshold value. The threshold gate is an OR gate when kx = 1, and an AND gate when kx=numx. An attribute and a threshold value kx = 1 are assigned to each tree leaf node x. In T, the kofn threshold gate is also allowed; in this case, kx= k, where k is the kofn gate's threshold value. T is referred to as an access control policy ACP in our approach.

The classic CP-ABE is made up of four major algorithms.

Setup. Except for the implicit security parameter, the setup algorithm requires no input. The public parameters are output.

The attributes of the bilinear map e are as follows:increase or decrease it depending on your needs. As a result, you're ready to focus on your apps and give them the speed, accessibility, and similarity they demand without exerting any effort. There are six popular database engines to choose from, including Amazon Aurora and PostgreSQL. Making use of the PK parameters and MK master key.

Generation X (MK, S). The master key MK and a collection of properties S that describe the key are fed into the algorithm. It generates a UDK (user decryption key). Bilinear Maps are used to generate keys technically.

Secure (PK, M, A). Over the universe of characteristics, the encryption method takes as input the public parameters PK, a message M, and an access structure A (also known as an access tree T). M is encrypted and a ciphertext CT is generated using the algorithm. . The algorithm will decrypt the ciphertext and return a message M if the set S of characteristics matches the access structure A.

Secure (PK, M, A). Over the universe of characteristics, the encryption method takes as input the public parameters PK, a message M, and an access structure A (also known as an access tree T). M is encrypted and a ciphertext CT is generated using the algorithm.

.OUR PROPOSED APPROACH:

The system concept and cryptographic construct of our proposed approach are presented in this section.

Model of the System

In our model, PHR owners upload encrypted data files to the cloud server, such as treatment records and patient profiles, and users, such as doctors, can access the shared file if they have the capability (i.e. decryption key) and meet the access control policy.

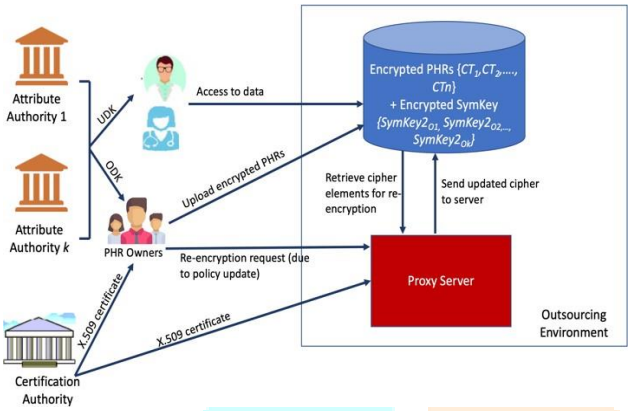

Figure 1: Outsourced PHRs Sharing System Model

As shown in Figure 1, attribute authorities offer PHR owners and users a collection of attributes in the form of a user decryption key. Multiple authorities can issue the attributes to users in our model. A patient, for example, may receive keys from several agencies, such as hospitals and insurance companies. In the field of data outsourcing, When any policy is updated in a cloud computing environment, a proxy, which is a semi-trusted server, is used to accomplish the re-encryption job. As a result, the proxy handles the heavy cryptographic operations and secure processing. An X.509 certificate issued by a trusted certification authority is installed on the proxy. Fig.1. System Model of Outsourced PHRs sharing Fig.1. System Model of Outsourced PHRs sharingsuperiority (CA). Other system entities' identities are verified using the certificate. As a result, the proxy will only communicate with entities that have a valid certificate, as defined by the proxy's configuration system.

B. System Development

The proposed cryptographic methods are based on an expansion of the original CP-ABE [1] in our system. Our solution is made up of two encryption components. Data is first encrypted using AES symmetric encryption. Second, CP-ABE encrypts the symmetric key. The outsourced server stores these two encrypted results. For characterising our cryptographic methods, we define the notations displayed in Table 1.

Table 1: Our model's annotations

Notation	Meaning
$S_{uid.k}$	Set of all attributes issued to user *uid* and managed by authority *k*.
SK_k	A secret key which belongs to authority *k*.
PK_k	Public key which belongs to authority *k*.
$ODK_{oid,k}$	PHR owner decryption key issued by an authority *k*.
$UDK_{uid.k}$	User Decryption key issued by an authority *k*.
ACP_{Pid}	Access control policy identified by policy id *Pid*.
$SymKey1_{oid}$	A symmetric key created from the AES algorithm used to encrypt $ODK_{oid,k}$
$SymKey2_{oid}$	A symmetric key created from the AES algorithm used to encrypt message *M*.
$ENCODK_{oid,k}$	An encrypted form of a $ODK_{oid,k}$ which is encrypted by a AES symmetric key.
RSk	PseudoRandom value used to encode $SymKey1_{oid}$.
ENC_{AES}/DEC_{AES}	Encryption with AES algorithm /Decryption with AES algorithm
$ENC_{CP\text{-}ABE}/DEC_{CP\text{-}ABE}$	Encryption with CP-ABE /Decryption with CP-ABE

System Setup, Key Generation, Encryption, Decryption, and Re-encryption are the five primary phases of our concept. The notations utilised in our model are listed in Table 1.

