ABSTRACT

Storage is the most prominent feature of cloud computing, growing rapidly in quality which gives immediate access to information through web service application programming interface (or) web-based content management systems. Cloud storage providers store the data in multiple servers when it is distributed. These servers are maintained by hosting companies for providing immediate access increasing the risk of unauthorized access to the private content of data. The risk of unauthorized access can be reduced by using encryption techniques.

In the proposed system user encrypts all the files with distinct keys before uploading them into the cloud. The user can upload the files as private or public. However, public files can be downloaded directly, but to download the private files a user will send a request to the file owner. The user has a flexibility to request single or multiple files at a time. When the file owner accepts the request the application server provides a single Access key extracted from the attributes of the requested files. This Access key is shared to the requesting user which further retrieves the private key of the files. Using the private key cipher text is converted into plain text, and the plain text gets downloaded. This technique increases the flexibility of sharing the files as we are sharing single Access key for multiple files requested.
TABLE OF CONTENTS

Abstract............................................................................................................................ ii

Table of Contents............................................................................................................. iii

List of Figures................................................................................................................ vi

1 BACKGROUND AND RATIONALE ........................................................................... 1

   1.1 Introduction ............................................................................................................. 1

   1.2 Cloud Service Models ......................................................................................... 1

      1.2.1 Software as a Service ................................................................................ 1

      1.2.2 Platform as a Service ................................................................................ 2

      1.2.3 Infrastructure as a Service ................................................................. 3

      1.2.4 Deployment Models ............................................................................ 3

   1.3 Cloud Storage ..................................................................................................... 4

   1.4 Security Issues with Cloud Storage ................................................................... 6

   1.5 Encryption .......................................................................................................... 7

      1.5.1 Encryption Algorithms ............................................................................ 8

2 NARRATIVE .......................................................................................................... 10

   2.1 Problem Statement ............................................................................................ 10

   2.2 Motivation ......................................................................................................... 11
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3</td>
<td>Project Objective</td>
<td>12</td>
</tr>
<tr>
<td>2.4</td>
<td>System</td>
<td>12</td>
</tr>
<tr>
<td>2.5</td>
<td>Project Functionality</td>
<td>13</td>
</tr>
<tr>
<td>3</td>
<td>System Design</td>
<td>14</td>
</tr>
<tr>
<td>3.1</td>
<td>System Architecture</td>
<td>14</td>
</tr>
<tr>
<td>3.2</td>
<td>System Design</td>
<td>15</td>
</tr>
<tr>
<td>3.3</td>
<td>Data Flow Diagrams</td>
<td>16</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Data Flow Diagram for Uploading the Files</td>
<td>17</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Data Flow Diagram for Accepting/Rejecting the Request</td>
<td>18</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Dataflow Diagram for Downloading the Requested Files</td>
<td>19</td>
</tr>
<tr>
<td>3.4</td>
<td>UML Diagrams</td>
<td>20</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Use Case Diagram</td>
<td>20</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Sequence Diagram</td>
<td>21</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Activity Diagram</td>
<td>22</td>
</tr>
<tr>
<td>4</td>
<td>System Implementation</td>
<td>23</td>
</tr>
<tr>
<td>4.1</td>
<td>Environment</td>
<td>23</td>
</tr>
<tr>
<td>1.</td>
<td>Java / JSP</td>
<td>23</td>
</tr>
<tr>
<td>3.</td>
<td>MYSQL</td>
<td>24</td>
</tr>
<tr>
<td>4.</td>
<td>XAMPP</td>
<td>24</td>
</tr>
<tr>
<td>4.2</td>
<td>Application Modules</td>
<td>26</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

FIGURE 1.1 RELATION BETWEEN SERVICE MODEL AND DEPLOYMENT MODEL [4] ........................................ 4
FIGURE 1.2: STATISTICS OF DATA USED FOR CLOUD STORAGE [5] .......................................................... 5
FIGURE 1.3: STATISTICS OF ISSUES WITH CLOUD STORAGE [6] .............................................................. 6
FIGURE 1.4: WORKING OF ENCRYPTION TECHNIQUE ............................................................................. 8
FIGURE 1.5: WORKING OF SYMMETRIC ENCRYPTION .......................................................................... 9
FIGURE 1.6: WORKING OF ASYMMETRIC ENCRYPTION ...................................................................... 9
FIGURE 3.1: CLOUD REPOSITORY SYSTEM ARCHITECTURE ................................................................. 14
FIGURE 3.2: SYSTEM DESIGN ........................................................................................................... 16
FIGURE 3.3: DATA FLOW DIAGRAM OF USER UPLOADING FILES ....................................................... 17
FIGURE 3.4: DATA FLOW DIAGRAM OF USER FOR ACCEPTING/REJECTING A REQUEST ............ 18
FIGURE 3.5: DATA FLOW DIAGRAM OF A USER FOR DOWNLOADING THE FILE ............................ 19
FIGURE 3.6: USE CASE DIAGRAM .................................................................................................. 20
FIGURE 3.7: SEQUENCE DIAGRAM .................................................................................................. 21
FIGURE 3.8: ACTIVITY DIAGRAM .................................................................................................... 22
FIGURE 4.1: GENERATING UNIQUE ID TO THE EACH USER REGISTERED ........................................ 26
FIGURE 4.2: ENCRYPTING USER PASSWORD ..................................................................................... 27
FIGURE 4.3: REGISTRATION PAGE ................................................................................................... 28
FIGURE 4.4: LOGIN PAGE ................................................................................................................ 29
FIGURE 4.5: COMPARING ENTERED PASSWORD DIGEST WITH DIGEST STORED IN THE CLOUD .... 30
FIGURE 4.6: UPLOADING FILES TO THE CLOUD ............................................................................... 31
FIGURE 4.7: CLASS FOR GENERATING PRIVATE KEY ......................................................................... 32
FIGURE 4.8: CLASS FOR CONVERTING PLAINTEXT INTO CIPHERTEXT ......................................... 33
1 BACKGROUND AND RATIONALE

