**Enhancing The Data Integrity In Group Data Sharing For**

**Cloud Storage**

**Abstract**

With cloud storage services, users can remotely store their data to the cloud and realize the data sharing with others. Remote data integrity auditing is proposed to guarantee the integrity of the data stored in the cloud. In this project, we enhance the security of group data sharing. The sensitive information can be protected by encrypting the shared file, generating the signature and finally uploading the encrypted file on the cloud. Since we Propose a Multi-Level secure file sharing scheme for group data sharing in cloud, we Encrypt the data using two techniques attribute based key generation and ECC algorithm and also Providing access control for group user based on multi tenancy access control technique. Meanwhile, the proposed scheme is based on multi level secure file sharing; the security analysis and the performance evaluation show that the proposed scheme is secure and efficient.

**CHAPTER 1**

**INTRODUCTION**

* 1. **OVERVIEW OF THE PROJECT**
     1. **What is cloud?**

Cloud computing involves deploying group of remote servers and software networked that allow centralized data storage and online access to computer services or resources. Cloud computing continues to be one of most hyped subject in IT, but it is becoming more and more integral concept in IT overall. Cloud computing providers offer their services according to several fundamental models**.**

**1.1.2 Cloud Computing**

Cloud computing is the delivery of different services through the internet. These resources include tools and applications like data storage, servers, databases, networking, and software. Rather than keeping files on a proprietary hard drive or local storage device, cloud-based storage makes it possible to save them to a remote database. As long as an electronic device has access to the web, it has access to the data and the software programs to run it. Cloud computing is a popular option for people and businesses for a number of reasons including cost savings, increased productivity, speed and efficiency, performance, and security.

* 1. **Cloud Security**

Cloud security refers to the set of procedures, processes and standards designed to provide information security assurance in a cloud environment. Cloud security is important for both business and personal users. Everyone wants to know that their information is safe and secure and also many businesses have legal obligations to keep client data secure, with certain sectors having rules about data storage.

Cloud based applications are convenient for many business; they enable secure data management, analysis and access from anywhere. Security features which involves advanced configuration, automated encryption and access control. Some of the security risks which include storing data without encryption or lack of multifactor authentication to access the service. Cloud computing is mainly used for resource sharing and on demand services.

Cloud Providers provides a fundamental service is data storage (Storage as-a service).An organization allows its group members in the same group or department to store and share files in the cloud. Data security is a major issue in cloud

Issues in cloud:

* + Access control
  + Data integrity
  + Confidentiality
  + Data Privacy
  + Key management

**1.2.1Access control**

Access control is a way of limiting access to a system or to physical or virtual resources. In computing, access control is a process by which users are granted access and certain privileges to systems, resources or information.

In access control systems, users must present credentials before they can be granted access. In physical systems, these credentials may come in many forms, but credentials that can't be transferred provide the most security.

**There are three factors that can be used for authentication:**

1. Something only known to the user, such as a password or PIN

2. Something that is part of the user, such as a fingerprint, retina scan or another

Biometric measurement

3. Something that belongs to the user, such as a card or a key

In the fields of physical security and information security, access control (AC) is the selective restriction of access to a place or other resource. The act of accessing may mean consuming, entering, or using. Permission to access a resource is called authorization. Locks and login credentials are two analogous mechanisms of access control.

In object-oriented programming languages, access control is a part of the apparatus of achieving encapsulation, one of four fundamentals of object-oriented programming. The goal is to establish a clear separation between interface (visible and accessible parts of the class) and implementation (internal representation and helper methods).

**1.2.2 Data integrity**

Data quality is referred to as “data integrity.” It is maintaining and assuring the accuracy and consistency of data over its entire life-cycle. Data integrity means that the data is accurate and reliable.

Data integrity is the maintenance of, and the assurance of the accuracy and consistency of, data over its entire life-cycle, and is a critical aspect to the design, implementation and usage of any system which stores, processes, or retrieves data.

Any unintended changes to data as the result of a storage, retrieval or processing operation, including malicious intent, unexpected hardware failure, and human error, is failure of data integrity. If the changes are the result of unauthorized access, it may also be a failure of data security.

Data integrity is the overall completeness, accuracy and consistency of data. This can be indicated by the absence of alteration between two instances or between two updates of a data record, meaning data is intact and unchanged. Data integrity can be maintained through the use of error checking methods and validation procedures.

