International Journal of Soft Computing 11 (6): 362-373, 2016
ISSN: 1816-9503

CB-HPAC Cluster Based–Hierarchical Privacy Preserving Access Control Technique in Cloud

[1]Sudha Senthilkumar and [2]Madhu Viswanatham
[1]School of Information Technology and Engineering
[2]School of Computer Science and Engineering, VIT University, 632 014 Vellore, India

Abstract: Cloud computing is an evolving technology that relies on resource sharing to attain coherence and economies of balance, similar to a utility hosted over a network. Cloud enables a decentralized data storage and online access to data and computing services. However, there exists an access control challenge for sharing data in public clouds, to share data selectively with different levels of user without compromising the confidentiality and privacy of data and user respectively. Several existing methods used to solve this problem such as attribute based encryption; access control policy based encryption which has its own weakness in terms of heavy computation for deriving access structure, user addition and revocation and policy management. To alleviate this weakness, the proposed technique uses the combination of blinded RSA algorithm and cuckoo filter data structure which can solve above addressed problem. According to this general cluster based algorithm can be used for clustering the users according to their policies and mapped with cuckoo hash table to flexible user addition and revocation. Further, to ensure confidentiality and support anonymous authentication within the cluster, the blinded RSA encryption and group signature scheme included with the data stored into cloud. The experimental analysis shows that our proposed scheme proven to provide flexible, efficient and fine grained access control for cloud outsourced data.

Key words: Access control, encryption, blinded RSA, decision value pair, cuckoo filter, system authority

INTRODUCTION

The cloud computing is an emerging technology which is rapidly gaining popularity as an alternative to traditional information technology. Cloud computing delivers a more scalable environment for enormous amounts of data that are used in various applications and services through on-demand self-services. One fundamental characteristic towards this paradigm shifting is that data are being consolidated and outsourced into clouds. The cloud outsourced storage services afford a new profit growth for the user by means of offering a location-independent platform, scalability support for managing their data. The Cloud Storage Service (CSS) releases the burden of data storage management and maintenance. Even though, if this service susceptible to any attacks or failures, it would bring severe economic losses to users as their data are stored into an untrusted pool of storage outside the user enterprises (Takabi *et al*., 2010; Marston *et al*., 2011). These types of security risks imposed due to following reasons: the cloud computing infrastructures are much more reliable and powerful than user personal computing devices. However, they are still vulnerable to various security threats both from inside and outside the cloud users for the benefits of their possession (Pearson, 2009). Therefore, it is necessary for cloud service providers to offer high confidentiality to the data that is stored on it. Therefore, a suitable access control techniques to secure the data access in the cloud is required (Su *et al*., 2011; Yu *et al*., 2010; Wan *et al*., 2012).

The traditional access control systems that are used in unchangeable distributed environment can be classified as Discretionary Access Control (DAC), Mandatory Access Control (MAC) and Role Based Access Controls (RBAC). DAC is generally used in circumstances that adopt that every object has an owner that could control the access rights to the object. MAC assumes that system administrator has the control to create the policies and subjects or owners do not have the capability to override the policy. RBAC is the model which provides the user access based on the roles of the users. In all these models users and resources are identified by the unique names. These models are not appropriate for the dynamic distributed environment such as grid and cloud computing where users are generally identified by their

Corresponding Author: Sudha Senthilkumar,School of Information Technology and Engineering, VIT University,632 014 Vellore, India

attributes and not by predefined identities. Extensive focus towards providing the fine-grained access control for the outsourced data in the cloud has been done by several researchers. Of which wide-spread focus shown for ABE (Su *et al*., 2011; Yu *et al*., 2010) which is classified as key-policy attribute based encryption (Goyal *et al*., 2006) and ciphertext-policy attribute based encryption (Bethencourt *et al*., 2007). The KP-ABE in which (Su *et al*., 2011), access tree structure linked with user's decryption key and ciphertext linked with set of attributes. The attributes linked with the cipher text satisfy the access structure can decrypt the ciphertext. The CP-ABE in which policies associated with cipher text and attributes associated with keys, keys with related attributes satisfy the policy related with the data is allowed to decrypt the data. However, these types of scheme has drawback in introducing heavy computation overhead in terms of deriving the unique logic structure of policy tree and assigning it with file or key. But using this method directly to an organization was not appropriate which structured with hierarchical level of users. Furthermore, various researchers (Wan *et al*., 2012; Freudenthal *et al*., 2002) worked towards to make ABE scheme suitable for the hierarchical nature of organization. Yu *et al*. (2010) suggested a scheme to accomplish secure, scalable and fine grained access control in cloud computing. It adopts KP-ABE together with a re-encryption technique for supporting efficient user revocation. With KP-ABE scheme, the data owner cannot have control that who can decrypt the text (Wan *et al*., 2012) suggested a scheme to achieve scalable and fine-grained access control for the users who works in hierarchical organization structure. They extend the existing CP-ABE scheme with the hierarchical user structure. This scheme lacks in overall system scalability as it makes the data consumer to perform the decryption. Wang *et al*. (2011) offered a scheme by combining CP-ABE with Hierarchical Identity-Based Encryption (HIBE) to accomplish scalable and fine-grained access control with the hierarchical user structure. This scheme does not support compound attributes and multiple value assignment to attributes. Ruj *et al*. (2014) suggested a distributed access control scheme for data access in cloud environment. It uses both attribute based encryption and attribute based signature schemes to provide efficient user revocation and anonymous authentication. It uses the claim policy to prevent replay attacks in cloud outsourced data (Mohan and Elayidom, 2015) proposed a fine-grained access control scheme which combines Ciphertext Policy Attribute Based Encryption (CP-ABE), Counter Mode Eencryption (CTR) and Linear Secret Sharing Schemes (LSSS). They have done the attribute revocation rather than using user level revocation. In all these ABE based schemes, coming up with the unique access tree structure at the owner and storing it in the cloud introduces heavy computation overhead.

