Chapter 4

Fine-Grained Access Provision in Cloud Computing

The chapter presents a conceptual design that accomplishes fine-grained data access and offers an adaptable access control scheme to cloud users. This chapter makes use of the concepts and techniques described in chapter 2 of the thesis. Chapter 3 describes the experimental results and real time implementation benefits of the concepts presented in this chapter.

4.1 INTRODUCTION

The possession of fine-grained data access control and storage correctness verification remains to be a mandatory feature in any system, which shares the data contents among multiple users with different level of trust. In order to ensure the property of cloud data security, highly trusted cloud users may be allowed with full access rights while the other users were assigned with a partial access rights over the outsourced data. Effective management of the fine-grained access provision in a system with users having different access privileges remains to be a challenging issue in cloud computing. In order to solve these issues, the chapter defines a Storage Correctness and Fine-grained Access Provision (SCFAP) scheme that provides appropriate user access rights with improved storage correctness verification techniques. The assumptions made for this chapter are described in section 4.2. Related works are defined in section 4.3. Section 4.4 of this chapter provides brief description about
the proposed SCFAP scheme. Case study is described in Section 4.5. Implementation details, results and discussions are described in section 4.6, 4.7 and 4.8. Section 4.9 summarizes the chapter.

4.2 ASSUMPTIONS

This chapter assumes an existing data access control model to build upon, and the proposed design makes use of the access control properties defined in chapter 2 of the thesis. The hierarchical structure defined in this chapter is assumed to provide many-to-many data sharing in a secured manner through which the property of fine-grained access control, confidentiality and non-repudiation of the outsourced data were achieved.

4.3 RELATED WORKS

This section describes and analyses other approaches towards facing the challenge of fine-grained access provision to cloud users. Multiple solutions are examined, after which an overview of their works were given. This section also describes the comparison of two major approaches that is related to the fine-grained access provision techniques.

4.3.1 OVERVIEW

This section presents an overview of the works, which is related to the proposed SCFAP scheme described in this chapter.

(Yang and Jia, 2014) presents a data access control scheme DACMAC for multi authority cloud storage system. It provides a multi authority CP-ABE scheme with efficient data decryption and user revocation func-
tions. This work further offers an Extensive Data Access Control Scheme (EDAC-MACS) that provides secured user data access even at weaker security assumptions. The security analysis results of this scheme prove that this scheme is collusion resistance but lacks at the property of fine-grained access provision to the individual users of the system.

The ID based cryptographic scheme (Kaaniche et al., 2013), makes use of the user attributes such as user id for encryption and decryption process of the outsourced data. The development of ID based cryptographic scheme provides the secured data storage over public cloud and improved client authorisation for other users to access the data content.

In HASBE (Wan et al., 2012), the user access rights were provided on the basis of the hierarchical access structure framed for each user of the system. This scheme ensures the property of scalability through the extension of ASBE (Attribute-Set Based Encryption) technique. It defines a hierarchical structure that delegates the operation of trusted authoritys private key generation to the domain authorities of lower level. Here the user attributes were converted into the set structure of recursive type that permits the users to define constraints dynamically by representing different combination of attributes, which satisfies the user access policy. This ensures the property of flexibility and fine-grained access control over HASBE systems. The concept of Hierarchical Based Access Structure is extended to form the Hierarchical Structure used in this chapter.

(Wang et al., 2012) proposed a flexible distributed storage integrity auditing mechanism that consists of homomorphic tokens and erasure-coded data. Tokens are provided to the users from randomly chosen block indices from each data vector space analogous to memory loca-
tion of the user requested file in the cloud. The use of erasure coded data technique protects the user data and eliminates the system errors such as data redundancy, fault tolerance and server crashes.

In Privacy-Preserving Public Auditing for Secure Cloud Storage by (Wang et al., 2013) comprises a third-party auditor (TPA) for auditing the integrity of outsourced data; this eradicates the new threats and realizes the data privacy. This scheme uses random masking technique integrated with a homomorphic authenticator that ensures the privacy of public auditing.

Flexible distributed storage integrity checking mechanism is proposed by (Bobba et al., 2009) using homomorphic tokens and it avoids security problems like identifying unknown users. Through the use of homomorphic tokens and distributed erasure coded data, users were permitted to audit the outsourced data. This auditing permits the users to identify both the improper data access and cloud server misbehaviours. This scheme even ensures the cloud data security, which allows the users to perform dynamic operations easily over the outsourced data. Experimental analysis of their proposed scheme proves that it provides high efficiency against Byzantine failure, unknown user attacks and attacks on cloud data modification. Access control schemes based on the token system were developed to provide higher security over the cloud storage systems.

### 4.3.2 COMPARISON OF RELATED WORKS

This section presents a brief summary about two major approaches relating to the proposed SCFAP scheme. The properties of each analysed...
solutions provided by related works were compared to the thesis goals according to the levels described in Table 4.1. The HASBE scheme presented by (Wan et al., 2012), and the flexible integrity auditing mechanism provided by Wang Cong et al were taken into comparison and it is described as follows:

**WORK BY WAN ZHIGUO ET AL**

To ensure the property of scalability and flexibility over outsourced data a solution is presented in work done by (Wan et al., 2012). This work presents a Hierarchical Attribute-Set-Based Encryption (HASBE) scheme to cloud users, which extends the property of Cipher-text attribute-set-based encryption technique. This scheme not only aims in the achievement of scalability, it even inherits the property of flexibility and fine-grained access provision through the management of compound attributes. The HASBE scheme makes use of multiple value access expiration time to deal with user revocation problems. First part of this work describes the extension of HASBE from ASBE technique using hierarchical structure. Whereas the second part provides a clear demonstration about the implementation of access control scheme based on HASBE for cloud computing.