The concept of data integrity ensures that all data in a database can be traced and connected to other data. This ensures that everything is recoverable and searchable. Having a single, well-defined and well-controlled data integrity system increases stability and maintainability.

**1.2.3 Confidentiality**

Confidentiality refers to protecting information from being accessed by unauthorized parties. In other words, only the people who are authorized to do so can gain access to sensitive data. Data confidentiality is often a measure of ability of the system to protect its data. Accordingly, this is an integral component of security.

Data security in the cloud is ensured by the confidentiality of sensitive data. That can be done with the help of storing encrypted data on storage servers.

**1.2.4 Data Privacy**

Data privacy relates to how a piece of information—or data—should be handled based on its relative importance. In the digital age, we typically apply the concept of data privacy to critical personal information, also known as personally identifiable information (PII) and personal health information (PHI) or a business, data privacy goes beyond the PII of its employees and customers. It also includes the information that helps the company operate, whether its proprietary research and development data or financial information that shows how it’s spending and investing its money.

**Why is data privacy important?**

When data that should be kept private gets in the wrong hands, bad things can happen. A data breach at a government agency can, for example, put top secret information in the hands of an enemy state. A breach at a corporation can put proprietary data in the hands of a competitor. A breach at a school could put students’ PII in the hands of criminals who could commit identity theft. A breach at a hospital or doctor’s office can put PHI in the hands of those who might misuse it.

**1.2.5 Key Management**

Key management refers to management of cryptographic keys in a cryptosystem. This includes dealing with the generation, exchange, storage, use, crypto-shredding (destruction) and replacement of keys. It includes cryptographic protocol design, key servers, user procedures, and other relevant protocols. Key management concerns keys at the user level, either between users or systems. This is in contrast to key scheduling, which typically refers to the internal handling of keys within the operation of a cipher.

Key management is the process of administering or managing cryptographic keys for a cryptosystem. It involves the generation, creation, protection, storage, exchange, replacement and use of said keys and with another type of security system built into large cryptosystems, enables selective restriction for certain keys. In addition to access restriction, key management also involves the monitoring and recording of each key's access, use and context.

A critical cryptosystem component. Key management is also one of the most challenging aspects of cryptography because it deals with many types of security liabilities beyond encryption, such as people and flawed policies. It also involves creating a corresponding system policy, user training, interdepartmental interactions and proper coordination.

For a multicast group, security is a large issue, as all group members have the ability to receive the multicast message. The solution is a multicast group key management system, in which specific keys are securely provided to each member. In this manner, an encryption using a specific member’s key means that the message can only be accessed and read by that group member.

* 1. **Cloud Storage**

Cloud storage is a service model in which data is maintained, managed, backed up remotely and made available to users over a network (typically the Internet). Users generally pay for their cloud data storage on a per-consumption, monthly rate. Although the per-gigabyte cost has been radically driven down, cloud storage providers have added operating expenses that can make the technology more expensive than users bargained for. Cloud security continues to be a concern among users. Providers have tried to deal with those fears by building security capabilities, such as encryption and authentication, into their services.

* 1. **Cryptosystem**

In cryptography, a cryptosystem is a suite of cryptographic algorithms needed to implement a particular security service, most commonly for achieving confidentiality (encryption). Typically, a cryptosystem consists of three algorithms: one for key generation, one for encryption, and one for decryption. The term cipher (sometimes cipher) is often used to refer to a pair of algorithms, one for encryption and one for decryption. Therefore, the term cryptosystem is most often used when the key generation algorithm is important. For this reason, the term cryptosystem is commonly used to refer to public key techniques; however both "cipher" and "cryptosystem" are used for symmetric key techniques.

* 1. **Techniques**
     1. **Attribute Based Encryption**

Attribute-based encryption is a type of public-key encryption in which the secret key of a user and the cipher text are dependent upon attributes. In some systems, the decryption of a cipher text is possible only if the set of attributes of the user key matches the attributes of the cipher text.

**Types of attribute-based encryption**

There are mainly two types of attribute-based encryption schemes: Key-policy attribute-based encryption (KP-ABE) and cipher text-policy attribute-based encryption (CP-ABE).

In KP-ABE, users' secret keys are generated based on an access tree that defines the privileges scope of the concerned user and data are encrypted over a set of attributes. However, CP-ABE uses access trees to encrypt data and users' secret keys are generated over a set of attributes.