Another scheme called Role Based Access Control (RBAC) (Li *et al*., 2015) had been developed which grants users access based on the roles they possess. Roles are defined according to job competency, authority and responsibility within the organization. Within an organization, roles are created for various job functions. The permissions to perform certain operations are assigned to specific roles. Members of staff are assigned particular roles and through those roles assignments acquire the computer permissions to perform particular computer system functions. Even though, the model gives efficient access for the users, it struggles by role explosion problem (Nabeel *et al*., 2013). This model cannot be implemented directly as it faces issues such as storing several users with different roles and with different access permissions, dynamic addition of new users and new roles that implies large storage space for the organization. This issue mostly arises in the large scale enterprises such as software sectors, medical sectors, banking sectors and academic sectors. Several RBAC variation model (Freudenthal *et al*., 2002) is proposed to suite the distributed environment needs that is needed to satisfy the large scale enterprises. The study (Freudenthal *et al*., 2002) pointed out this issue and proposed a role and organization based access control model to overcome this issue. In study, Elliott and Knight (2010) the distributed role based access control model is proposed to overcome the scalability problem. Even though, these model shows improvement in terms of performance efficiency and flexibility, the methods failed to provide extensive solution to scalability problem. In addition to this, the RBAC model still suffer by role explosion problem (Shang *et al*., 2010) which is faced by several large-scale enterprise that requires fine grain access control than what the current RBAC supports.

The ABAC (Nabeel and Bertino, 2014; Fan *et al*., 2014) technique offers comfortable policy creation based on user identity attributes such as empid, department, object size, object created time. Since attributes can be expressed in detail based on user, object and general environment attributes, ABAC guarantees to provide more flexible access control technique that enhances user privacy by providing flexible access control which satisfies the organizational access policies. Several ABAC (Nabeel *et al*., 2013). Nabeel and Bertino (2014), Fan *et al*. (2014) based access control techniques proposed for providing privacy and confidentiality of the outsourced data in cloud. In study, Nabeel *et al*. (2013) Suggested a

solution for selective distribution of content to the users based on group key management techniques. The study (Nabeel and Bertino, 2014) uses the ACP based system which uses the single Layer Encryption Approach (SLE) for supporting fine-grained access control. In this approach the users obtain the unique token from the identity provider and it uses this token as the authenticated identity for the further cloud based data access. Though, this approach provides fine grained access control, it produces heavy computation overhead at the owner end, whenever there is a policy change or file content change. The study, Fan *et al*. (2014) uses the two-layer encryption approach in which the policies are partitioned into two parts. The data owner that carries the first part of ACP policies performs the inner layer encryption. The outer layer encryption performed by the cloud service providers according to the second part of ACP's. But none of these ABAC based schemes suggested a solution to form users groups according to the different requirements of enterprise polices.

By keeping all these issues in mind, the cluster based hierarchical privacy preserving access control technique in public cloud is proposed in this study. The following are the contributions of our scheme:

- Used a general clustering algorithm to form sets groups of users based on certain attributes that are considered important by the organization to define the access based on those attributes (conditional and decisive attributes). Further, sub-groups are created based on their policies
- The cuckoo filters data structure used to provide the efficient addition and revocation of users according to their policies
- Group signatures added to authenticate users who store and modify their data on the cloud

Literture review: Access control issue in the cloud has been addressed by several researchers using various access control methods. ABE (Su *et al*., 2011; Yu *et al*., 2010; Goyal *et al*., 2006; Wan *et al*., 2012), RBAC (Li *et al*., 2008; Freudenthal *et al*., 2002; Elliott and Knight, 2010) and ABAC (Nabeel and Bertino, 2014; Ruj *et al*., 2014) are those methods which are used for outsourced data in the cloud.

ABE scheme called as attribute-based encryption introduced by Sahai and waters (Bethencourt *et al*., 2007) to implement access control techniques based on various attributes. The two variation of ABE such as KP-ABE (Goyal *et al*., 2006) and CP-ABE (Bethencourt *et al*., 2007). proposed by Goyal and Bettencourt *et al*. But using this method directly, to an organization was not appropriate which structured with the hierarchical level of users. Furthermore, various researchers (Wan *et al*., 2012; Cadenhead *et al*., 2010) worked towards, to make ABE scheme suitable for the hierarchical nature of the organization. Yu *et al*. (2010) suggested a scheme to accomplish secure, scalable and fine-grained access control in cloud computing. It adopts KP-ABE together with a re-encryption technique for supporting efficient user revocation. With KP-ABE scheme, the data owner cannot have to control that who can decrypt the text. Wan *et al*. (2012) suggested a scheme to achieve scalable and fine-grained access control for the users who works in the hierarchical organization structure. They extend the existing CP-ABE scheme with the hierarchical user structure. This scheme lacks in overall system scalability as it makes the data consumer to perform the decryption. Wang *et al*. (2011) offered a scheme by combining CP-ABE with Hierarchical Identity-Based Encryption (HIBE) to accomplish scalable and fine-grained access control with the hierarchical user structure. This scheme does not support compound attributes and multiple value assignments to attributes. Ruj *et al*. (2014) suggested a distributed access control scheme for data access in a cloud environment. It uses both attribute-based encryption and attribute-based signature schemes to provide efficient user revocation and anonymous authentication. It uses the claim policy to prevent replay attacks in cloud outsourced data. Lingwei *et al*. (2015) proposed a fine-grained access control scheme which combines Ciphertext Policy Attribute-Based Encryption (CP-ABE), Counter mode encryption (CTR) and Linear Secret Sharing Schemes (LSSS). They have done the attribute revocation rather than using user level revocation. In all these ABE-based schemes, coming up with the unique access tree structure at the owner and storing it in the cloud introduces heavy computation overhead.

RBAC model works on the roles and permissions of the user granted by the system administrator. RBAC model cannot be implemented directly as it faces issues such as storing several users with different roles and with different access permissions, dynamic addition of new users and new roles that imply large storage space for the organization. This issue mostly arises in the large scale enterprises such as software sectors, medical sectors, banking sectors and academic sectors. Several RBAC variation model (Elliott and Knight, 2010) is proposed to suite the distributed environment needs that is needed to satisfy the large scale enterprises. The study (Freudenthal *et al*., 2002) pointed out this issue and proposed a role and organization based access control model to overcome this issue. In study, Elliott and Knight (2010) the

distributed role based access control model is proposed to overcome the scalability problem. Even though these model shows improvement in terms of performance efficiency and flexibility, the methods failed to provide an extensive solution to scalability problem. In addition to this, the RBAC model still suffers by role explosion problem (Shang *et al*., 2012) which is faced by several large-scale enterprise that requires fine-grain access control than what the current RBAC supports.