The cloud computing system considered in this work consists of five major entities. The cloud service provider provides services to users. The data owners share their data contents through cloud in an encrypted manner. Data consumers decrypt the shared contents to perform their respective access operations. Each data owner and data consumer was assigned with a domain authority, where each domain authorities could be managed through parent domain authorities or through trusted domain
authorities. The major responsibility of every domain authority is to manage the domain authorities at next level or the data owner or consumer in its domain. In HASBE scheme the data consumers were only assumed to possess read access. All the entities associated with this scheme were organised in a hierarchical manner to accomplish their tasks.

A recursive set based key structure is formed for every user, where each element of the set is either a set or an element corresponding to a user attribute. The depth of the key structure is found using the level of recursions in the recursive set, which is similar to the definition of depth tree. For a key structure of depth 2, members of the set can either be sets or attribute elements at depth1. At depth 2 it is mandatory that all the members of set should be of attribute elements. A unique label for the user attributes were formed using key structure. The access structure to the users in HASBE were formed in a similar way to the ASBE scheme given by (Bobba et al., 2009). In access tree structures the leaf nodes were considered to be the attributes and non-leaf nodes represents the threshold gates. The non-leaf nodes were defined using its children and threshold values.

This work provides user access provision with the help of the hierarchical access structure and it is formed using appropriate user key structure and access structures. This means that the user with private key corresponding to attributes in key structure would be able to access the data, only when their attributes satisfies the access policies defined at the access structure. System Setup, Top-Level Domain Authority Grant, New Domain Authority/User Grant, New File Creation, User Revocation, File Access, and File Deletion are the seven major operations associated with
Wan Zhiguo et al HASBE scheme. Each major system operations associated with the HASBE scheme invokes the appropriate algorithms associated with it to accomplish their tasks and it works on the basis of bilinear mapping concepts. Through the use of this operations every users of the system shares and uses their data contents using HASBE scheme.

Though Wan Zhiguo et al scheme provides better solution to scalability and flexibility issues, the complete support for compound values and multiple value assignments are measured and found to be lagging in efficiency. This reduces the level of fine-grained data access. The proposed SCFAP scheme defines users with their role-based classification. This provides efficient support for compound attributes and multiple value assignments. The hierarchical structure defined in SCFAP scheme improves the level of fine-grained access provision associated with individual users of the system. The HASBE scheme further does not provide write access to the data users of the system. This makes its application inappropriate to critical systems like financial sectors, where several users require write operations to be performed. SCFAP scheme allows the users to perform write operations in an effective manner and it is achieved through the use of token granting system, which preserves the storage correctness of the outsourced data.

**WORK BY WANG CONG ET AL**

An approach to form solution for security risks accompanying the correctness of physical possession over outsourced data were done by (Wan et al., 2012). This work presents a flexible distributed storage integrity auditing mechanism, which ensures the correctness of outsourced data through the use of homomorphic token and distributed erasure-coded
data. This scheme provides efficient user auditing of cloud data with very lightweight communication and computation cost. The auditing result provides both storage correctness guarantee as well as fast data error localization (identification of server misbehaviours). It even allows user access operations over outsourced data including block deletion, modification and appends functionalities. The overall contributions of this work is summarised as follows:

1. In comparison to many of its predecessors, this scheme achieves both the storage correctness insurance and data error localisations.

2. This scheme further supports secure and dynamic operation over data blocks including update, delete and append.

3. The work further makes an extensive security analysis that shows its resistance towards byzantine failures and malicious data modification attack and server colluding attacks.

The flexible integrity auditing mechanism discussed in this section consists of four major entities, which includes User, Cloud Service Provider (CSP), Cloud Server (CS) and Third Party Auditor (TPA). Users share their data through cloud storage services and a user can be either enterprise or an individual customer. Cloud Server (CS) is managed by the CSP to provide better computation and storage facilities to the users of the system. TPA is an optional entity with expertise qualities that user does not possess. TPA assesses and describes the risk of cloud storage services on behalf of users upon request. This work provides more focus towards file oriented data rather than non-file oriented applications like social networking systems. Block level operations over user data
were considered as block update, block delete, block insert and append operations. The major focus of this work is to identify the key integrity issues like unauthorised data modifications and corruptions, caused due to server compromises and random byzantine failures.

Users store their valid credential to cloud servers through CSPs. The problem of data redundancy could be employed through the technique of erasure correcting code. This scheme further tolerates faults and server crashes that happens due to increasing data users. The users interact with cloud servers for processing file retrieval request through CSPs. As it is not feasible for the users to possess their data locally, it is necessary to verify the correctness and maintenance of the cloud data. The users were provided with the pre computed tokens that provides correctness assurance to the users of the system. Tokens were derived from the subset of file blocks in a random manner. The verification token helps the users to ensure correctness of data operation request processed by the CSP. Tokens are issued to the user based on randomly chosen block indices from each data vector space corresponding to memory position of the requested file in the cloud and erasure-coded data. In case of inappropriate situations like insufficient resources and time the users can delegate their responsibilities to TPA. The system is designed in such a way that leakages of users outsourced data towards auditing protocol were prohibited. This work achieves secure data storage through five major steps, which includes file distribution preparation, challenge token pre-computation, correctness verification and error localisation, file retrieving and auditing and finally, towards third party auditing. The algorithms associated with each stage helps in the management of activities accompanying data
storage management and correctness verification processes. This scheme provides an approach methodology that prevents CSP to process data dynamics without knowing user secret key materials and ensures users that dynamic data operation request done by CSP were processed faithfully. In this manner the property of integrity assurance and storage correctness where done in (Wan et al., 2012) scheme.