Attribute-based encryption (ABE) can be used for log encryption. Instead of encrypting each part of a log with the keys of all recipients, it is possible to encrypt the log only with attributes which match recipients' attributes. This primitive can also be used for broadcast encryption in order to decrease the number of keys used.

**1.5.2 ELLIPTIC CURVE CRYPTOGRAPHY**

Elliptic-curve cryptography (ECC) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. ECC requires smaller keys compared to non-EC cryptography (based on plain Galois fields) to provide equivalent security.

Elliptic curves are applicable for key agreement, digital signatures, pseudo-random generators and other tasks. Indirectly, they can be used for encryption by combining the key agreement with a symmetric encryption scheme. They are also used in several integer factorization algorithms based on elliptic curves that have applications in cryptography, such as Lenstra elliptic-curve factorization.

Elliptical curve cryptography (ECC) is a public key encryption technique based on elliptic curve theory that can be used to create faster, smaller, and more efficient cryptographic keys. ECC generates keys through the properties of the elliptic curve equation instead of the traditional method of generation as the product of very large prime numbers. The technology can be used in conjunction with most public key encryption methods, such as RSA, and Diffie-Hellman. According to some researchers, ECC can yield a level of security with a 164-bit key that other systems require a 1,024-bit key to achieve. Because ECC helps to establish equivalent security with lower computing power and battery resource usage, it is becoming widely used for mobile applications.

**Key Benefits of ECC**

ECC key is very helpful for the current generation as more people are moving to the Smartphone. As the utilization of Smartphone extends to grow, there is an emerging need for a more flexible encryption for business to meet with increasing security requirements.

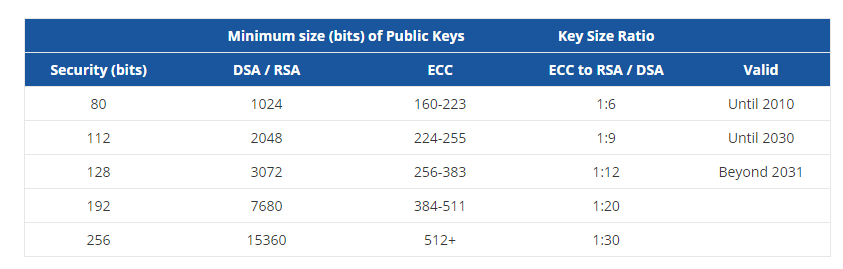
**Stronger Keys**

ECC stands for Elliptic Curve Cryptography is the latest encryption method offers stronger security. If we compare to the RSA and DSA algorithms, then 256-bit ECC is equal to 3072-bit RSA key. The reason behind keeping short key is the use of less computational power, fast and secures connection, ideal for Smartphone and tablet too.

The US government and the National Security Agency have certified ECC encryption method. The mathematical problem of the ECC algorithm, It is harder to break for hackers compare to RSA and DSA, which means the ECC algorithm ensures web site and infrastructure safety than traditional methods in a more secure manner.

**Shorter Key Size**

The elliptic curve cryptography (ECC) certificates allow key size to remain small while providing a higher level of security. ECC certificates key creation method is entirely different from previous algorithms, while relying on the use of a public key for encryption and a private key for decryption. By starting small and with a slow growth potential, ECC has longer potential lifespan. Elliptic curves are likely to be the next generation of cryptographic algorithms, and we are seeing the beginning of their use now.



If we examine the above table, there is a considerable growth in DSA and RSA key than ECC key size. A longer key requires more space, more bandwidth, and additional processor power. Even, it will take a time to generate a key, encrypt data, and decrypt the data.

**Why Elliptic Curve Cryptography is required?**

Encryption experts are pressed to find ever more effective methods, measured in security and performance, because the threats presented by hackers are ever greater – partly because the hackers themselves become more sophisticated in their attacks, and also because the fallout from an attack gets ever more dangerous as our use of data grows. It creates an urgency of new algorithms with a goal to provide a higher level of security by having keys that are more difficult to break, while offering better performance across the network and while working with large data sets.

Several factors are contributing to its increasing popularity. First of all, the security of 1024-bit encryption is degrading, due to faster computing and a better understanding and analysis of encryption methods. While brute force is still unlikely to crack 1024-bit encryption, other approaches, including highly intensive parallel computing in distributed computing arrays, are resulting in more sophisticated attacks. These attacks have reduced the effectiveness of this level of security. Even 2048-bit encryption is estimated by the RSA Security to be effective only until 2030.