1.1 Introduction

Cloud computing has become an emerging infrastructure for organizations throughout the world. The cloud computing uses specialized connections with a network of servers gathered substantially for data processing across them. Frequently, virtualization techniques are utilized to maximize the power of cloud computing [1]. Through the use of virtualization, it reduces the need of purchasing, maintaining and updating their own networks and computer systems as it uses the computing resources as a service over a network.

1.2 Cloud Service Models

There are many diverse cloud computing service models. Most fundamental service models include Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS). The acknowledged development models are: (i) public cloud (ii) private cloud (iii) community cloud (iv) hybrid cloud. The most common protocols used to achieve the services of cloud computing are HTTP (Hyper Text Transfer Protocol), HTTPS (Hyper Text Transfer Protocol Secure) to achieve information security and data integrity, and Secure Shell [2]. The overall functionality of these three services is to keep the flow on data and computation.

1.2.1 Software as a Service

According to the definition of Wikipedia, **Software as a service (SaaS)** is a software licensing and delivery model in which software is licensed on a subscription basis and is centrally hosted [3]. This is also popularly known as “on-demand software”. As the name suggests, this is service which provides software for a client request.

Characteristics of SaaS are as follows [2]:

- Provides web access facility for commercial software
• This service provides a central location where the software can be accessed.

• Allows user to integrate different pieces of software using Application Programming Interfaces (APIs)

• This service is “one to many” model.

• User do not need to worry about software upgrades or patches, this service handles by itself.

1.2.2 Platform as a Service

This is a service where a user can develop the applications or programs without worrying about the problems of buying or maintaining the infrastructure. By using this service, the user can easily and quickly create the web application. The only difference between the SaaS and PaaS is that a PaaS service additionally provides the facility to create our own application rather than using the existing ones.

Characteristics of PaaS are as follows [2]:

• This service provides a single integrated development environment for developing, testing, deploying, hosting and also for maintaining the application.

• User can create, modify, test and deploy different UI scenarios in web based user interface which is provided by this service.

• This service contributes Multi-tenant architecture where multiple concurrent users utilize the same development application.

• It provides built in scalability, load balancing and failover.

• Web service and databases are integrated with the help of common standards.

• Subscription and billings can be managed by provided tools.
1.2.3 Infrastructure as a Service

The Service in this layer is the total infrastructure of software like a network, Operating systems and storage too. This is similar to SaaS discussed above where software is served as a client request, but in this layer, users buy those resources as fully outsourced service on demand. The characteristics of IaaS are as follows [2]:

- This service distributes resources as another form of service
- Dynamic scaling is allowed
- This service also has a variable cost, utility pricing model.
- Multiple users can use this service on a single piece of hardware.

1.2.4 Deployment Models

- **Private Cloud** – private infrastructure for an organization and managed by third party.
- **Community Cloud** – shared infrastructure by several Organizations.
- **Public Cloud** – Infrastructure available for everyone.
- **Hybrid Cloud** - This is the combination of two or more cloud infrastructures.

Figure1.1 [4] shows the relation between the deployment models and service models clearly.

- IaaS service models use both Private Cloud and Hybrid Cloud
- PaaS service model use Community Cloud and Public Cloud
- SaaS service model use all other models expect the Private Cloud
  - Since, IaaS service model serves the client request for purchase and the private cloud is solemnly used for them [5].
1.3 Cloud Storage

Cloud storage is the place where digital data is saved in logical pools. Cloud storage spans multiple servers. In favor of that, all the rights for the physical environment belong to the hosting company. Here, clients purchase storage capacity of the providers to host assets facilitated with them in the remote server. In return, cloud storage provides instant access to the information through web service application programming interface (or) web-based content management systems.

With the use of cloud storage, we can store, update, and retrieve data. Since it is stored online there is no data loss and users can get the data from anywhere at any time as long as they have internet access. There are tons of cloud service providers available popular among them are Drop Box, SkyDrive, Google Drive, iCloud, which give limited free storage capacity and can be extended using premium account features.
In Figure 1.2 [6], a large portion of the records maintained on the cloud storage are photographs and individual information. Thus, users are concerned about data integrity and security. Accordingly, they like to protect their data from being abused. Here cloud storage suppliers are responsible for keeping the data available and accessible, and the physical environment protected and running.

Cloud storage is:

- Made up of numerous distributed resources but still acts as one - often referred to as federated storage clouds [7]
- High fault tolerance through redundancy and distribution of data
- Highly durable through the creation of versioned copies
- Typically eventually consistent with regard to data replicas
1.4 Security Issues with Cloud Storage

Security is the huge complication in the cloud storage. It is evident from Figure 1.3 [8], that the highest issue in cloud storage is the security concern. When data is distributed it is stored at more locations increasing the risk of unauthorized physical access to the data.