Several ABAC (Nabeel *et al*., 2013; Nabeel and Bertino, 2014; Ruj *et al*., 2014) based access control techniques proposed for providing privacy and confidentiality of the outsourced data in cloud. In study suggested a solution for selective distribution of content to the users based on group key management techniques. The study Nabeel *et al*. (2013) uses the ACP based system which uses the Single Layer Encryption approach (SLE) for supporting fine-grained access control. In this approach, the users obtain the unique token from the identity provider and it uses this token as the authenticated identity for the further cloud-based data access. Though this approach provides fine-grained access control, it produces heavy computation overhead at the owner end whenever there is a policy change or file content change. The study , Ruj *et al*. (2014) uses the two-layer encryption approach in which the policies are partitioned into two parts. The data owner that carries the first part of ACP policies performs the inner layer encryption. The outer layer encryption performed by the cloud service providers according to the second part of ACP's. But none of these ABAC based schemes suggested a solution to form users groups according to the different requirements of enterprise policies.

In recent times, several researchers focused on providing fine-grained access control solution for the mobile cloud computing environment. Li *et al*. (2015) proposed an access control scheme for mobile cloud environment which uses dynamic attributes related to mobile devices and static attributes used in traditional ABE scheme to ensure an additional level of security in MCC. However, their scheme needs predefined application installed in mobile devices to capture dynamic attributes (Lin *et al*., 2015) suggested a scheme to provide a trust-based access control model for mobile cloud computing environment to protect the privacy information specifically from internal malicious users for big data applications. It uses Vickrey-Clark-Groves (VCG) based adaptive reputation mechanism and the distributed multi-level security scheme and hierarchy key management scheme to achieve fine-grained access control (Mohan and Elayidom, 2015) proposed an access control scheme in cloud environment based on polynomial based approach. The data owner uses the polynomial based secret sharing scheme for key sharing and uses symmetric encryption technique for data encryption. However, no experimental analysis included for comparing the performance of this scheme with conventional signatures schemes. By considering all these factors in various schemes, the cluster-based hierarchical privacy preserving access control technique is proposed to provide the solution to form the group and subgroup according to the enterprise policy, authentication based on group signatures and further provide the solution to support efficient utilization of storage space and support for dynamic user addition, revocation and deletion using cuckoo filter approach.

MATERIALS AND METHODS

Preliminaries

Cuckoo filter: A new data structure called cuckoo filter is used to support the addition, lookup and deletion of items dynamically which improves the performance of our scheme (Fan *et al*., 2014). A cuckoo hashing has the following advantages:

- Support the dynamic addition and deletion of items
- Higher performance
- Easier to implement
- Uses less space than bloom filter

Cuckoo filter uses cuckoo hash table that stores the "fingerprints". A query for an item x checks for the fingerprint of x. If it is present then the access is allowed or else it is denied. Construction of cuckoo filter depends on false positive rate epsilon. The size of the filter increases logarithmically with increase in the number of entries in the table. Cuckoo uses less space while deleting an entry.

Cuckoo hash tables: Hash table contains the array of buckets h1 (x) and h2 (x). Inserting a new item x in the hash table of 8 buckets. If the bucket is empty the insertion is done. If it is not empty item selects one of the position from the bucket and moves the existing element to alternate position within the hash table. The order of execution is O (1). The implementation is performed until the vacant table is found. This can be implemented for more number of displacements (example: 500).

In order to support high speed lookup and high table occupancy, the concept of "Partial-key cuckoo hashing" for moving the entries to alternate locations are used. In terms of false positive rate, space efficiency and lookup

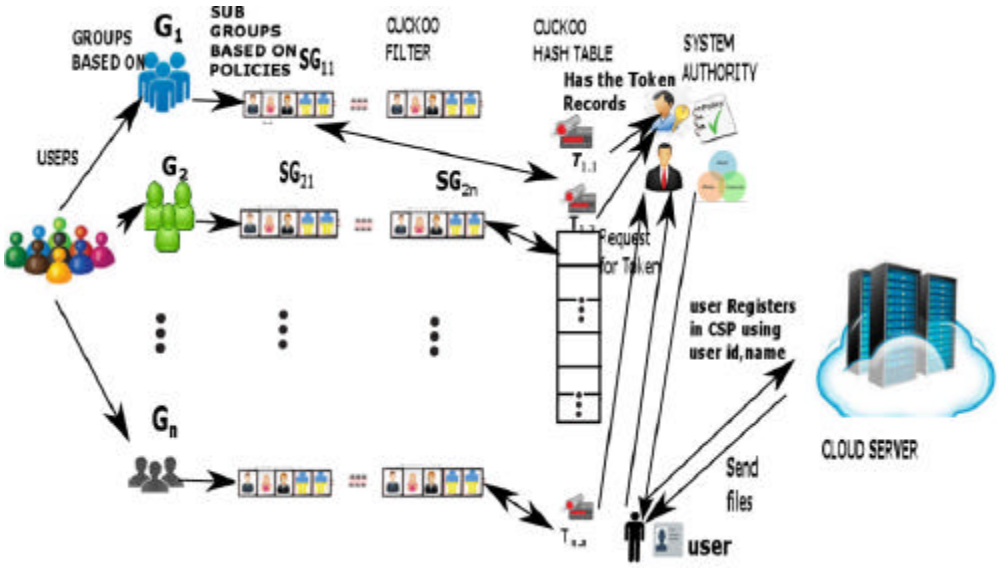

Fig. 1: CB-HPAC system model

insert, delete operations cuckoo filter provide better performance than bloom filter (Ruj *et al.*, 2014).

False positive rate: It is measured by querying a cuckoo filter with non-existing items and counting the possibility of positive return values. To calculate false positive rate the following formula is used:

$$FPR = FP / (FP + TN) \quad (1)$$

Where:

FP = Number of false positive rate and

TN = Number of true negative. Cuckoo filter provides better false positive rate than the other data structure like bloom filter

Group signature: The group signature (Elliottand Knight, 2010) includes the following steps. A dynamic group signature consists of five polynomial time algorithms.

Key generation: The key generation algorithm GKey takes the input parameter 1^k, $k \in N$ and returns the output (grpk, grmsk, regl) in which grpk is group public key and grmsk is group manager key and reg L is the registration list.