Tokens were provided to the users, based upon randomly generated block indices and memory position of the file. This makes the property of storage correctness associated with integrity auditing scheme to be a probabilistic feature. The proposed SCFAP scheme solves this issue by granting tokens to the users in a deterministic manner. In SCFAP scheme tokens were derived from Meta data containing file locations and distributed to all the users of the system during appropriate phases. Here the major advantage is that as a result of the write operation done by the authorised system an updated token would be provided to all the users of the system, through which the property of storage correctness is achieved.
Table 4.1: Comparisons of Works Relating to SCFAP Scheme

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Security</strong></td>
<td>Cloud and User</td>
<td>Data Confidentiality</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>Data Integrity</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>Cloud and User</td>
<td>Data Availability</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>Non-Repudiation</td>
<td>++</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Cloud and User</td>
<td>Accessibility</td>
<td>–</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>Expressiveness</td>
<td>+ + +</td>
<td>++</td>
</tr>
<tr>
<td><strong>Access Control</strong></td>
<td>Cloud</td>
<td>User Addition and Revocation</td>
<td>+ + +</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>User and Cloud</td>
<td>Fine-grained Access Control</td>
<td>+ +</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>Access Control Delegation</td>
<td>+ +</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>User and Cloud</td>
<td>Many-Many File Sharing</td>
<td>–</td>
<td>++</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
<td>Cloud</td>
<td>Scalability</td>
<td>+ + +</td>
<td>+ + +</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>Flexibility</td>
<td>+ + +</td>
<td>+ + +</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>Durability</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>
4.4 FINE-GRAINED ACCESS PROVISION IN CLOUD COMPUTING

4.4.1 OVERVIEW

This section describes the Storage Correctness and Fine-grained Access Provision (SCFAP) Scheme, which is our conceptual design that applies the elements from the research and experiences of related work to form a fine-grained access control scheme for cloud environment. An essential part of SCFAP scheme is based on the concept of attribute based hierarchical structure and token granting system that provides fine-grained data access and storage correctness to the cloud data users. The SCFAP scheme will be presented in a top-down manner by first defining the key concepts and terminologies and then defining the structure of the access control scheme. Finally, details regarding SCFAP applications and implementation results were described. The SCFAP scheme described in this chapter is an extension of (Wan et al., 2012) and (Wang et al., 2012).

4.4.2 MOTIVATIONS AND OBJECTIVES

The wide spread adoption of cloud computing services increases the need for security features over outsourced data. The competition among different cloud service providers enriched the quality of cloud computational services in terms of flexibility, scalability, adaptability and security of the outsourced data. It has been found that most of the existing access control scheme provides the user access privileges centered on their user job functions. This reduces the fine-grained access level associated with
an individual user access rights. This leads to the development of an access policy with fine-grained access provision and storage correctness verification for cloud data users. The major objective of the proposed SCFAP scheme is to improvise the fine-grained access level and storage efficiency associated with the individual data users of the cloud computing environment.

4.4.3 SYSTEM DESIGN

This section presents a conceptual design of the novel scheme called Storage Correctness and Fine-grained Access Provision (SCFAP) scheme, which is described in Figure reffig:cf1. The proposed SCFAP scheme consists of two parts. The first part deals with the construction of hierarchical based user access structures and the second part depicts the algorithmic phases associated with SCFAP scheme that helps in the achievement of fine-grained data access and improved storage efficiency across the outsourced cloud data storage. A set of appropriate cryptographic keys and access structures derived from the exact user attributes were distributed to all the users of the system. Through the use of the access structures and cryptographic keys every user of the system performs the cloud data access in a secured way. As a result of the encryption process, both the data owners and users were provided with a token, which assists in the process of integrity and security verification over the outsourced data.
4.4.4 SYSTEM ENTITIES

The entities involved with the system are described as follows:

Attribute Authority (AA) manages all the attribute related activities with specialisation to activities associating to user role management. This includes management of activities like user role revocation, delegation, key allocation and validation of user given inputs. Cloud Server (CS) computes the user given inputs and produces the respective user outputs and acknowledgements.

The Cloud Service Provider (CSP) affords the services to cloud data users. A Service consumer consumes the services provided by the cloud service providers. Service Consumers were also called as the users of the system.
4.4.5 ACCESS ASSIGNMENT STRUCTURE

HIERARCHICAL STRUCTURE

The hierarchical structure defines the access policy associated with the individual users of the system. A hierarchy is framed from the combination of the user unique and common attributes. Each hierarchy represents the one to one relationship between the user and their access policies. The access policy defines the set of operations (read or write access) the user could perform over the outsourced data. A description to access assignment structure is depicted in Figure 4.2.

KEY STRUCTURE

Key structures were designed to preserve the security of the outsourced data. Key structures are derivatives from the user common attributes like roles. The formation of key structure assigns the access privileges to the set of the common users over the outsourced data. This states that users beneath a particular role were assigned with a key structure such that they could gain access to a particular set of files.

ACCESS STRUCTURE

Access structures were designed to achieve the property of fine-grained
user access and it is derived from the user unique attributes like user id. It defines the extent to which an individual user could access the data.