**Web standards:**

Business owner has to mull over web server standards. Many web servers running on a single domain name can handle RSA, DSA, and ECC configuration. On the other side, few web servers do not have the ability to handle multiple algorithms and can utilize a single certificate on a single web server.

**Authentication speed:**

RSA, DSA, and ECC have diverse velocity for verification and authentication. RSA is a rapid algorithm in terms of client authentication while ECC is faster in terms of server authentication. Signature verification is rapidly in case of RSA key comparing to ECC key. There are transaction types, the processing power of the device; storage capacity, bandwidth, and consumption of power also influence the algorithm selection.

**Customer’s identity:**

Many government entities have started to accept DSA and ECC. They required for government subcontracts, government branches for their internal exchange of communication.

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 Identity-based remote data integrity checking with perfect**

**Data privacy preserving for cloud storage**

**Author**: Y. Yu, M. H. Au, G. Ateniese, X. Huang, W. Susilo, Y. Dai, and G. Min

Remote data integrity checking (RDIC) enables a data storage server, says a cloud server, to prove to a verifier that it is actually storing a data owner's data honestly. In this paper, they propose a new construction of identity-based (ID-based) RDIC protocol by making use of key- homomorphism cryptographic primitive to reduce the system complexity and the cost for establishing and managing the public key authentication framework in PKI based RDIC schemes. The proposed ID-based RDIC protocol leaks no information of the stored data to the verifier during the RDIC process. The new construction is proven secure against the malicious server in the generic group model and achieves zero knowledge privacy against a verifier.

**2.2 Checking integrity of data and recovery in the cloud**

**Environment**

**Author**: Neha Narayan Kulkarni, Shitalkumar A. Jain

Cloud computing provides many services to access them dynamically over the internet as per the user's demand. The data is growing in tremendous amount and it should be managed correctly where storage service proves efficient. The Data stored online can be hacked by the third party so to secure this data verifying integrity is essential. The proposed system of this paper is having the feature of verifying data integrity using Identity Based Remote Data Integrity Checking and recovery using the XOR operation. When the data is unavailable to the user, the proposed system provides flexibility for the user to regain data from the remote server.

**2.3 Data Integrity and Privacy Model in Cloud Computing**

**Author:** Mohammed Faez Al-Jaberi and Anazida Zainal

Cloud computing is the future of computing industry and it is believed to be the next generation of computing technology. Among the major concern in cloud computing is data integrity and privacy. This paper proposes an architecture based model that provides data integrity verification and privacy

Preserving in cloud computing. This model reduces the burden of checking the integrity of data stored in cloud storage by utilizing a third party , integrity checking service, and applies security mechanism that ensure privacy and confidentiality of data stored in cloud computing.

**2.4 Enabling Efficient User Revocation in Identity-based Cloud Storage Auditing for Shared Big Data**

**Author**: Y. Zhang, J. Yu, R. Hao, C. Wang, and K. Ren

In this paper, they propose a novel storage auditing scheme that achieves highly-efficient user revocation independent of the total number of file blocks possessed by the revoked user in the cloud. This is achieved by exploring a novel strategy for key generation and a new private key update technique.

Using this strategy and the technique, we realize user revocation by just updating the no revoked group users' private keys rather than authenticators of the revoked user. The integrity auditing of the revoked user's data can still be correctly performed when the authenticators are not updated. Meanwhile, the proposed scheme is based on identity-base cryptography, which eliminates the complicated certificate management in traditional Public Key Infrastructure (PKI) systems. The security and efficiency of the proposed scheme are validated via both analysis and experimental results.

**2.5 Maintaining data integrity for shared data in cloud**

Qian Wang et.al. has proposed a model to solve the problem of integrity of data stored in the cloud. The TPA has allowed verifying data in cloud storage through auditing process and motivating public auditing system in the cloud. TPA check the outsourced data integrity. The advantages of auditing are to detect, prevent errors and maintain the database regularly. Auditing should not bring any new vulnerability towards privacy of data. Based on the proxy re-signature method designs a public auditing scheme for data storage with proficient user revocation in cloud. The original user can acts as a group manager and able to retract the users if necessary. For each block of data to be stored in cloud server, data owner is assigned with a signature and the integrity of data relies in the correctness of all the signatures. In a cloud if a user modifies a single block including insertion or deletion, the index of the modified block is changed and the user needs to compute a new signature for the modified block. User access the modified data with the new signature generated to perform.