![Figure 1.3: Statistics of Issues with Cloud Storage [8]](image)

The major vulnerabilities in cloud storage are:

- Data Leakage
- Cloud Credentials
- Snooping
- Key Management
Performance

When data is distributed it is stored at several locations increasing the risk of unauthorized physical access to the data. Sometimes the computationally strong client service provider servers cannot be trusted as clients do not exhibit full control over them [9]. This implies that the major challenges that any cloud computing service provider must overcome is to make sure that if its servers are attacked by hackers, the client data cannot be stolen or misused [10]. Moreover, the confidential client data must remain invisible even to the cloud service providers. However, to overcome the above mentioned security issues in the cloud storage encryption techniques are used.

1.5 Encryption

Encryption is a process used to protect data stored in the cloud from unauthorized users. In other words, the primary purpose of encryption is confidentiality. This technique is advancing day by day. The modern encryption along with confidentiality, they provide key elements of security. They are listed below:

1. Authentication
2. Integrity
3. Non-repudiation

Encryption is also referred as cryptography, which uses a cipher system to change plaintext transforming it into a non-intelligible text. An authorized user can be able to easily decipher the message with the key provided by the owner to recipients, but not the unauthorized interceptors can decipher the message. Figure 1.4 shows the working of encryption technique.
1.5.1 Encryption Algorithms

There are three different basic encryption methods, each with their own merits.

i. Hashing Method

ii. Symmetric Method

iii. Asymmetric Method

i. Hashing

Hashing creates a novel, fixed-length signature for a message or data set. Every “hash” is unique to a selected message. Therefore, minor changes to that message would be easy to track. Once the data is encrypted utilizing hashing technique it is difficult to decipher or reverse the message [11]. Hashing, however not actually an encryption method as such, is still helpful for providing data proficiency and proving data hasn’t been tampered with.

ii. Symmetric method

Symmetric encryption is also known as private-key cryptography and is called so because the key used to encrypt and decrypt the message must remain secure because anyone with access to it can decrypt the data [11]. Using symmetric method, a sender encrypts the data with one key,
sends the data (the ciphertext) and then the receiver uses the key to decrypt the data as shown in Figure 1.5.

![Figure 1.5: Working of Symmetric Encryption](image)

### iii. Asymmetric method

Asymmetric encryption or public-key cryptography is different than the previous method because it uses two keys for encryption or decryption (it has the potential to be more secure as such) [11]. In this method, a public key made is free and will be available to everyone and is used to encrypt messages, and a different, private key is used by the recipient to decrypt messages as shown in Figure 1.6.

![Figure 1.6: Working of Asymmetric Encryption](image)
2 NARRATIVE

2.1 Problem Statement

Users are concerned about their security and privacy of data uploaded into the cloud. As all the cloud services are available at the remote locations, users can’t have the complete control over their data. It is always their basic right to protect their data from unauthorized access. In essence a user will upload the data into cloud using encryption technology where plaintext is changed into ciphertext. To view the unintelligible ciphertext, it need to be decrypted using an instance of encryption algorithm called “Secret Key”. This secret key is shared with the users who would like to access the data. Encryption of the files that need to be uploaded into cloud can be done in two of the following ways:

I. The user can encrypt all the files using single encryption key and upload data to the cloud.

II. The user can encrypt each file with distinct key and upload data to the cloud.

In either way, the user will upload his/her data to the cloud storage system to avoid the access to private content of their data in the cloud storage system. If he/she wants to share their encrypted data with their circle

I. The user needs to send their single encryption key which is used to decrypt the saved data in the cloud storage system.

II. The user needs to send the corresponding distinct keys which are used to decrypt the files that are intended to share.

In the first approach, providing a single encryption key would be inadequate since all the undesirable data may be likewise revealed. Whereas in the second, sharing large amount of
cipher data with their corresponding private keys to their circle will increase the cost. Although, decrypting cipher data with distinct keys will result in loss of efficiency as the number of such keys is as many as the number of shared files [12].

2.2 Motivation

Cloud computing, is trending in all sectors like governments, non-profits or small businesses and even unto fortune 500 companies. However, as organizations continue to take benefits of cloud services, they must consider how the introduction of cloud services affects their privacy and security [13].

The motivation of the cloud repository system is as follows:

i. Providing security to the data stored on the cloud from unauthorized access, intruders, employees of the enterprise, and even from the cloud service providers.

ii. Identity Management to avoid serious crimes involving identity theft.

iii. Increasing the efficiency of the cloud storage system by encrypting files with distinct keys.

iv. Sharing multiple files securely with the registered users in the system.

v. Restricting access control levels for private and public files.

vi. Focus on the decryption of the distinct set of cipher data by using Access key.

vii. Decrease the amount of cost while decrypting the cipher data from the cloud storage system.
2.3 Project Objective

The primary objective of this project is to maintain security of the data stored in the cloud that runs on the network. To process with the cloud system, the system should have network facility. The data stored will be encrypted by the system using symmetric encryption. This encryption is to prevent the unauthorized access, from intruders including employee of enterprise, which attempts to retrieve data of the cloud storage user while the data is in transmission.