**GRADE**

![Figure 4.3: Access Limits Associated With a Grade](image)

Grade denotes the level of extent to which the set of common users could gain access to a particular set of files. Each grade formally represents a key structure, such that a user with certain grade could gain access to all the files that comes under the scope of a particular grade. Grades were derived from the user common attributes like dep id, such that it represents a set of files that belongs to the particular department. An example to measure of grade is described in Figure 4.3.

**BARRIERS**

Barriers are restrictions imposed over the grades to achieve the fine-grained user access level. It has been found that it is not necessary for a user with a particular grade to access all the files that comes under a particular grade. In order to solve this issue, barriers were designed and imposed over the user grades.
From figure 4.4 it is understood that though the user belongs to grade 1 which provides access rights to three files F1, F2, F3, the imposition of the barrier B1 over the particular user grade G denies the user file access request to the file F2; such that the user could only be able to access the files F1 and F2. This provides the appropriate access rights to the users through which the property of fine-grained access provision is achieved.

**TOKENS**

Tokens were derived from the metadata containing the file location and it acts as a user authentication entity. As it is described in section 4.3 tokens were issued to the data users as a result of the data encryption process. Through the use of the tokens, the user could easily verify the existence of their corresponding files in a convenient manner. Since the token represents the Meta data about the file location, it assists in the process of easier file retrieval. This improves the storage efficiency of the proposed system. Further, clear descriptions about the working of the
tokens were described in phased 6, 7 and 8 of the SCFAP scheme. A new method of mathematical modelling was used to identify the functions and variables of the proposed scheme.

4.4.6 PRELIMINARY ALGORITHMS

KeyGen( )

KeyGen( ) is a basic algorithm for key generation through which all the other keys associated with the data users were derived. This algorithm is invoked automatically whenever the process of key generation is required.

Let us consider the set of keys $k$ such that it contains a set of integers up to $n$. Such that $K_n = \{k_1, k_2, ..., k_n\}$.

$$k_n : \sum_{Z=0}^{n} K_Z \approx \sum_{m=0}^{n} d_m$$  \hspace{1cm} (4.1)

Where $d$ is the set of derived keys. It could be of any other keylike master key, public key and private key. In a simpler way the KeyGen algorithm generates random number of keys in accordance to the user given input parameters.

FORMATION OF HIERARCHICAL ACCESS STRUCTURE

The proposed SCFAP scheme makes use of the hierarchical access structure to define the user access rights. The basic concept behind the hierarchical access structure was described in section 4.4.5 of the thesis. In SCFAP scheme each user is assigned with a hierarchical structure, which is derived from their respective key and access structures.

KEY STRUCTURE

Key structure is premeditated to preserve the security of the outsourced
data and it represents the access rights to the group of users with common identity. Section 4.4.5 of the thesis provides basic concepts behind the formation of key structure. It is formed using the user common attributes like dep id. In an organisation the most important or most secured files could be accessed only by the personals at the top-most designation order, least important files by the low level personals and ordinary files could be accessed by the mid-level personals. In correspondence to the user designation order grades for a group of members with common identity (users under a particular role) is calculated from “grade1.grade n”. For every user role grades were allocated with respect to their access privilege that defines the level of extent to which the users could access the data.

**ACCESS STRUCTURE**

The access structure represents the access rights to the individual user of the system. The basic concepts of the access structure were described in section 4.4.5 of the thesis. Even though a particular user is assigned with a grade representing the key structure, it is not mandatory that the user could access all the files that come under a particular grade. The access structures associated with the SCFAP scheme were designed in such a way that solves the problem of above mentioned issue. The access structure was framed from the user barrier limits, which are derived from the user unique attributes like user id. Barriers are restrictions that were imposed over the user access grades to achieve the fine-grained access control. The concept of barriers was described in detail at section 4.4.5 of the thesis. The assignment of the access structure defines the individual access limits over the set of files. In addition to this, phase 3 of the
storage correctness scheme provides a brief summary about the algorithmic implementation of the user access structure assignment. Through the use of the key and access structure discussed above, a hierarchical access structure is formed in the proposed SCFAP scheme, and it is illustrated in Figure 4.2.

**TOKEN GRANTING SYSTEM**

The proposed SCFAP scheme makes use of the token granting system through which the property of storage correctness is achieved. As it is described in section 4.4.5, tokens were derived from the Meta data containing the file location that assist in both ways, through which the process of storage correctness as well as the easier retrieval of the outsourced files could be made. The prime idea behind the use of token granting system in SCFAP scheme is that at the end of every successful data encryption process the data users were provided with the tokens, through which the data users verifies the existence of the outsourced data. The users could also be able to perform the decryption process only when the Meta data of the user given token points to the user requested file.

**4.4.7 SCFAP PHASES**

The storage correctness phases and fine grained access provision scheme consists of eight phases through which the property of fine-grained access provision and storage correctness verification is achieved. The SCFAP phases apply the concept of hierarchical access structure and token granting system described in section 4.4.6 of the thesis to achieve its intended objectives.
Phase 1: SetUp ( )

It takes the user security parameter $\lambda$ as an input and generates master key $m_k$ as an output. This step is done by the cloud server through automatically invoking the KeyGen algorithm.

$$K : m_k = \lambda \ast k_n$$

Equation 4.2 joins the user security parameter with the unique key generated by KeyGen( ) algorithm and distributes the master key to the corresponding users of the system.