**2.6 Secure Multi-Owner Data Sharing for Dynamic Groups in the Cloud**

With the character of low maintenance, cloud computing provides an economical and efficient solution for sharing group resource among cloud users. In this paper, author proposed a secure multi owner data sharing scheme, named Mona, for dynamic groups in the cloud. By leveraging group signature and dynamic broadcast encryption techniques, any cloud user can anonymously share data with others. Meanwhile, the storage overhead and encryption computation cost of this scheme are independent with the number of revoked users. In addition, author analyze the security of our scheme with rigorous proofs, and demonstrate the efficiency of our scheme in experiments.

**2.7 Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage**

Data sharing is an important functionality in cloud storage. In this article, we show how to securely, efficiently, and flexibly share data with others in cloud storage. Authors describe new public-key cryptosystems which produce constant size cipher texts such that efficient delegation of decryption rights for any set of cipher texts are possible. The novelty is that one can aggregate any set of secret keys and make them as compact as a single key, but encompassing the power of all the keys being aggregated. In other words, the secret key holder can release a constant-size aggregate key for flexible choices of cipher text set in cloud storage, but the other encrypted files outside the set remain confidential. This compact aggregate key can be conveniently sent to others or be stored in a smart card with very limited secure storage. Authors provide formal security analysis of this scheme in the standard model. Authors also describe other application of this scheme. In particular, these schemes give the first public-key patient-controlled encryption for flexible hierarchy, which was yet to be known.

**2.8** **Practical Techniques for Searches on Encrypted Data**

It is desirable to store data on data storage servers such as mail servers and file servers in encrypted form to reduce security and privacy risks. But this usually implies that one has to sacrifice functionality for security. For example, if a client wishes to retrieve only documents containing certain words, it was not previously known how to let the data storage server perform the search and answer the query without loss of data confidentiality. In this paper, author describes these cryptographic schemes for the problem of searching on encrypted data and provides proofs of security for the resulting crypto systems. This technique has a number of crucial advantages. They are provably secure: they provide provable secrecy for encryption, in the sense that the entrusted server cannot learn anything about the plaintext when only given the cipher text; they provide query isolation for searches, meaning that the entrusted server cannot learn anything more about the plaintext than the search result; they provide controlled searching, so that the entrusted server cannot search for an arbitrary word without the user’s authorization; they also support hidden queries, so that the user may ask the entrusted server to search for a secret word without revealing the word to the server.

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 Existing System:**

To preserve data privacy, a basic solution is to encrypt data files, and then upload the encrypted data into the cloud. Unfortunately, designing an efficient and secure data sharing scheme for groups in the cloud is not an easy task. In the existing System data owners store the encrypted data files in untrusted storage and distribute the corresponding decryption keys only to authorized users. Thus, unauthorized users as well as storage servers cannot learn the content of the data files because they have no knowledge of the decryption keys. However, the complexities of user participation and revocation in these schemes are linearly increasing with the number of data owners and the number of revoked users, respectively.

**DISADVANTAGES OF EXISTING SYSTEM**

In the existing Systems, identity privacy is one of the most significant obstacles for the wide deployment of cloud computing. Without the guarantee of identity privacy, users may be unwilling to join in cloud computing systems because their real identities could be easily disclosed to cloud providers and attackers. On the other hand, unconditional identity privacy may incur the abuse of privacy. For example, a misbehaved staff can deceive others in the company by sharing false files without being traceable. Only the group manager can store and modify data in the cloud .The changes of membership make secure data sharing extremely difficult the issue of user revocation is not addressed.

**3.2 PROPOSED SYSTEM**

We propose a secure multi-owner data sharing scheme. It implies that any user in the group can securely share data with others by the entrusted cloud. Our proposed scheme is able to support dynamic groups efficiently. Specifically, new granted users can directly decrypt data files uploaded before their participation without contacting with data owners. User revocation can be easily achieved through a novel revocation list without updating the secret keys of the remaining users. The size and computation overhead of encryption are constant and independent with the number of revoked users. We provide secure and privacy-preserving access control to users, which guarantees any member in a group to anonymously utilize the cloud resource. Moreover, the real identities of data owners can be revealed by the group manager when disputes occur. We provide rigorous security analysis, and perform extensive simulations to demonstrate the efficiency of our scheme in terms of storage and computation overhead.