Join protocol: The join protocol consists of Join Mgr, Join Ur algorithms. JoinMgr takes input as grmsk and its identity $i \in [1, n]$ and add the registration entry regL [i] at the end. JoinUsr takes input as grpk, identity $i \in [1, n]$ and at the end output the secret signing key grsk [i]. Incase JoinMgr or JoinUsr fails then the respective output is set to.

Group signature generation: The group signature algorithm GrSign uses a secret signing key grsk [i] and a message m and returns a group signature σ.

Group signature verification: The group signature verification algorithm GrVerify uses the group public key grpk as input, a message m and a group signature for m and outputs 1 if signature is valid or else 0 if signature is invalid.

Opening procedure: The opening algorithm open takes as input the group manager's secret key grmsk, a message m, a signature and the registration list regL and returns either a signer's identity $i \in [1, n]$ or 0 (to indicate a failure).

CB-HPAC scheme

System model and security model: Our system consists of following entities as shown in Fig. 1.

Data owner: The data owner is a type of user entity that can store the files into the cloud. The data owner will not perform any encryption at the owner's side. The owner will apply the blind exponent value to the file and transmit to the cloud. At the same time, an un-blind value is transmitted to the SA that required to be specified to the user to attain the original file and use it.

Table 1: Glossary

Notations	Description
G_i	$Group_i$
SG_i	Sub $group_i$
SA	System authority
ACP	Access control policy
ACPB	Access control policy bundle
DO	Data owner
CSP	Cloud service provider
P_i	$Policy_i$
TK	Access token
S_i	Session key
Sk	Symmetric key
F	File
R^{-1}	Un-blind value
R	Blind value
RB	Role bundle
R_i	$Role_i$

SA: System authority is an entity that will have the set of access policies and un-blind values included in it. SA will forward the un-blind values for particular access policies as per the request given by the authorized user in encrypted form and in addition to this it will generate and issue the token to the users based upon the user's request.

CSP: The data are stored in the cloud server. It performs the file encryption for the blind file component and stores it in secure way. It provides the file access to the users as per the users request after verifying their token.

Users: An entity which will access the data from cloud. The user can contact the system authority and the cloud service provider entities through requests for attaining the token and un-blind values and access the file, respectively.

The CB-HPAC scheme begins with mining the database with respect to primary role of users and further decomposes the groups into several sub groups based on access policy conditions. The cuckoo filter hash table is used to map the unique token with respect to subgroups and access control policies, further the token information and access policy information are stored in system authority. The access control is applied to the file stored in cloud depends upon the different subgroup of the users. The user can access the file after they obtain the valid token from the system authority. Figure 1 shows the system model of CB-HPAC scheme and Table 1 shows the notation and its description used throughout this study.

Algorithm 1 (CB-HPAC procedure):

Step 1: The users are grouped according to their roles by invoking the Group Formation Algorithm

$$GF(S) \Rightarrow \{X_1, X_2 .... X_n\}$$

Step 2: Further the user groups based on roles are decomposed into number of subgroups using Subgroup formation algorithm based on policies

$$SF(X_1, X_2 ....X_n) \Rightarrow \{X_1 / P_1, X_2 / P_2 ..... X_i / P_i\}$$

Step 3: The sub groups are managed in the cuckoo filter data structure.

Step 4: Each subgroups are associated with the access token generated by System Authority

$$SG_{i,i} \Leftrightarrow TK_{i,i}$$

Step 5: The System Authority maintain the set of policies and associate it with the tokens

$$SA \Rightarrow \{P_1, P_2 .... P_n\}$$

Step 6: Whenever the data owner wants to upload the file into cloud, it applies the exponentiation with R for file and group signature and submit its subgroup and level to CSP.

$$DO \xrightarrow{(F,Grsign)^R, SG_i, L_i} CSP$$

Step 7: Parallely, data owner generates R^{-1} per ACP's and submit it to System Authority

$$DO \xrightarrow{R^{-1}per\_ACP} SA$$

Step 8: Cloud Service Provider computes encryption and store it in its cloud server along with the key details by using RSA in secure way.

$$E_{sk}\left((file,Grsign)^R\right), L_i, SG_i, S_i(sk), S_i^{ei}$$

Step 9: The user request the CSP to access the file with unique R_{id}

9.1 $$user \xleftarrow{R_{id}} CSP$$

Step 10: Cloud service provider decides the access grant as per the user level and the subgroup they belong to as mentioned in the token.

10.1 CSP $\Rightarrow$ fetch L_i in security token TK

If $((i \in L_i) \leq (j \in L_j)$ in file) then

grant access swith to step 9.2

else deny access

end if

10.2 Cloud Service Provider computes decryption by attaining the key sk.

10.2.1 To attain S_i it multiply with $_iS_i^{ei})^{di}$ infers S_i

10.3 The decryption regains the file with the blind value

$D_{sk}(file,Grsign)^R \Rightarrow (file,Grsign)^R$

Step 11: CSP transmits decrypted file to User

$$CPS \xrightarrow{(file,Grsign)^R} U_i$$

Step 12: User request for un-blind value from SA

$$U_i \xrightarrow{req} SA$$

Step 13: SA sends un-blind value to User in secure way.

$$SA \xrightarrow{E_k(R^{-1})} U_i$$

Step14: User applies un-blind value and verify the group signature and finally it retrieves the file.

$$U_i \xrightarrow{((file,Grsign)^R)^{R^{-1}}} file$$

Framework of CB-HPAC scheme

Phase 1: group formation: The entire data set is grouped according to the role of users that they perform. The Algorithm 2 represents the group formation based on primary role of the users. Suppose the sample space is defined as S = (U, A = c U {d}) where U is the universal set and A is the set of attributes, the entire set of conditional and decisive attributes. Let e be defined as the element in set S and N be the number of elements in the set.

Algorithm 2: User Groups based on roles:

```
Procedure GF(S)
input: the entire decision table S
output: different subsets {X1, X2.....Xn}
begin
for each subsets Xi, i = 1….n
begin
  Xi = ϕ
RB = ∑_{i=1}^{n} Ri
end for
 compute subsets Xi, i = 1….n
 begin
      check attribute 'role' ε {RB} then
      for each Ri ε RB, i = 1…. N
      for each element ej ε S, j = 1 to N,k = 1 begin
          if attr(role) in Cj,k, ε Ri then
                        begin
      Xi:=Xi U ej
         else
          continue
   end for
continue
 end for
end
```

Initially all the subsets [X_1, X_2,.....X_i are empty. The users are moved to any one of the subsets according to their roles which are belongs to the role bundle. This algorithm is repeated until all the users in the database are moved to any one of the subsets. The algorithm should be executed whenever the new user is added into the database.