Phase 2: GradeGen($m_k, R_{id}$)

The GradeGen algorithm defined at this phase make use of the concept of the key structure described in section 4.4.5 of the thesis. This phase is performed by the Attribute Authority and it takes the master key $m_k$ and Role id $R_{id}$ as an input, produces public key $p_k$ and grade $g$ as an output. Public key is derived from the master key $m_k$ by manually invoking the KeyGen( ) algorithm, which is defined at section 4.4.6 of the thesis. Let us consider two sets, $R = \{R_1, R_2, R_3, ..., R_n\}$ and $G = \{g_1, g_2, g_3, ..g_n\}$ be the set of roles and grades. Such that $R \approx G$ (means that the role R is isomorphic to grade G).

$$Z : \forall R_{id} \in R | R_{id} \subseteq R$$

Any $R_{id}$ that belongs to R is the subset of R.

$$Z : \forall R_{id} : P(R_{id}) | R_{id} < \bullet R$$

At least for one value of $R_{id}$ the value of $R_{id}$ in R is true. Such that $R_{id}$
is covered by \( r \) where \( r \in R \).

\[
\therefore Z : \exists R_{id} \rightarrow r | r \Leftrightarrow G
\]  

There exists \( G \) and \( R_{id} \) that implies a role such that the role corresponds to a grade \( G \).

**Phase 3: BarrierGen\((U_{id}, r_k, p_k)\)**

This phase makes use of the concept of access structure defined at section 4.4.6 of the thesis and it is performed by the role admin. It takes \((U_{id}, r_k, p_k)\) use rid, role key, and public key as an input and as a result of computation the barrier limit \( b_l \) and the private key \( p_{rk} \) is returned to the users. The private key is manually generated by role admin through the invocation of KeyGen ( ) algorithm. Let \( U \) be the universal set that contains all the users of the system and can be written as \( U = \{U_1, U_2, \ldots , U_n\} \) and \( B \) be the set of barriers such that can be written as \( B = \{b_1, b_2, \ldots , b_n\} \).

\[
Z : \forall U_{id} \in U_{id} \subseteq U
\]  

\[
\forall U_{id} : P(U_{id})|U_{id} < \cdot
\]  

(4.6) (4.7)

Similarly from 4.4 this step is derived.

\[
\sum_{k=1}^{n} U_k \exists B | \sum_{k=1}^{n} U_k \in B | \sum_{k=1}^{n} U_k \subseteq B
\]  

(4.8)

Means that for all the users there exists a barrier limit such that all the users in \( U \) belong to barriers in \( B \). So that \( U \) is the subset of \( B \). Such that there exists an \( U_{id} \in B \), where

\[
Z : b = \bigsqcup_{u \in U} bu : \bigsqcup_{u \in U} Ru \rightarrow G
\]  

(4.9)
since, all the users U is the subset of B there exists a b corresponding to the user u, where the barriers can be calculated as a co-product of user and barrier sets.

**Phase 4: Encrypt** \((f, r_k, p_k)\)

This phase is done by the CSP and it takes the file f, role key \(r_k\) and Public key \(p_k\) as an input and the outputs cipher text \(c_p\) to the users of the system. Data encryption is done as a part of file upload.

\[
Z : f \times r_k \times p_k \leftrightarrow f^{(r_k, p_k)}
\]  
(4.10)

\[
Z : f \times r_k \times p_k \leftrightarrow c.t | c.t \approx f^{(r_k, p_k)}
\]  
(4.11)

Encryption is done as a combination of input parameters.

**Phase 5: TokGen** \((f, r_k, p_k)\)

TokGen is an algorithm used for token generation and it is done by the CSP. This phase applies the concept of token granting system provided at the section 4.4.6 of the thesis. It takes file f, role key \(r_k\) and Public key \(p_k\) as an input. It is the most important part of the encryption process and it is done during the process of file upload. Since tokens were derived from the data containing file locations, here we use the concept of reduction to reduce a file to tokens. Let \(F = \{f_1, f_2, ..., f_n\}\) be the set of files such that by property of reduction

\[
\exists f \in d_b \bullet \forall R \in F \leftrightarrow f(R) \in t_i
\]

then

\[
F \leq_{db} t_i
\]

\(F \leq_{db} t\) Denotes that a file set F can be reduced to token \(t_i\) and it is
achieved through the data blocks. At the end of this phase tokens generated were distributed to the data users to verify the correctness of the outsourced data.

**Phase 6: Token Computation**

It is done by the CSP and cloud server as a part of the data decryption process. Let $F = \{f_1, f_2, \ldots, f_n\}$ be the set of files and $T_i = \{t_{i1}, t_{i2}, \ldots, t_{in}\}$ be the set of tokens associated with the files. Then,

$$\bigcup [t_{i1}, t_{i2}, \ldots, t_{in}](F_i) = \{d_b[t_{i1}, t_{i2}, \ldots, t_{in}]; d_b \in F\} \quad (4.12)$$

Where, $F_i = \{1, 2, \ldots, n\}$ (Only the tokens between $[t_{i1}, t_{i2}, \ldots, t_{in}]$ can access the files in data blocks). Tokens out of this scope would be computed as corrupted and cannot be accessed. It is based on the projection property. Where,

$$d_b[t_{i1}, t_{i2}, \ldots, t_{in}] = \{(t_i, v) \in d, t \in [t_{i1}, t_{i2}, \ldots, t_{in}]\} \quad (4.13)$$

Means the remaining set of data blocks corresponds to some other tokens. The result of proportion $\bigcup [t_{i1}, t_{i2}, \ldots, t_{in}](F)$ can be found only if $[t_{i1}, t_{i2}, \ldots, t_{in}] \subseteq (F)$. It means a file would be accessed only when token matches with it.