**Advantages Of Proposed System**

1. Any user in the group can store and share data files with others by the cloud.

2. The encryption complexity and size of cipher texts are independent with the number of revoked users in the system.

3. User revocation can be achieved without updating the private keys of the remaining users.

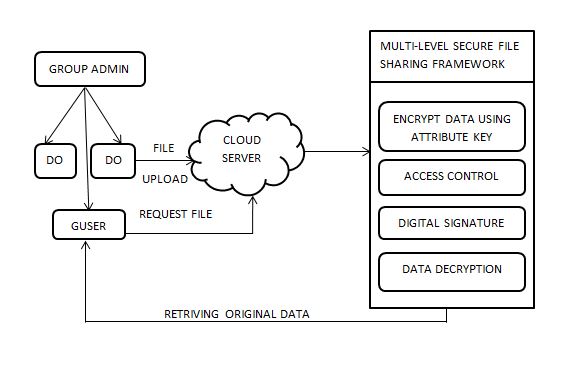
4. A new user can directly decrypt the files stored in the cloud before his participation.

5. Third level of security mechanism included for existing and new user as follows, when user logins with the help of their login credentials , their next step is download a file from the cloud by using the group secret key.

6. Instead of using group secret key individual secret key is generated and sent to their mobile with the help of this key user can download the file from the cloud this process repeats every time when user login due to this ,security level is increased and also avoids unauthorized user to access the group file.

**CHAPTER 4**

**SYSTEM DESIGN**

****

**FIGURE 4.1 PROPOSED ARCHITECTURAL DESIGN**

**4.1 DropBox:**

Drop box is a personal cloud storage service (sometimes referred to as an online backup service) that is frequently used for file sharing and collaboration. The Drop box application is available for Windows, Macintosh and Linux desktop operating systems. There are also apps for iPhone, iPod, Android, and BlackBerry devices. The service provides 2 gigabytes (GB) of storage for free and up to 100 GB on various for-fee plans. Another option, Drop box for Teams, provides 350 GB storage. The user data is stored on Amazon’s Simple Storage Service (S3) and protected with Secure Sockets Layer (SSL) and Advanced Encryption System (AES) 256-bit encryption.

After installation of the associated application, a Drop box folder appears with the user’s other folders. Users can save files to the folder, add new folders, and drag and drop files among folders just as if they were all local. Files in the Drop box folder can be accessed from anywhere with an Internet connection – the user just has to log in to his account to upload, download and share files.

1. **2 Cloud Storage:**

Cloud storage is a model of computer data storage in which the digital data is stored in logical pools. The physical storage spans multiple servers (sometimes in multiple locations), and the physical environment is typically owned and managed by a hosting company. These cloud storage providers are responsible for keeping the data available and accessible, and the physical environment protected and running. People and organizations buy or lease storage capacity from the providers to store user, organization, or application data.

Cloud storage services may be accessed through a collocated cloud computing service, a web service application programming interface (API) or by applications that utilize the API, such as cloud desktop storage, a cloud storage gateway or Web-based content management systems.

**4.3 Cloud Service Provider:**

A cloud service provider, or CSP, is a company that offers some component of cloud computing -- typically infrastructure as a service (IaaS), software as a service (SaaS) or platform as a service (PaaS) -- to other businesses or individuals. Cloud service providers (CSP) are companies that offer network services, infrastructure, or business applications in the cloud. The cloud services are hosted in a data centre that can be accessed by companies or individuals using network connectivity.

**4.4 Encryption:**

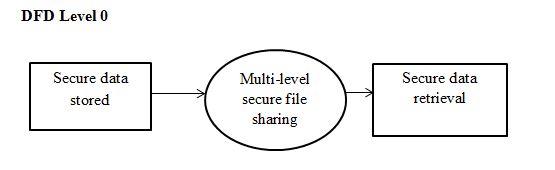
Encryption is the process of using an algorithm to transform information to make it unreadable for unauthorized users. This cryptographic method protects sensitive data such as credit card numbers by encoding and transforming information into unreadable cipher text. This encoded data may only be decrypted or made readable with a key. Symmetric-key and asymmetric-key are the two primary types of encryption.