In this technique, a user will encrypt the data using the private key and converts plain text into cipher text. Extracted cipher text will be stored in the cloud and the private key used for encryption will be stored in the local database. As the data stored is secure, any type of data such as personal or computed or an application data can be stored. To access the files of other users he/she can make a request. Whenever a request is made, the file owner generates an Access key for the requested set of files. The user can retrieve the shared data based upon the user credentials, file attributes and the Access key [12].

2.4 System

The user needs to be registered in cloud repository system. Once registered he/she can login to the system and upload their files into the cloud. The user can upload their files in two categories,

1. Public files
2. Private files

Uploaded file names and attributes of all users can be seen by registered users. In order to get access to files of other users, they need to be downloaded. However, files uploaded as public
can be directly downloaded and to download the private files, the user needs to request for an Access key. The user can request single/multiple private files to the file owner. The file owner can share the Access key for single/multiple requested files. Additionally, the file owner has the flexibility to accept or reject the request made. The user can download the private files only if the file owner sends an Access key for the requested set of private files.

2.5 Project Functionality

The main functionalities of cloud repository system are listed below

- User Authentication
- Providing security to the data stored in the cloud
- Restricting access control levels
- Requesting access for multiple private files
- Sharing access to multiple requested files
- Generating Access key for requested set of files
- Reducing the decryption cost
- Maintaining logs of downloaded files
3 System Design

This chapter discusses about the architecture of the entire system. This chapter also discusses about data flow diagram, use case diagram, sequence diagram, and activity diagram.

3.1 System Architecture

![Cloud Repository System Architecture Diagram]

Figure 3.1: Cloud Repository System Architecture

The System Architecture of the cloud repository system shown in Figure 3.1 describes various components and communication between those components. A user as depicted in the system architecture, should be authorized to login to the system. The user will communicate with
the application server to store the data onto the cloud through a web browser. When the user upload the data it is encrypted using a key generated and thus uploaded in the cloud. Whenever a user requests for the files stored in the cloud, the file owner shares an Access key for requested files. As soon as the user enters the Access key, it gets the private key used to encrypt that file from the local database and decrypts the file using the private key and gets downloaded.

### 3.2 System Design

Figure 3.2 shows the system design of the cloud repository system. It uses cloud to store information about the users, files uploaded by the users, requests made, Access keys generated for the requested files for the requesting user. The login validations checks the username and password entered with the username and password in the database and confirms or rejects login accordingly. Upon confirmation, the application server will establish a connection with the cloud repository system. After that it will pull all the information from the cloud and show it to the user. This application allows the user to store or retrieve data from cloud repository system. Whenever a user tries to upload a file, a private key will be generated and that key will be used to encrypt the file. The key used to encrypt the file is stored in the local database and the encrypted data is stored in the cloud. Whenever a user tries to retrieve the data the public file can be downloaded directly whereas to retrieve the private files the user needs to request for an Access key. Using this Access key and file name, the private key for that particular file can be taken from the local database by the application server and file can be decrypted and downloaded.
3.3 Data Flow Diagrams

Data flow diagram (DFD) is one of the prominent modelling tools which is used to model system components. These components include input data to the system, various processing carried, external entity that interacts with the system and the information flow in the system.
3.3.1 Data Flow Diagram for Uploading the Files

Figure 3.3: Data Flow Diagram of User Uploading Files

Figure 3.3 shows the flow of process between the components while uploading the files. The user can upload either text or image files. Whenever the user uploads a file, a private key will be generated for that uploaded file. Further, files get encrypted using the private key generated. Here private key is stored in the local database and the encrypted content is stored in the cloud.
3.3.2 Data Flow Diagram for Accepting/Rejecting the Request

![Data Flow Diagram](image)

Figure 3.4: Data Flow Diagram of User for Accepting/Rejecting a Request

Figure 3.4 shows the data flow process when a user gets a request. When the user gets a request for a file he can either accept it or reject the request. If the user rejects the request process will be terminated otherwise, a key will be generated in the process.
Figure 3.5 shows the data flow process of a user for downloading the file. When user downloads the data flow process would start from downloading the encrypted content. By using
Access key it retrieves the private key generated while data is uploaded. So with the Access key and encrypted content it decrypts the file.

3.4 UML Diagrams

3.4.1 Use Case Diagram

![Use Case Diagram](image)

**Figure 3.6: Use Case Diagram**

Use case diagram is a behavioral diagram which depicts the behavior of the system. Use cases represent the activities or the functionalities in the system. Figure 3.6 represents the use
case diagram for the project where actors are the users and the components are the functions performed. Use case diagrams are mostly used in requirements analysis phase of the system.

3.4.2 Sequence Diagram

A sequence diagram is a Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Figure 3.7 shows the sequence diagram for the activities in the cloud repository system.

Figure 3.7: Sequence Diagram
3.4.3 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control. Figure 3.8 show the activity diagram for cloud repository system.