Phase 2 (subgroup formation based on policies): The Algorithm 3 takes the input as subset X and output the sub groups based on different access control policies. For each subset, the conditional attribute is mapped with polices present in policy bundle. If they match that users are grouped under that policies.

Algorithm 3 (forming subsets based on policies):

```
Procedure SF (X1,X2….Xn)
input: set X which is a group based on roles in S= (U,A=c U {d})
output: set Xi/Pi , groups based on policies,
begin
for each Xi ∈ , {X1, X2.....Xn} i =1……n
begin
 for each Pi ? PB , the policy bundle,  I = 1 to n
 begin
     select subsets Xi: = {|X| X (Pi)}
     Xi/Pi: = Xi – {ei}
end for
end for
end
```

Phase 3 (token generation): The token is generated for each policy that is managed by the system authority and it is associated with policy files. The token consists of Token ID, Subgroup ID, List of File ID's, expiry date, hash value for integrity check and digital signature to ensure confidentiality as shown below:

The tokens are maintained in cuckoo hash table and it is mapped with its respective sub groups to provide common token for all the members in the subgroup. If there are new users inserted into the database, the Algorithm 2 and Algorithm 3 will be executed to decide the users are belong to which group. The deletion of users depends on identifying the subgroup they belong to by tracking with the specific access token and removing it from the database.

Phase 4 (user insertion and user revocation): The user insertion can be performed using the data structure called cuckoo filter. The algorithm 4 (Fan *et al.*, 2014) shows how the new user x1 can be inserted into the subgroup based on the policies.

Algorithm 4(inserting a new user):

```
Function insert(x1)
      fp = fingerprint(x1);
      j1 = hash(x1);
      j2 = i1 ⊕ hash(f);
      if( bucket[j1] ˇ bucket[j2]) =  ⊕ then
      (bucket [j1]  bucket[j2]) = {f};
      return done;
      j = randomly choose j1 or j2;
      for (n = 0; n < MaxNumKicks; n++)  do
      randomly select an entry e from bucket[j];
      Exchange f and the fingerprint stored in entry e;
      j = j  ⊕ hash (f );

      if bucket[j] = ϕ   then
      bucket[j]={fp};
      return Done;
      return failure;
```

The algorithm 5 represents the deletion of the user:

```
 Algorithm 5: Deletion of user
      Function delete(x1)
      fp = fingerprint(x1);
      j1 = hash(x1);
      j2 = j1 ⊕ hash( fp );
      if (bucket[j1] ˇ     bucket[j2]) = { fp } then
      Delete a copy of fp from this bucket;
      return True;
      else
      return False;
```

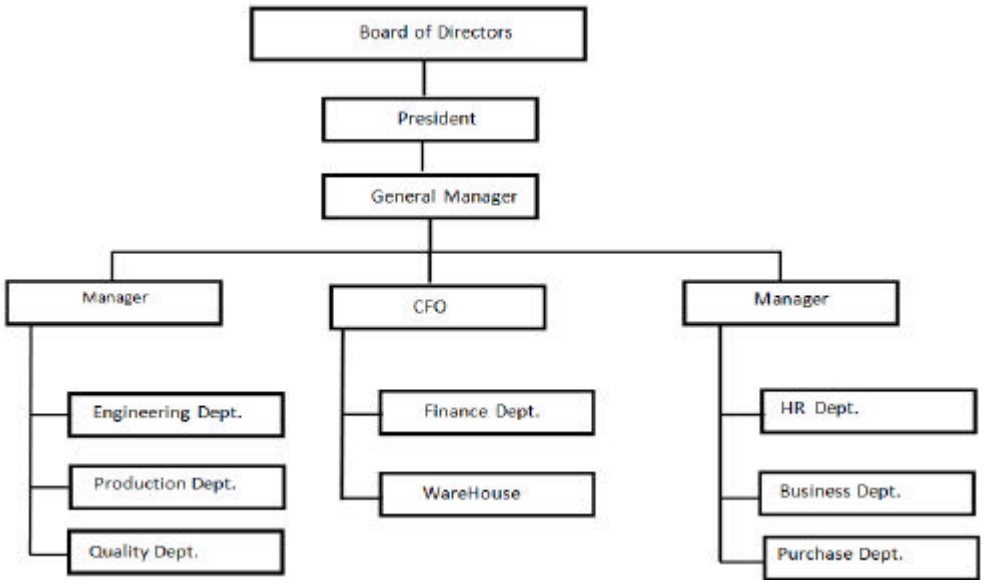

Fig. 2: Manufacturing company hierarchy

Lookup function shown in algorithm 6 involves examinations of two locations in the hash table which uses constant time in the worst case.

Algorithm 6: search item ×1

```
Function Lookup(x1)
fp = fingerprint(x1);
j1 = hash(x1);
j2 = i1 ⊕ hash( fp );
if (fp ∈ (bucket[j1] ˜bucket[j2] ) ) then
return True;
return False;
```

Phase 4 (encryption and file upload): The data owner requests the system authority to issue the keys ei and ni to perform the RSA encryption on the secret key. The data owner encrypts the file F using the secret key and to ensure security, the secret key is encrypted with the session key Si and the session key modified with ei such that $(S_i)^{ei}$. All this encrypted set E_{sk}(F,GrSign), E_{Si}{sk}, Si^{ei} is sent to the cloud and the cloud stores all this information (Senthilkumar and Viswanatham, 2014).

Phase 5 (decryption and file download): The user sends a request with the details of the subgroup they belongs to access the files from the system authority. The system authority contains all the access policies with the RSA keys. Once the user requests the SA, it checks the access policies for the rights of the user and sends a token by which authenticates the user to access the file. The token denotes the set of file the specific subgroup users can access.

When a file is requested by the user to the cloud, it sends the encrypted content E_{sk}(F, GrSign),D_{Si}{sk}, Si^{ei} session key along with a blinded component R is sent to the user. When the user receives the key di using secure SSL connection from SA, the session key can be decrypted which is used to decrypt the secret key and further this secret key used to decrypt the file. The decrypted file used by the users.

Phase 6 (user revocation): The user revocation decomposed into two parts.