Phase 1: System Configuration

The AA or data owner runs the following six algorithms during this step.

1. PKk, SKk, PKx.k CreateAttributeAuthority(k) This algorithm takes an attribute authority ID as input (k). The procedure selects a bilinear group G&with generator g of prime order p. It will then choose two numbers at random, Z'. The public key is calculated as follows: Note that the secret key SKk is (, g) and is only used for delegation. For every attributes issued by the Ak, the algorithm additionally exposes public attribute keys (PKx,k).

2.EncODKoid,k(ODKoid,k,SymKey1oid) ENCODKoid,k(ODKoid,k,SymKey1oid) The approach encrypts PHR owner decryption key ODKoid,k with AES encryption using symmetric key1.

ENCODKoid,k = ENCAES (ODKoid,k, SymKey1Oid)

3. Make up a random secret

The PHR owner performs this step by using the following algorithm.

(1) R GenR(r1,r2,...rn)

The technique takes as input a set of random seeds rs and generates a 256-bit random number R (also known as a random string) with the position Rp. Then R and Rp are returned. (2) To a shared symmetric key, add R. RSk AddR(SymKey1Oid)

The proxy server then stores a randomised secret RSk.

Phase 2: Key Development

The AA is in charge of this phase. The user decryption key is generated using the UserKeyGen algorithm (CP-ABE decryption key). The algorithm is described in more depth below.

UDKuid,k. UserKeyGen(Suid,k, SKk,) The KeyGen algorithm accepts a set of attributes (Suid,k) that specify the uid's user decryption key, as well as the attribute authority's secret key (SKk), as input and outputs a set of user decryption keys (UDKs).

The AA,Ak selects a random r and rjZ' for each attribute jS*(for each user uid. The user decryption key (UDKuid.k) is then calculated as follows:

UDK j,k = (D = g()(,-)/((, AiS: Di = g-. H(i)-), D'0 = g-), D'0 = g-)).

This key type is known as an owner decryption key for PHR owners (ODKoid,k).

We presume that ODKoid,k can decode any ciphertexts encrypted by the PHR owner oid,k in our scheme.

Encryption (Phase 3)

The encryption function is handled by the owners of PHRs. This phase performs the following two encryption steps:

(1) Message encryption CT. ENCAES(SymKey2Oid, M)

Message M is encrypted using the encryption technique.

Algorithm for Policy Updates

(1) Updating the policy utilising update operators such as add, update, and delete, (2) Assigning multi-thread to handle transactions, and (3) Re-encrypting encrypted symmetric key are the three states of our proposed policy update technique. The latter procedure runs concurrently with the ReENC function. Our proposed policy update algorithm is shown in Figure 2. Technically, the PHRs owner or administrator can add, amend, or delete each access policy tree's attribute name or attribute value. The thread will be forked and assigned to the encrypted keys that need to be re-encrypted by giving priority to large-file size based on the parallel threshold value, and they will be forwarded to the third state of re-encryption when the policy is modified. Other smaller files are moved to the front of the queue. All affected encrypted keys are updated with a new access control policy in the final state. Our policy update algorithm is shown in Figure 2.

All major procedures are fully offloaded and processed in the outsourcing environment using our proposed technique. As a result, the cost of calculation and communication for the data owner is greatly decreased. In essence, instead of re-encrypting all affected ciphertexts, our technique allows for the re-encryption of an encrypted symmetric key. In comparison to existing PRE schemes, this reduces the complexity of the re-encryption procedure.

4. ANALYSIS OF SECURITY

The proof of security in this paper is based on a game. Because our scheme is based on CP-ABE, we can resort to the original CP-ABE [2, 15] for a full confirmation of its security.

We presume that data owners are completely trustworthy in the data outsourcing context. Users are assumed to be dishonest, which means they may conspire to gain illegal access to data. The attacker is likewise considered to be able to corrupt authorities solely in a static manner, however key inquiries can be made adaptively. An adversary can compromise security by requesting a key from the attribute authority.

Between an adversary A and a challenger C, our suggested system's security model is as follows:

5.CONCLUSION:

A policy update technique based on policy outsourcing and proxy re-encryption has been presented. Our method completely offloads the cost of policy updates to the outsourced server. In addition, the re-encryption process encompasses multi-thread processing for greater scalability and improved overall system performance. We created a GUI tool for implementing CP- ABE policy updates for the trial. Data owners can use our system to upload encrypted data and policies to our outsourced storage. Administrators or data owners do not need to retrieve policies from a local database or communicate with an outsourced server to re-encrypt data. Policies can be modified at any time and from anywhere using our web-based application. As a result, the file storage system and policy update management provide transparent access control. We also presented the policy versioning technique, which allows for the rapid reconstruction of historical policies for thorough auditing. Finally, we demonstrated the performance of file re-encryption. The results showed that the multi-thread re-encryption procedure outperformed the method without one. In the future, we will conduct comprehensive experiments in the real cloud environment to evaluate the cloud-based proxy with greater volumes of data and larger access policies.