Figure 3.8: Activity Diagram
4 System Implementation

4.1 Environment

The following are used in developing the project:

1. Java / JSP programming
2. NetBeans IDE
3. MySQL database
4. XAMPP control panel
5. Libraries

1. Java / JSP

In the project, J2EE is used in developing Java Servlets. Since it is platform independent and also contains a set of services, APIs, and protocols that can be used for developing web based applications, this technology is used for developing, building and deploying of online Web application. In brief, Java Servlets are java programs written at server side [14]. Whenever the application server gets a client request, servlets are executed at server side. Additionally, these servlets provide the following:

1. Security: Java Servlets inherits the security feature that the Web container provides.
2. Session Management: User identity and state is kept intact across more than one requests.
3. Instance persistence: Frequent disk access is prevented. This enhances server performance.

On the other hand, JSP is a technology used for both web designing and web developing. To put it more clearly, we use HTML for the layout of web page and then Java code or other JSP related tags are used to develop main logic inside the layout. For instance, these JSPs by
using special tags can embed the java functionality into HTML page directly. Hence, lots of time and effort can be saved.

2. NetBeans IDE

NetBeans IDE is the most powerful tool available in the present market. IDE stands for Integrated Development Environment which means it is an integrated tool where various programming applications like C, C++, python, Java and many more can be developed. The most important feature in NetBeans is that, it has various plugins which comes handy in developing any project. It can be installed on any operating system that supports java. NetBeans IDE 7.2 version is used in this project

3. MYSQL

There are two different editions: the open source MySQL community Server and the proprietary Enterprise Server. Out of which, MySQL community Server is most widely used Relational database management system. As discussed above, Apache server uses XAMPP to store all the data like files, username and encrypted password in MySQL database.

4. XAMPP

XAMPP is an open source platform developed by apache. It is web server solution stack package. The main components in XAMPP are

- Apache 2.4.12
- MySQL 5.6.24
- PHP 5.6.8
- phpMyAdmin 4.3.11
- OpenSSL 1.0.11
- XAMPP Control Panel 3.2.1
- Webalizer 2.23-04
- Mercury Mail Transport System 4.63
- FileZilla FTP Server 0.9.41
- Tomcat 7.0.56 (with mod_proxy_ajp as connector)
- Strawberry Perl 7.0.56 Portable

It uses MYSQL database to store data using apache server which is called by tomcat. More importantly, the XAMPP requires only one zip, tar, 7z or exe file to download and run.

5. Libraries

- Activation
- Bcprov-ext-jdk15on-151
- Cos-multipart
- Cos
- Javax.servelet
- Mysql-connector-java-5.0.5
- Servlet-api
- Standard
4.2 Application Modules

The Application Modules for the cloud repository system are as follows:

1. Registration/login
2. Uploading Files
3. Requesting Files
4. Sharing Files
5. Downloading Files

4.2.1 Registration/Login

In this module for the first time login user needs to register with the system to use the application. In the registration page as shown in Figure 4.3 a form will be displayed to the user where valid information needs to be filled in the provided fields with a generated unique user id. A unique user id will be generated using the code shown in Figure 4.1.

```java
Connection con = databasecon.getConnection();
Statement st = con.createStatement();
ResultSet rs = st.executeQuery("select count(id) from reg");
if (rs.next()) {
    String u = rs.getString(1);
    int u1 = Integer.parseInt(u);
    int u2 = u1 + 101;
    String u3 = Integer.toString(u2);
    session.setAttribute("u2", u2);
}
```

Figure 4.1: Generating Unique Id to the Each User Registered
All the required fields need to be filled appropriately. Validations are performed on the fields entered. If the information filled in the form are not according to the requirements the query fails and a catch statement will be able to determine the reason and prompt error messages to the user for resolving this issue. Once user clicks the submit button with valid information it needs to be uploaded in the cloud. However, before uploading the user information into the cloud the application server creates a digest for the password entered by the user as shown in Figure 4.2. The application server will replace the password entered by the user with the digest created and updates into the cloud server. If the registration is successful, the user is redirected to the login page prompting successful registration.

```java
Connection con1 = databasecon.getConnection();
PreparedStatement ps = con1.prepareStatement("insert into reg (id,name,username,password,email,mobileno,date,age,address,gender) values (?,?,?,?,?,?,?,?,?,?)");
ps.setString(1, id);
ps.setString(2, f);
ps.setString(3, username1);
ps.setString(4, password1);
int a1 = st1.executeUpdate("update reg set password=AES_ENCRYPT('"+data1++"', dnynL1xYAjvWiK0cHSQrw') where id='"+ab++"'");
int x = ps.executeUpdate();
```

**Figure 4.2: Encrypting User Password**
Figure 4.3: Registration Page
In the login page, a form will be displayed to the user as shown in Figure 4.4 to enter his credentials provided during registration.