User identification: User can be easily identified in the subgroups based on access token mapping. In order to find the user, the searching from the access tokens that is mapped with subgroup user's maintained in cuckoo filters. In case if the assumptions made to revoke the user present in subgroup 2, then the searching starts with access token T_{11} to start searching with subgroup 1 then it uses access token T_{12} to search sub group 2 and further to find the users. This way easily the users can be found by searching based on access token mapping with subgroups.

Re-encryption: Once the users are revoked, the entire files which are accessed by those subgroup members are re-encrypted to support forward and backword security and re-encrypted files are stored in the CSP. The data owner re-encrypts the file with new symmetric key sk and sk encrypted with S_i and S_i raised with e_i given by SA. The entire details E_{sk}(F,GrSign),E_{Si}{sk}, Si^{ei} are stored into CSP.

Case study: Consider an organization which has the various departments like production department, HR department, quality department, Engineering department, Purchase department, Business department and its controlled by various superior caters employees. There are several employees will be working in each department. With the help of CB-HPAC scheme, the employee groups created based on their role and further the users classified according to the various access policies.

Figure 2 shows the manufacturing company hierarchy with the different departments depends upon the nature of work and authorization level of each user. The access policies can be created and managed by system authorities according to the nature of the documents uploaded into the cloud. Consider the role of various employees are denoted with the notations as Board Of Directors (BOC), Presidents (PR), General Manager (GM), Managers (Mr), Chief Finance Officer (CFO), Engineering Department employees (ED), Production Department (PD), Quality Department (QD), Finance Department (FD), HR Department (HD), Business Department (BD), Purchase Department (PD) and the various documents which are uploaded into cloud with respect to various departments. The hundreds or thousands of access control policies that can be easily created and maintained by system authorities using cuckoo filter data structure. The purpose of using cuckoo filter is to alleviate the storage problem and role explosion issues which is normally faced in role based access control techniques. The following are the sample access control policies which are managed by the system authorities:

ACP_1 = (role = "BOD" {< over all administration activities>, <Revenue details}
ACP_2 = (role = "PR" {<set and monitor the organizational goals >, < View production reports>}
ACP_3 = (role = "GM" {< monitor the various department Details >})
ACP_4 = (role ="MR" {< view the department details controlled by them >})
ACP_5 = (role = "CFO" {< view the finance related documents >})
ACP_6 = (role ="QD" {< view the quality related documents > })
ACP_7 = (role ="HR" {< view the employees personal details > })
ACP_8 = (role = "BD" {< view the business related document>})

The system authority generates the RSA keys and access token for each policy. The access token are unique for each sub group and it is maintained in cuckoo hash table, further the token is mapped with each sub group users. As per example, the first ACP says that board of directors can monitor the administration activities and revenue details. The ACP_2 intended for presidents who can set and monitor the organizational goals and view production reports and general managers can monitor various department details as shown in ACP_3. The remaining ACP's are related to monitoring the documents related to different departments by various department employees. The CB-HPAC scheme can be applied to group the members and further identify the users for subgroups according to the access control policies. Then the user can access the file based on the token issued by the system authority.

RESULTS AND DISCUSSION

The experiment was conducted on a laptop with 3.10-GHz CPU and 3-GB RAM, running Ubuntu 14.04. The performance is evaluated and found to be better than the existing ABAC scheme HB-PPAC. The data sets used to store it in cloud for our experiments are taken from the site https://archive.ics.uci.edu/ml/datasets.html to perform the performance analysis and present as a graph. The performance evaluation of CB-HPAC scheme with 5000 user record samples for grouping and sub grouping the users based on access control policies is considered. CB-HPAC scheme has proven to be efficient and secure in terms of classifying the user according to the policies and performing the user operation effectively with the help of cuckoo filter data structure. (Fan *et al*., 2014). The AES Encryption Algorithm with 256 bit key size is used for performing symmetric key encryption for the files stored into the cloud.

Time complexity: The CB-HPAC scheme executes the group formation and subgroup formation with the quartic time i.e. $O(n^2)$ and user insertion and user revocation being executed in constant time O. The encryption and decryption of both the scheme executed with O(fs). However, the user Insertion and Revocation takes O(n) for HB-PPAC scheme and it takes only O in CB-HPAC scheme (Table 2).

Performance analysis: The result shows that the encryption, decryption for various files sizes of HB-PPAC

Table 2: Comparison of computation for HB-PPAC (Senthilkumar and Viswanatham, 2014 Inpress) and CB-HPAC scheme

Scheme	HB-PPAC (Senthilkumar and Viswanatham, 2014 inpress)	CB-HPAC
Encryption	O(fs)	O(fs)
Decryption	O(fs)	O(fs)
Group and sub-group formation	-	$O(n^2)$
User insertion	O(n)	O(1)
User revocation	O(n)	O(1)

n-Total number of users in the organization; fs-Input file size for encryption/decryption