**Phase 7: Token update**

Whenever the data user performs the write operation the tokens associated with the users were updated and distributed to all the associated system entities. This is due to the reason that the process of write operation may extend or delete some part of the file that leads to the change of the Meta data containing the file location. The process of token update is
described as follows:

\[ \text{newt}_i = \frac{t_i}{\bowtie} w_c \]  
(4.14)

Where \( w_c \) is the newly written content.

**Step 8: Token Correctness**

It is done as a part of data decryption during the process of file download. It takes \((t_i, c_p)\) as an input. Let \(t_i, c_p\) be an algebraic function over \(F\) then \(Z: t_i \in T_i; c_p \in C_p\); let us take an element \(t_i \in f\). Such that,

\[
Z: (t_i, c_{p1}) + (t_i, c_{p1}) \sim (t_{i1} + t_{i2} + c_p) 
\]
(4.15)

\[
Z: (t_i, c_{p1}) + (t_i, c_{p1}) \sim (t_{i1}, c_{p1} + c_{p2}) 
\]
(4.16)

\[
Z: f(t_i, c_p) \sim (ft_i, c_p) \sim (t_i, fc_p) \iff t_i \bowtie c_p 
\]
(4.17)

It matches the values in the token and cipher text and returns the mismatch thus the token correctness is verified.

**Phase 9: Decryption\((c_p, r_k, b_l, p_{rk}, t_i)\)** Data decryption is done as a part of the file download process. It takes cipher text, role key, barrier limits, private key and token as input and returns the plain text to the users based on their respective access structures.

\[
Z: c_p \bowtie t_i = \bigsqcup b_1(c_p \bowtie t_i) 
\]
(4.18)

\[
Z: c_p \bowtie t_i = \bigsqcup b_1(c_p \bowtie t_i) ||(c_p \bowtie t_i) = \bigsqcup b_1(c_p \bowtie t_i) \iff P_t 
\]
(4.19)

It combines the cipher text and token depending upon the user barrier levels and provides the plain text.
Table 4.2: Summary of SCFAP Phases

<table>
<thead>
<tr>
<th>Phase No</th>
<th>Phase Name</th>
<th>Input</th>
<th>Output</th>
<th>Doneby</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SetUp( )</td>
<td>$\lambda$</td>
<td>$M_k$</td>
<td>CS</td>
</tr>
<tr>
<td>2</td>
<td>GradeGen( )</td>
<td>$M_k, R_{id}$</td>
<td>$P_k, G$</td>
<td>AA</td>
</tr>
<tr>
<td>3</td>
<td>BarrierGen( )</td>
<td>$U_{id}, R_k, P_k$</td>
<td>$B, P, P_k$</td>
<td>RA</td>
</tr>
<tr>
<td>4</td>
<td>Encrypt( )</td>
<td>$F, R_k, P_k$</td>
<td>$C_t$</td>
<td>CSP</td>
</tr>
<tr>
<td>5</td>
<td>TokGen( )</td>
<td>$F, R_k, P_k$</td>
<td>$T_i$</td>
<td>CS, CSP</td>
</tr>
<tr>
<td>6</td>
<td>TokenComp( )</td>
<td>$F, T_i$</td>
<td>$C_t$</td>
<td>CS, CSP</td>
</tr>
<tr>
<td>7</td>
<td>TokenUpdate( )</td>
<td>$T_i$</td>
<td>$newT_i$</td>
<td>CS, CSP</td>
</tr>
<tr>
<td>8</td>
<td>TokenCorrectness( )</td>
<td>$T_i, C_t$</td>
<td>File Validity</td>
<td>CS, CSP</td>
</tr>
<tr>
<td>9</td>
<td>Decrypt( )</td>
<td>$T_i, C_t, R_k, B, P_{rk}$</td>
<td>Plaintext</td>
<td>CSP</td>
</tr>
</tbody>
</table>

ADVANTAGES OF PROPOSED SCFAP SCHEME

1. Through the use of the barrier limits in user hierarchical access structure helps in the achievement of fine-grained access rights to the users of the system.

2. The algorithmic deployment of the token granting system helps in the achievement of the storage correctness verification of the outsourced data with improved storage efficiency.

3. The tokens were derived from the Meta data containing the file location. This reduces the file retrieval time associated with the user data access request.

4.5 CASE STUDY

It has been found that government agencies adopt cloud services to meet their scaling industrial needs. Though social networking sites were found to be the major users of cloud usage, currently banking contributes to the most activities in the cloud, utilizing 64% of the overall cloud services.
Factors like modernization, innovation in information technology, financial products, liberalization and consolidation of financial markets enforces the transformation of ordinary banking system to the cloud based one.

In cloud banking systems, the clients could access different types of banking services like balance enquiry, fund transformations, etc. The banking system taken in this scenario falls under the category of fully electronic based transaction systems, where the banking sites exhibits all the information like bank locations, bank products, loan enquiry, loan eligibility details, transaction details, statement of accounts and money transfer facilities.