![Login Page](image)

**Figure 4.4: Login Page**

Validations will be performed on the values entered. When the user clicks the submit button creates a digest for the password entered and compares with the digest stored in the cloud server as shown in Figure 4.5. The user can login if the username and password entered by the user matches with the records in the cloud server or else error message will be given to the user for resolving the issue. If login is successful, the user can start managing the files in the cloud server.
Connection con = databasecon.getConnection();
String name1=request.getParameter("uname");out.println(name1);
session.setAttribute("name2",name1);
String password1=request.getParameter("pwd");
Statement st=con.createStatement();
ResultSet rs=st.executeQuery("select * from reg where username='"+name1+"' and password=AES_ENCRYPT('"+password1+"','dnynIL1xYAjvWiK0cHSQrw')");
if(rs.next()){
    response.sendRedirect("userhome.jsp?success");
}

Figure 4.5: Comparing Entered Password Digest with Digest Stored in the Cloud

4.2.2 Uploading Files

In this module, a user can upload text files and image files as shown in Figure 4.6 (a). For each uploaded file a unique id is generated by the application server as shown in Figure 4.6 (b). Additionally, he/she can upload the files as public or private. However, both private files and public files are encrypted and stored using AES algorithm. The class for encrypting the files is shown in Figure 4.8. While uploading, the user needs to mention the file name and upload it. When the user clicks on submit button a private key will be generated. This key is used for converting plain text into cipher text as shown in Figure 4.9. The private key used for encryption is generated using the class shown in Figure 4.7 and is stored in the local database and the cipher text extracted will be stored in the cloud server. If the file is uploaded, success message is displayed as shown in Figure 4.6(c) or else catch block can determine the failure and prompts the error message to the user to resolve the issues.
Figure 4.6: Uploading Files to the Cloud
import com.sun.org.apache.xerces.internal.impl.dv.util.Base64;
import java.io.ByteArrayOutputStream;
import java.io.FileInputStream;
import java.io.FileWriter;
import java.util.Scanner;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import javax.swing.JOptionPane;
import sun.misc.BASE64Encoder;

public class AesEncrDec
{
    SecretKey secretKey;
    public String keytostring(SecretKey skey){
        //converting secretkey to String
        byte[] b=skey.getEncoded();//encoding secretkey
        String stringkey=Base64.encode(b);
        System.out.println("secretkey:"+skey);
        System.out.println("converted secretkey to string:"+stringkey);
        return stringkey;
    }
    public SecretKey Stringtokey(String stringkey){
        //converting String to secretkey
        byte[] bs=Base64.decode(stringkey);
        SecretKey sec=new SecretKeySpec(bs, "AES");
        System.out.println("converted string to secretkey:"+sec);
        return secretKey;
    }
}

Figure 4.7: Class for Generating Private Key
import com.sun.org.apache.xerces.internal.impl.dv.util.Base64;
import java.io.ByteArrayOutputStream;
import java.io.FileInputStream;
import java.io.FileWriter;
import java.util.Scanner;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import javax.swing.JOptionPane;
import sun.misc.BASE64Encoder;

public class Encryption {
    public String encrypt(String text, SecretKey seckey) {
        String plainData = text, cipherText = null;
        try {
            SecretKey secretKey = seckey;
            System.out.println("secret key:" + secretKey);
            //converting secretkey to String
            byte[] b = secretKey.getEncoded(); //encoding secretkey
            String skey = Base64.encode(b);
            System.out.println("converted secretkey to string:" + skey);
            Cipher aesCipher = Cipher.getInstance("AES"); //getting AES instance
            aesCipher.init(Cipher.ENCRYPT_MODE, secretKey); //initiating ciper encryption using secretkey
            byte[] byteDataToEncrypt = plainData.getBytes();
            byte[] byteCipherText = aesCipher.doFinal(byteDataToEncrypt); //encrypting data
            cipherText = new BASE64Encoder().encode(byteCipherText); //converting encrypted data to string
            System.out.println("\n Given text :"+plainData+" \n Cipher Data :"+cipherText+"\n Secretkey:"+secretKey); return cipherText;
        } catch (Exception e) {
            System.out.println(e);
        }
        return cipherText;
    }
}

Figure 4.8: Class for Converting Plaintext into Ciphertext
Figure 4.9: Plain Text Converted to Cipher Text
All the uploaded files of a user can be seen in “MY FILES” screen as shown in Figure 4.10. Additionally, a user can download (or) delete his/her files from the cloud repository system.

Figure 4.10: My Files Screen
Furthermore, a user can see files of all other users in a single window i.e. in “FILES” screen as shown in Figure 4.11. However, the user can only download the files that are made as public. In order to download the private files, the user needs to request the file owner to share the private key.

![Figure 4.11: Files Screen](image-url)
4.2.3 Requesting Files

In this system, a user can see the files uploaded by all the users registered into the system as shown in Figure 4.11. However, files made as public can be downloaded directly. To download the private files, a user needs to send a request to the file owner to share the private key used for encryption as shown in Figure 26. To request the file owner, a user needs to navigate to the request page. Thereafter, the user needs to select type of the file as shown in Figure 4.12 (a) and the file owner name as shown in Figure 4.12 (b). Eventually, all the private files of the selected file owners are displayed to the user where he/she can request the private key for a single or multiple files as shown in Figure 4.12 (d). The request made by the user is sent to the file owner and the success message is displayed to the user as shown in Figure 4.12 (c). File owner can see requests made by all the users in a single window in “REQUESTED FILES SCREEN” as shown in Figure 4.13.