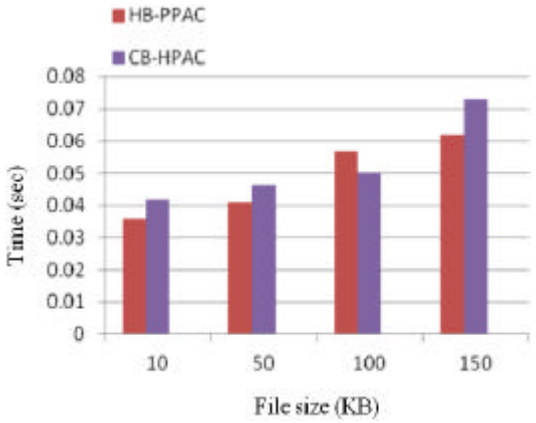

Fig. 3: Average computation time comparison for file upload operation

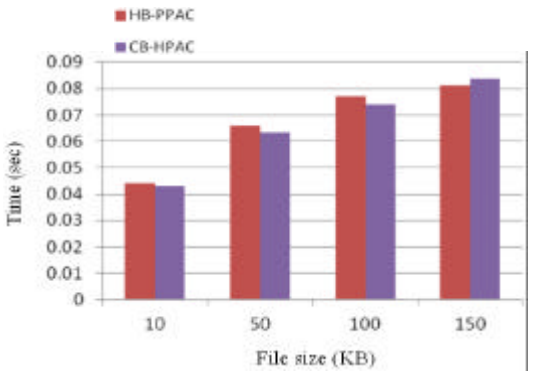

Fig. 4: Average computation time comparison for file download operation

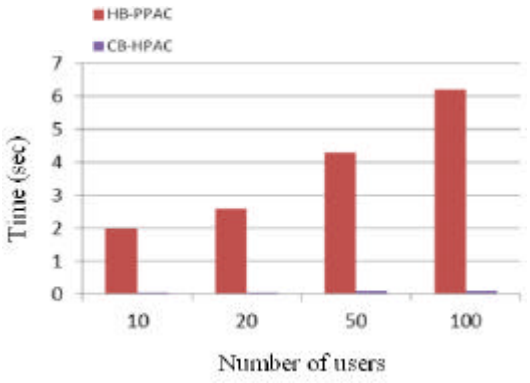

Fig. 5: Average computation time comparison for user Insertion

and CB-HPAC scheme. Similarly the operations for user insertion and user revocation with respect to different users for both the schemes are shown in Table 3.

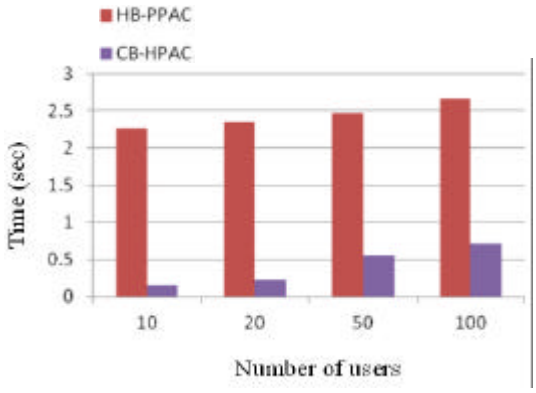

Fig. 6: Average computation time comparison for user revocation

Table 3: Comparison of average computation time for various phases in CB-HPAC scheme with HB-PPAC (Senthilkumar and Viswanatham, 2014 Inpress) scheme

Operation time (sec)	File size (kb)	HB-PPAC (Yu *et al.*, 2010)	CB-HPAC
File upload total time	10	0.036	0.0416
(Encryption +	50	0.041	0.0463
Data transmission)	100	0.057	0.050
	150	0.062	0.073
File download total time	10	0.044	0.043
(Decryption +	50	0.066	0.0632
data transmission)	100	0.077	0.074
	150	0.081	0.084
User insertion	10	2.0	0.05
(In terms of number of users)	20	2.6	0.07
	50	4.3	0.11
	100	6.2	0.12
User revocation	10	2.254	0.16
(Revocation +	20	2.35	0.23
Re-encryption)	50	2.47	0.55
(In terms of number of users)	100	2.66	0.71
RSA key management		0.0045	0.0045

Figure 3 shows the comparison of file upload for the HB-PPAC and CB-HPAC scheme. The encryption time of both the schemes varies with respect to different file size.

Figure 4 shows the comparison of file download operation for the HB-PPAC and CB-HPAC scheme. The decryption time also varies with respect to the various file size for both the approach and time spent also nearly equal to both the schemes.

Figure 5 and 6 show the comparison of user insertion and revocation for different number of users for HB-PPAC and CB-HPAC scheme. The computation time varies with respect to number of users are inserted or removed from its database. Hence the CB-HPAC scheme applies the clustering approach and uses the cuckoo filter techniques, the time it takes to perform user insertion or user revocation is very less compare to HB-PPAC. However both the scheme takes the similar key

Table 4: Comparison of various schemes using different security parameters

Scheme	SR1	SR2	R3	SR4	SR5	SR6	SR7
HASBE approach (Fan *et al*., 2014)	Yes	Yes	No	Yes	Yes	No	No
ABAC based-TLE approach (Nabeel *et al*., 2013)	Yes	No	No	Yes	Yes	No	No
HB-PPAC approach (Senthilkumar and Viswanatham, 2014 Inpress)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
ABAC based -CB-HPAC approach	Yes	Yes	Yes	Yes	Yes	Yes	Yes

management time which is nearly 0.0045 sec. Similar way the clustering process in our scheme takes approximately few milliseconds which is negligible.

Security and privacy analysis: The security properties of CB-HPAC scheme is analyzed afterwards the performance analysis of CB-HPAC scheme is done.

Confidentiality and integrity: The proposed scheme reveals only the encrypted file details to CSP, excluding other information like policy details, token details and RSA key management are preserved by System Authority SA. This guarantees that our scheme preclude the CSP to leak or misuse any user information. Furthermore, the message exchange between all the parties done through SSL secure channel. To ensure the anonymous authenticity, the group signature generated for particular group is added with the file before it gets uploaded into cloud. The file integrity verified with the help of HMAC computed by Data owner.

Freshness guarantees: In order to ensure the freshness of token distributed by system athority, it appends the expiration time for each token; the CSP can provide the access according to the token expiration time. This can avoid the role back attacks initiated by the illegal users with expired token.

Assured file deletion: Assured file deletion are ensured with the help of blind RSA scheme. The SA destroys the RSA private key of the files whenever the file needs to be revoked from the CSP. Even though the file deleted from CSP, it may retain the redundant backup copy in other server. As per our system design it is not probable to attain the symmetric key without that file RSA private key. Hence the file cannot be recovered by anyone else. This implies the assured file deletion once the file revoked from the CSP.

Prevention of eavesdropping: The proposed system prevent the eavesdropper to access the data transferred in the communication channel between the communicating parties such as data owner, cloud service provider, data user and system authority by enforcing the encrypted form of data transfer. In addition to this, the group signature added to ensure the integrity and authenticity of data.

Prevention of side channel attacks: An attacker tries to compromise the cloud environment by inserting a malware virtual machine in nearby location to an estimated target cloud servers and then initiate a side channel attack. In the proposed scheme even if the attacker initiates the side channel attacks, the secured encrypted form of information present in the virtual machine is not leaking the any user information to the malware virtual machine.

Prevention of authenticity attacks: The security token is used to protect the authentication attack by including the hashing code and including the expiration time is one of the parameter to protect the authenticity attack to be issued by the hackers.

Prevention of man-in-the middle cryptographic attacks: Further the man-in-the middle attack is eliminated by including the SHA-256 hash based authentication code during the data transmission between the communicating parties.