6.REFERENCES:

[1]A. Sahai and B. Waters, "Fuzzy Identity-Based Encryption," LNCS, May 2005.

[2] J. Bethencourt, A. Sahai, and B. Waters, Ciphertext policy Attribute-based Encryption, IEEE Symposium on Security and Privacy, Oakland, CA, USA, May 20-23, 2007.

L. Cheung, J. Cooley, R. Khazan, and C. Newport are three of the authors. Using attribute-based encryption, collusion-resistant group key management is possible. Report 2007/161 of the Cryptology ePrint Archive.

PU-ABE: Lightweight attribute-based encryption allowing access policy updating for cloud aided IoT, in Proc. IEEE Comput. Soc. 11th Int. Conf. Cloud Comput., pp. 924–927, 2018.

[5] H. Wang, Huiwen Wang, Huihui Wang, J. Li, Shulan Wang, Yuan Li, H. Wang, Huiwen Wang, Huihui Wang Scheme in Cloud Computing with Policy and File Updates," IEEE Transactions on Industrial Informatics, Vol. 15 (2), pp. 6500-6509, 2019.

IEICE Transactions, E80-A(1): pp.54-63, 1997. [6] M. Mambo and E. Okamoto, Proxy cryptosystems: Delegation of the capacity to decipher ciphertexts, IEICE Transactions, E80-A(1): pp.54-63, 1997.

[7] K.Liang, W.Susilo, and J.K.Liu, Privacy-preserving ciphertext sharing strategy for massive data storage, IEEE Trans. Inf.

August 2015, (8), pp.1578-1589.

[8] Embedding lightweight proxy re-encryption for efficient attribute revocation in cloud computing, Int. Journal High Performance and Computing Networking, Inderscience, Vol. 9 (4), pp. 299–309, 2016.

[9] Y. Kawai, In: International Conference on Information Security Practice and Experience, ISPEC 2015, Beijing, China, 2015.

[10] X. Liang, Z. Cao, H. Lin, and J. Shao, Attribute-based proxy re-encryption with delegation capabilities, ACM, pp. 276–286, 2009.
[11] Yacine Challal and Lyes Touati, "Instantaneous Proxy-Based Key Update for CP-ABE," IEEE Local Network Conference (LNC 2016), Dubai, December, 2016.
[12] K. Yang, X. Jia, K. Ren, R. Xie, and L. Huang, Enabling efficient access control with dynamic policy updating for big data in the cloud, IEEE Conference on Computer Communications (INFOCOM'14), 2014.
[13] K. Yang, X. Jia, and K. Ren, IEEE Trans. Parallel Distrib. Syst., vol. 26, no. 12, pp. 3461–3470, Dec. 2015. Scalable and safe access control policy update for outsourced big data, S. Fugkeaw and H. Sato, Future Generation Computer Systems, Vol. 19, pp. 364-373, 2018.
[14] Scalable and safe access control policy update for outsourced big data, S. Fugkeaw and H. Sato, Future Generation Computer Systems, Vol. 19, pp. 364-373, 2018.
[15] L. Cheung and C. Newport, "Provably ciphertext policy ABE," in Proceedings of the ACM Conference on Computer and Communication Security (CCS 2007), Virginia, USA, October 2007, ACM.
[16] W. Yuan, IACR Cryptology, ePrint Archive vol. 2016. p.457-468, 2016. Dynamic policy updating for ciphertext-policy attribute-based encryption.
[17] U. S. Varri, S. K. Pasupuleti, and K. V. Kadambari, Key-Escrow Free Attribute-Based Multi-Keyword Search in Cloud Computing with Dynamic Policy Update, In Proc. of IEEE/ACM International Symposium on Cluster, Cloud, and Internet Computing (CCGRID 2020), Melbourne, Australia, 11-14 May, pp.450-458, 2020.
[18] J. Cui, H. Zhou, H. Zhong, and Y. Xu, Akser: Attribute-based key word search in cloud computing with efficient revocation, Information Sciences, vol. 423, pp.343-352, 2018.
[18] J. Cui, H. Zhou, H. Zhong, and Y. Xu, Akser: Attribute-based key word search in cloud computing with efficient revocation, Information Sciences, vol. 423, pp.343-352, 2018.
[19] CAPODAZ: A containerized authorisation and policy-driven architecture employing microservices, by Dimitrios Kallergis, Zacharenia Garofalaki, Georgios Katsikogiannis, and Christos Douligeris, Ad Hoc Networks, Vol.104,102153, 2020.
[20] U. S. Varri, S. K. Pasupuleti, and K. V. Kadambari, In Proc. of IEEE/ACM International Symposium on Cluster, Cloud, and Internet, Key-Escrow Free Attribute-Based Multi-Keyword Search with Dynamic Policy Update in Cloud Computing.