![Figure 4.12: Requesting Private Files](image-url)
Figure 4.13: Requested Files Screen
4.2.4 Sharing Files

Whenever a request is made by the user it is shown in “REQUESTED FILES SCREEN” of the file owner as shown in Figure 4.13. Here user has the flexibility to accept or reject the requests made. In order to accept/reject the requests made he/she needs to select the requested user name as shown in Figure 4.15 (a). Eventually, all the files requested by the user are displayed where he can accept/reject few or all the files requested as shown in Figure 4.15 (b). Whenever the file owner accepts the request a single Access key is generated for the accepted file(s) using the code shown in Figure 4.14 and is sent to the requesting user, and the success message is displayed as shown in Figure 4.15 (c). The Access key generated for the requested files is valid only to the requesting user. No other user will be able to decrypt the requested file using the same Access key. Once the requested file is accepted/rejected, that file is removed from the list of requested files as shown in the Figure 4.15 (c).

```java
StringBuffer sb=new StringBuffer();
    KeyGenerator keyGen = KeyGenerator.getInstance("AES");
    keyGen.init(128);
    SecretKey secretKey = keyGen.generateKey();

    Cipher aesCipher = Cipher.getInstance("AES");
    aesCipher.init(Cipher.ENCRYPT_MODE,secretKey);

    System.out.println("String buffer:"+sb.toString());
    String ss=new Encryption().encrypt(sb.toString(),secretKey);

    String skey=new AesEncrDec().keytostring(secretKey);
```

**Figure 4.14: Access Key Generation**
Figure 4.15: Accepting/Rejecting Requested Private Files
4.2.5 Downloading Files

A user can download his/her files directly from the “MY FILES” page and the requested files can be downloaded in the “RECEIVED FILES” page. All the requests made by the user and the key associated with it are displayed in the received files screen where he/she can download the accepted files as shown in Figure 4.16. Whenever a request is made key element maintains any one of the status mentioned below:

- Waiting
- Accept
- Reject

First, when the request is made by the user, the status of the key element will be in waiting until file owner accepts or rejects. Second, if the file owner accepts the request made then the status will be changed to the Access key shared. Last, if the file owner rejects the request made, the status will be changed from waiting to reject. Here only the accepted files would be able to download by the user. In order to download the accepted files, a user need to navigate to the “RECEIVED FILES SCREEN” and select the accepted file and enter the Access key shared by the file owner as shown in Figure 4.16. Whenever the Access key is entered, it compares with the requested username and file name associated with it. If it matches the application server will get the private key used to encrypt the file from the local database and decrypts the file using the class shown in the Figure 4.18 and downloads the file. If the Access key entered does not match, it gives appropriate error message to the user for resolving. A user can download multiple files requested at the same time by selecting the requested time and date as shown in Figure 4.17.
Figure 4.16: Downloading Received Files

Figure 4.17: Downloading Received Files at Same Time
import com.sun.org.apache.xerces.internal.impl.dv.util.Base64;
import java.io.ByteArrayOutputStream;
import java.io.FileInputStream;
import java.io.FileWriter;
import java.util.Scanner;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import javax.swing.JOptionPane;
import sun.misc.BASE64Decoder;
import sun.misc.BASE64Encoder;

public class Decryption{
    public String decrypt(String txt,String skey){
        String decryptedtext = null;
        try{
            //converting string to secretkey
            byte[] bs=Base64.decode(skey);
            SecretKey sec=new SecretKeySpec(bs, "AES");
            System.out.println("converted string to seretkey:"+sec);
            System.out.println("secret key:"+sec);

            Cipher aesCipher = Cipher.getInstance("AES"); //getting AES instance
            byte[] byteCipherText =new BASE64Decoder().decodeBuffer(txt);
            aesCipher.init(Cipher.DECRYPT_MODE,sec,aesCipher.getParameters());

            byte[] byteDecryptedText = aesCipher.doFinal(byteCipherText);
            System.out.println("upto this ok");
            decryptedtext = new String(byteDecryptedText);
            System.out.println("Decrypted Text:"+decryptedtext);

            return decryptedtext;
        } catch(Exception e){
            System.out.println(e);
        }
        return decryptedtext;}
}

Figure 4.18: Class for Decrypting Cipher Text to Plain Text
If any user downloads the file, immediately “file downloaded” message will be displayed in the “DOWNLOADED FILES” screen with the user name, file name, time and date as shown in Figure 4.19.

![Figure 4.19: Displaying Downloaded Files](image-url)
5 Testing and Evaluation

In this phase, functionalities of the application are to be tested like

1. User authentication
2. Restricting access control levels
3. Requesting and sharing multiple files
4. Generating user specific Access key,
5. Eliminating unauthorized access.

5.1 Test Case 1

In this test case user, authentication is tested in registration page and login page. Here validation for user credentials is verified as shown in Figure 5.1. In addition, validations are performed for empty fields for login page and registration page as shown in Figure 5.2. Validations for appropriate information is verified in the registration page as shown in Figure 5.2 where error message is displayed when the username given already exists.

![User Authentication](image)

**Figure 5.1: User Authentication**
5.2 Test Case 2

In this test case, multiple functionalities have been tested i.e. requesting and sharing multiple files, restricting access control levels, eliminating unauthorized access.