SR1-providing Confidentiality and Integrity SR2-Ensuring Freshness Guarantees SR3-Assured File deletion SR4-Prevention of Eavesdropping SR5-Prevention of Side Channel attacks SR6-Prevention of authenticity attacks SR7- Prevention of Man-in-the middle cryptographic attacks.

As denoted in Table 4, the proposed CB-HPAC scheme fulfils the various security parameters which may not be covered by other existing schemes.

CONCLUSION

In this study, a cluster based hierarchical privacy preserving access control techniques in cloud is proposed to provide a secured access control in public clouds. The general clustering algorithm used for performing effective user grouping in large data sets according to the access policies. The cuckoo filters data structure used for insertion, user searching and dynamic user deletion. This makes CB-HPAC scheme unique compared to other existing access control scheme. The access control also provided based on the access token issued for each sub group users. The experimental analysis proves that CB-HPAC scheme performs better by reducing the computation cost for group formation, user insertion and revocation. As a future work, the data auditing services can be included in the proposed scheme to track the user access.

REFERENCES

Bethencourt, J., A. Sahai and B. Waters, 2007. Ciphertext-policy attribute-based encryption. Proceedings of the IEEE Symposium on Security and Privacy, May 20-23, 2007, Berkeley, CA., USA., pp: 321-334.

Cadenhead, T., M. Kantarcioglu and B. Thuraisingham, 2010. Scalable and efficient reasoning for enforcing role-based access control. Proceedings of the IFIP 24th Annual Conference on Data and Applications Security and Privacy, June 21-23, 2010, IFIP, Rome, Itlay, ISBN:978-3-642-13738-9, pp: 209-224.

Elliott, A. and S. Knight, 2010. Role Explosion: Acknowledging the Problem. In: Software Engineering Research and Practice, Elliott, A.A. and G.S. Knight (Eds.). Royal Military College, Kingston, Ontario, Canada, pp: 349-355.

Fan, B., D.G. Andersen, M. Kaminsky and M.D. Mitzenmacher, 2014. Cuckoo filter: Practically better than bloom. Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies, December 2-5, 2014, ACM, New York, USA., ISBN:978-1-4503-3279-8, pp: 75-88.

Freudenthal, E., T. Pesin, L. Port, E. Keenan and V. Karamcheti, 2002. DRBAC: Distributed role-based access control for dynamic coalition environments. Proceedings of the 22nd International Conference on Distributed Computing Systems, July 2-5, 2002, IEEE, New York, USA., ISBN:0-7695-1585-1, pp: 411-420.

Goyal, V., O. Pandey, A. Sahai and B. Waters, 2006. Attribute-based encryption for fine-grained access control of encrypted data. Proceedings of the 13th ACM Conference on Computer and Communications Security, October 30-November 3, 2006, ACM Press, Alexandria, VA, USA., pp: 89-98.

Li, F., Y. Rahulamathavan, M. Conti and M. Rajarajan, 2015. Robust access control framework for mobile cloud computing network. Comput. Commun., 68: 61-72.

Li, Q., M. Xu and X. Zhang, 2008. Towards a group-based RBAC model and decentralized user-role administration. Proceedings of the 2008 28th International Conference on Distributed Computing Systems Workshops, June 17-20, 2008, IEEE, Beijing, pp: 441-446.

Lin, H., L. Xu, X. Huang, W. Wu and Y. Huang, 2015. A trustworthy access control model for mobile cloud computing based on reputation and mechanism design. Ad Hoc Networks, 35: 51-64.

Lingwei, S., Y. Fang, Z. Ru and N. Xinxin, 2015. Method of secure, scalable and fine-grained data access control with efficient revocation in untrusted cloud. J. China Universities Posts Telecommun., 22: 38-43.

Marston, S., Z. Li, S. Bandyopadhyay, J. Zhang and A. Ghalsasi, 2011. Cloud computing: The business perspective. Decis. Support Syst., 51: 176-189.

Mohan, L. and M.S. Elayidom, 2015. Fine grained access control and revocation for secure cloud environment: A polynomial based approach. Procedia Comput. Sci., 46: 719-724.

Nabeel, M. and E. Bertino, 2014. Privacy preserving delegated access control in public clouds. IEEE. Trans. Knowl. Data Eng., 26: 2268-2280.

Nabeel, M., N. Shang and E. Bertino, 2013. Privacy preserving policy-based content sharing in public clouds. IEEE. Trans. Knowl. Data Eng., 25: 2602-2614.

Pearson, S., 2009. Taking account of privacy when designing cloud computing services. Proceedings of the ICSE Workshop on Software Engineering Challenges of Cloud Computing, May 22-23, 2009, Vancouver, Canada, pp: 44-52.

Ruj, S., M. Stojmenovic and A. Nayak, 2014. Decentralized access control with anonymous authentication of data stored in clouds. Parallel Distrib. Syst. IEEE. Trans., 25: 384-394.

Senthilkumar, S. and M. Viswanatham, 2014. ACAFD: Secure and scalable access control with assured file deletion for outsourced data in cloud. J. ICT Res. Applic., 8: 18-30.

Shang, N., M. Nabeel, F. Paci and E. Bertino, 2010. A privacy-preserving approach to policy-based content dissemination. Proceedings of the 2010 IEEE 26th International Conference on Data Engineering, March 1-6, 2010, IEEE, Long Beach, California, ISBN: 978-1-4244-5444-0, pp: 944-955.

Su, J.S., D. Cao, X.F. Wang, Y.P. Sun and Q.L. Hu, 2011. Attribute based encryption schemes. J. Software, 22: 1299-1315.

Takabi, H., J.B. Joshi and G.J. Ahn, 2010. Security and privacy challenges in cloud computing environments. IEEE Secur. Privacy, 8: 24-31.

Wan, Z., J. Liu and R.H. Deng, 2012. HASBE: A hierarchical attribute-based solution for flexible and scalable access control in cloud computing. IEEE Trans. Inform. Forensics Secur., 7: 743-754.

Wang, G., Q. Liu, J. Wu and M. Guo, 2011. Hierarchical attribute-based encryption and scalable user revocation for sharing data in cloud servers. Comput. Secur., 30: 320-331.

Yu, S., C. Wang, K. Ren and W. Lou, 2010. Achieving and fine-grained data access control in cloud computing. Proceeding of the 29th Conference on Information Communications, March 15-19, 2010, San Diego, CA., USA., pp: 1-9.