The management of activities associated with this type of cloud banking system includes several entities like bank clients, cloud service providers
and entities associated with the payment gateway. In such a case it is not necessary for the system management entities with common attributes like similar roles to access all the end user provided valid credentials. This gives rise to the need for the application of fine grained-access control scheme over the cloud banking system. In order to solve these issues the proposed SCFAP scheme were applied to the Cloud banking system that solved the problem of fine-grained access level associated with the individual system entities. Let us consider the situation, where the bank client performs the money transaction across the cloud banking system. The process of money transfer requires the fulfillment of several essential details including the clients valid password. Once the client completes the form filling activity the client details were transformed to their respective financial institution cloud servers. The client transaction request could be processed through various system management entities like bank account manager, fund transfer manager, cloud service provider, etc. Though all the associated entities possess the same authority of power, it is not necessary for them to process or access all the user given credentials. During fund transfer, it is necessary that the fund transfer manager could access only the required client details like user account balance and eligibility to perform the transaction. But the fund transfer manager is no way related to the user personal and account credentials. It is appropriate for the account manager to verify and validate client account detail apart from the other user given inputs. In order to implement this restriction, a hierarchical access structure is formed in a similar way to the SCFAP scheme. In this case a hierarchy could be formed through the system entities, common attributes like organization ID along with
their unique user ID. Further, the storage correctness phases depicted in the section 4.4.7 of the thesis could be applied to the cases resembling the need for data integrity assurance. In cloud banking during the process of user registration the user uses their valid credentials as an input and as a result of the user registration a login id and password were given to them that act as a token. It could be used by the clients to verify the validity of the user registered details. Thus the storage correctness of the user given inputs such as bank details was verified.

The above scenario clearly explains the application of SCFAP scheme to cloud banking system and it is illustrated in Figure 4.5. This scheme can also be applied to similar scenarios ranging from the medium-sized to large sized enterprises. Our scheme is highly efficient in cases, where the large number of users with similar roles needs the fine-grained access provisions and frequent verifications to the outsourced data.

4.6 EXPERIMENTAL STUDY

4.6.1 DEPLOYMENT OF SCFAP SCHEME OVER EUCALYPTUS CLOUD

An application is created in Eucalyptus an open source cloud platform, which deploys the proposed SCFAP scheme. An interface is developed using JSP to enable users to authenticate and view the cloud storage. Eucalyptus consists of several other interfaces like cloud42, tAWSTanacasino, EC2 Dream, but it is not feasible for the proposed implementation. MySQL community server 5.7 is used for storage purposes. The proposed SCFAP scheme runs at the infrastructure layer of the eucalyptus and it works
on a layered basis to accomplish its tasks. The SCFAP scheme consists of four layers such as user registration layer, authentication layer, access security management layer and instance management layer as it is described in Figure 4.6. The registration layer performs the user registration process. Authentication layer validates the cloudlet credentials and the access security management layer allows or denies the user file access requests to the virtual machines with the aid of the functionalities present at the instance management layer. The application consists of the Command Line User Interface (CUI) using Euca2ools, which allows the users to interact with the system. Here the users of the system were considered to be the subjects and the files uploaded to the cloud were assumed to be the objects. Every subject creates the newer objects and requests their corresponding object access through the proposed SCFAP scheme that preserves the storage correctness and fine-grained access provision of the user data. The security management layer controls and directs the access control schemes. The implementation consists of the web interface that possesses the property of ease of use through which the Cloud Service Providers (CSP) or Attribute Authorities (AA) creates the restriction over the cloud instances. The implemented SCFAP scheme allows the AA to manage access to cloud resources, instances, virtual machines and common user groups associated with the cloud computing environment.
The first step associated with the implementation of SCFAP scheme over eucalyptus cloud consists of the set up phase. Once the subject is registered with the system, the AA adds the subject to the common user groups. This indicates that all the subjects under the common user group contain the common access privileges with certain individual restrictions. These restrictions were imposed over the subjects through the assignment key structure and access structure to the subjects. Every subject can create new objects and gain access to existing objects based upon the following access restrictions:

1. Grades  Defines the level of extent to which a common user group can access.

2. Barriers  Defines the level of extent through which an individual user can access.
3. Tokens Derived from Meta data containing file location.

Each subject shares their newly created objects to the other subjects of common user groups. The subject can also share their objects to other Common User groups which they were not a member of. In order to gain access to the shared objects and instances the subject accessing the object could not violate the SCFAP access policy. Each subject could ensure the property of integrity over their respective subjects through the use of token granting systems. An updated token would be distributed to every subject associated with the shared object in case of modifications to the shared data objects. In this manner the subject could gain access over the outsourced objects.

The implemented SCFAP consists of two distinct types of interfaces that include subject management and object management interfaces. The subject management interface assists in the management of all the subject related activities that includes subject authentication, user common group assignments, subject key allocation and management. The object management interface is responsible for the process of management of all the object related activities that include object storage, object retrieval, token generation, token computation and management.

4.6.2 VALIDATION OF PREMISES

Built on the open source cloud platform Eucalyptus the application of SCFAP scheme to the developed prototype could be validated through the verification of premises that happens in four steps, which are described as follows:

1. Each authenticated subject should be assigned to the common user
group.

2. Each subject should be assigned with an appropriate hierarchical access structure.

3. Token computation results should match with the user given token and the user given file access request.

4. Encryption and decryption processes could be performed only when the subject given inputs are valid.

4.6.3 ACCESS VERIFICATION TESTS

The first test comprises the access request to the object from the subject who is not the member of the any of the common user group associated with the developed system. The request would be denied by the setup algorithm present at the authentication layer. The next test comprises the file access request from the subject with inappropriate hierarchical access structure. This request is blocked by GradeGen and AccessGen algorithms functioning at the access security management layer. A subject's access request for an object with invalid user credentials like invalid tokens and secret key is denied and the subject is blocked from accessing the services, if he repeats the same for three times. The request for encryption or decryption processes with inappropriate cryptographic keys or inputs by the subjects is blocked through the several algorithms like Setup, GradeGen, BarrierGen, Token computation and Encrypt or Decrypt algorithm present at the user registration layer, Authentication layer Security management layer and instance management layer of the developed prototype. In this manner the exception handling capabilities of
the developed prototype where clearly described using the system access verification tests.