In this scenario, users LusianMiller, MichelCorwin and CassandraDavis send request to share the same set of files to user JohnathonTaylor as shown in Figure 5.3. JohnathonTaylor now accepts the request made by LusianMiller as shown in Figure 5.4(a), rejects the request made by CassandraDavis as shown in Figure 5.4(b) and neither accepts nor rejects the requests made by MichelCorwin and therefore the request remains in waiting state.
Figure 5.3: Multiple users requesting same files

<table>
<thead>
<tr>
<th>IMAGE NAME</th>
<th>REQUESTED USER</th>
<th>REQUESTED DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>lab</td>
<td>LusianMiller</td>
<td>05/05/2015 11:05:33</td>
</tr>
<tr>
<td>Lecture</td>
<td>LusianMiller</td>
<td>05/05/2015 11:05:33</td>
</tr>
<tr>
<td>lab</td>
<td>MichelCorwin</td>
<td>05/05/2015 11:05:11</td>
</tr>
<tr>
<td>Lecture</td>
<td>MichelCorwin</td>
<td>05/05/2015 11:05:11</td>
</tr>
<tr>
<td>lab</td>
<td>CassandraDavis</td>
<td>05/05/2015 11:05:53</td>
</tr>
<tr>
<td>Lecture</td>
<td>CassandraDavis</td>
<td>05/05/2015 11:05:53</td>
</tr>
</tbody>
</table>

Select User: CassandraDavis

Submit  clear

Figure 5.4: Accepting One Request and Rejecting One Request

(a)  (b)
By using the key shared by user JohnathonTaylor as shown in Figure 5.5, user LusianMiller can download the files.

<table>
<thead>
<tr>
<th>FILE NAME</th>
<th>ACCESS KEY</th>
<th>STATUS</th>
<th>OWNER NAME</th>
<th>DOWNLOAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>lab</td>
<td>riOSkxDgU7dD/kTZx8+4Jg==</td>
<td>accept</td>
<td>JohnathonTaylor</td>
<td>Download</td>
</tr>
<tr>
<td>Lecture</td>
<td>riOSkxDgU7dD/kTZx8+4Jg==</td>
<td>accept</td>
<td>JohnathonTaylor</td>
<td>Download</td>
</tr>
</tbody>
</table>

**Figure 5.5: Received keys shared**

As shown in Figure 5.6 CassandraDavis request is rejected and MichelCorwin’s request is in waiting as shown in Figure 5.7, both will not be able to download the file. If MichelCorwin, CassandraDavis get access to the key sent to LusianMiller even then they will not be able to download the files as the key is restricted to LusianMiller.

First, file name for that Access key is verified. Second, it checks with the owner name, owner id and requester user name, requested user id. If the comparison gets success it allows the requested user to download the file. If the comparison fails, it displays an error message to the user to enter the correct Access key which is demonstrated in the Figure 5.9.
### Figure 5.6: Rejected Request

<table>
<thead>
<tr>
<th>FILE NAME</th>
<th>ACCESS KEY</th>
<th>STATUS</th>
<th>OWNER NAME</th>
<th>DOWNLOAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>lab</td>
<td>reject</td>
<td>reject</td>
<td>JohnathonTaylor</td>
<td>Download</td>
</tr>
<tr>
<td>Lecture</td>
<td>reject</td>
<td>reject</td>
<td>JohnathonTaylor</td>
<td>Download</td>
</tr>
</tbody>
</table>

Select Request Date to Download Files

**Select Date:** 05/08/2015 16:05:27

Submit  clear

### Figure 5.7: Waiting Request

<table>
<thead>
<tr>
<th>FILE NAME</th>
<th>ACCESS KEY</th>
<th>STATUS</th>
<th>OWNER NAME</th>
<th>DOWNLOAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>lab</td>
<td>waiting</td>
<td>waiting</td>
<td>JohnathonTaylor</td>
<td>Download</td>
</tr>
<tr>
<td>Lecture</td>
<td>waiting</td>
<td>waiting</td>
<td>JohnathonTaylor</td>
<td>Download</td>
</tr>
</tbody>
</table>

Select Request Date to Download Files

**Select Date:** 05/08/2015 16:05:47

Submit  clear
Figure 5.8: Three Users Using Same Access Key

Here all the three users try to use the same Access key as shown in Figure 5.8, but only LusianMiller would be able to download the file. When LusianMiller downloads the file “file downloaded” message will be sent to JohnathonTaylor as shown in Figure 5.10. If the other two
user’s try to download the file with the same Access key Which is sent to Lusian Miller an error message will be displayed to enter correct Access key as shown in Figure 5.9.

Figure 5.9: Displaying Error Message for Unauthorized Access

Figure 5.10: Displaying Downloaded Files for Authorized Access
6 Conclusion and Future Work

This project contributes to provide security to the data stored in the cloud, by encrypting the data before uploading into the cloud. As encryption consumes more processing overhead, many cloud service providers will have basic encryption applied only on few data fields. If cloud service providers can encrypt data, then cloud service can providers can decrypt encrypted data. To keep the cost low and maintain high sensitive data, it would be better to encrypt the data before uploading. In this project, we encrypt data using symmetric key encryption where private keys of the files will be stored in the local database. The system generates a single key for accessing multiple files. This Access key is stored in the cloud which further helps to retrieve private keys that are stored in the local database. As a single key is stored in the cloud for multiple files, flexibility will be increased for sharing any number of files, cost for key management will be reduced.

In future, Access key generation can be enhanced. If the Access key itself decrypts the files requested, it would reduce maintenance of private keys in the local database. File Modification techniques without downloading the file can be improved. The encryption technique can be enhanced further.
7 References


[3] "Software as a service,"


of Computer Science & Information Technology, June 2013.


[Accessed March 2015].