4.7 RESULTS AND DISCUSSIONS

The major objective of the experimental implementation is to validate the level of extent to which the proposed SCFAP scheme provides the property of fine-grained data access to the cloud users. In order to validate this objective prototypes using traditional access control, techniques like ABE and RBAC were implemented and it is compared with the proposed SCFAP scheme. From the results of the implementation a comparison is made in terms of both system performance and fine-grained access provision, which is described in Figure 4.7. First a comparison is made between the amount of data to be retrieved and file retrieval time to find the system performance. At client side, n numbers of client nodes were created and large number of files from different client node was uploaded to the cloud storage. Client file access requests were given from various nodes and the number of client requests per minute was calculated in terms of size of the data files accompanying the client requests and it is kept as X limits. The time taken by the cloud server to respond to user file access requests was calculated in seconds and this forms the Y limits. It has been observed that the time taken for file retrieval with respect to the size of data file for our SCFAP scheme remains constant up to a particular threshold. Even though there is a tremendous increase in file size that happens after a particular threshold, the time taken for file retrieval increases in a consistent manner. But the observation of tradi-
tional access control schemes like ABE and RBAC deviates highly and takes more time for file retrieval after a particular threshold. This is due to the inconsistent nature of their underlying access policies. The comparison between SCFAP with traditional ABE and RBAC techniques in terms of file retrieval time proves that the proposed SCFAP scheme takes reduced file retrieval time than the existing schemes. This is due to the use of the token granting systems. Since the tokens were derived from the Meta data containing the file location, the time taken for file retrieval and storage correctness verification has been comparatively improved. The overall simulation results depicts that the file retrieval has been reduced by 0.5 seconds in comparative to the existing access control techniques. This improves the overall performance measure of the system.

Figure 4.7: Calculation of File Retrieval Time

A measure to the level of fine-graininess associated with the SCFAP
scheme in comparison to the traditional access control methods like ABE and RBAC were made, which is depicted in Figure 4.8.

The level of fine-grained access control is measured on the basis of the extent to which the appropriate access rights were provided to the users of the system. User access policies based upon SCFAP, ABE and RBAC models were designed for each client nodes associated with the system. Through the implementation of SCFAP scheme over the eucalyptus cloud and through the use of vast number of the client nodes, a hierarchy is formed for every user accompanying the client node. A comparative measure of fine-grained access level has been made in association with the depth of access structure. Depth of the access structures were kept in X limits and fine-grained access level in percentage were fixed at Y limits. It has been found that our proposed SCFAP scheme provides better fine-grained access level to the data users even at lesser access structure depth. The other access control techniques taken in to comparison were found to be lagging in efficiency at lower level of access structure depth. This is achieved through the derivation of appropriate hierarchical structures associated with SCFAP scheme, which provides better access provision even at the lower access structure depth. The existing technique lags at fine-grained access provision through the complex access structure formation. The tests were conducted using banking research dataset of federal bank of New York.
4.8 COMPARISON TO THESIS GOALS

Comparisons to thesis goal were made in table 4.3. The SCFAP scheme described in this chapter achieves all the properties of fine-grained data access control in a best way through the use of hierarchical access structures and token granting systems. The properties associated with security and performance was also achieved in a better way. This states that the SCFAP provides fine-grained data access rights with better security and system performance features.
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Security</td>
<td>Cloud and User</td>
<td>Data Confidentiality</td>
<td>++</td>
<td>+ + +</td>
<td>+ + +</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>Data Integrity</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>Cloud and User</td>
<td>Data Availability</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>Non- Repudiation</td>
<td>+ + +</td>
<td>+ + +</td>
<td>+ + +</td>
</tr>
<tr>
<td></td>
<td>Cloud and User</td>
<td>Accessibility</td>
<td>+</td>
<td>++</td>
<td>+ + +</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>Expressiveness</td>
<td>+</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>Access Control</td>
<td>Cloud</td>
<td>User Addition and Revocation</td>
<td>+ + +</td>
<td>++</td>
<td>−</td>
</tr>
<tr>
<td></td>
<td>User and Cloud</td>
<td>Fine-grained Access Control</td>
<td>+</td>
<td>++</td>
<td>+ + +</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>Access Control Delegation</td>
<td>+ + +</td>
<td>++</td>
<td>+ + +</td>
</tr>
<tr>
<td></td>
<td>User and Cloud</td>
<td>Many-Many File Sharing</td>
<td>+</td>
<td>+</td>
<td>+ + +</td>
</tr>
<tr>
<td>Performance</td>
<td>Cloud</td>
<td>Scalability</td>
<td>+ + +</td>
<td>+ +</td>
<td>+ + +</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>Flexibility</td>
<td>+</td>
<td>−</td>
<td>+ + +</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>Durability</td>
<td>+</td>
<td>+</td>
<td>+ +</td>
</tr>
</tbody>
</table>
4.9 SUMMARY

The chapter provides a brief summary about SCFAP scheme through which the challenge of fine-grained access provision and storage correctness associated with the existing access control techniques. The first part of the SCFAP scheme consists of the formation of hierarchical access structures and token granting systems that fixes the appropriate access policies to the cloud data users, thereby improving the fine-grained access level and storage efficiency associated with the outsourced cloud data. The next part applies the predefined SCFAP concepts to the algorithmic phases of the SCFAP scheme to achieve its intended objectives. This chapter explains only about the key structure and Access Structure associated with the plain text but not about the Cipher Text Access Structure. In future this work could be extended for outsourced data decryption techniques.