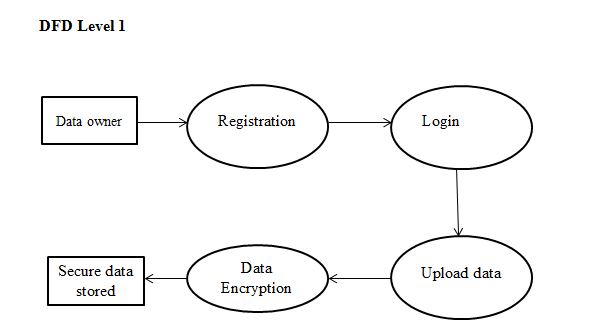
Encryption is essential for ensured and trusted delivery of sensitive information.

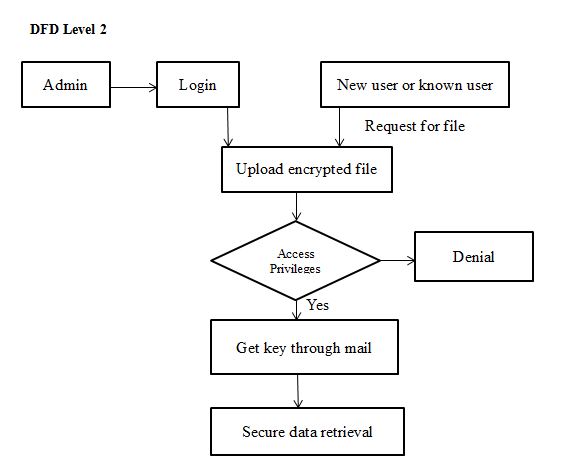
**4.5 Decryption:**

Decryption is generally the reverse process of encryption. It is the process of decoding the data which has been encrypted into a secret format. An authorized user can only decrypt data because decryption requires a secret key or password. Decryption is the process of decoding encrypted information so that is can be accessed again by authorized users.

**4.6 Data Flow Diagram**

****

****

****

Above Dataflow diagram describes the process of this project.

**CHAPTER 5**

**SYSTEM REQUIRMENTS**

This project concept can implement any platform and any device. Here demo of the project implement in Android Phone for user interface.

**5.1 Hardware requirements**

* Processor - Intel(R) Core(TM) i3 CPU M380
* Speed -2.53GHz
* RAM -4.00GB
* Keyboard -Standard Windows keyboard
* Mouse -Two or three button mouse
* Monitor -SVGA

**5.2 Software requirements**

* Operating System –Windows 10
* Application Server-Tomcat 5.0
* Front end -Java
* IDE -Net Beans 8.0.2
* Back end -My SQL

**5.2.1 Platform**

In this project Windows 10 OS is a platform for visualize demo of the project. If you interest any kind of file need to encrypt, you can use this project algorithm.

**5.2.2 Net beans IDE 8.0.2**

Net Beans is an integrated development environment (IDE) for Java. Net Beans allows applications to be developed from a set of modular software components called modules. ... Applications based on Net Beans, including the Net Beans IDE, can be extended by third party developers.An integrated Development Environment or IDE for developing primarily with Java, Net beans can also be used for other languages like HTML5, PHP, C or C++ and others. Net beans is also an application platform framework for developing and testing Java Desktop applications and others too.

**CHAPTER 6**

**IMPLEMENTATION**

**6.1 SOURCE CODE**

**Home Page Login**

<%@ Master Language="C#" AutoEventWireup="true" CodeFile="Home.master.cs" Inherits="Home" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<meta name="description" content="" />

<meta name="keywords" content="" />

<title>Group Data Sharing</title>

<meta http-equiv="content-type" content="text/html; charset=utf-8" />

<link rel="stylesheet" type="text/css" href="style.css" />

</head>

<body>

<div id="wrapper">

<div id="header">

<div id="logo">

<h1><a href="#">Group Data Sharing</a></h1>

</div>

</div>

<div id="menu">

<ul>

<li class="first current\_page\_item"><a href="Default.aspx">Homepage</a></li>

<li><a href="AdminLogin.aspx">AdminLogin</a></li>

<li><a href="ManagerLogin.aspx">ManagerLogin</a></li>

<li><a href="AsstManagerLogin.aspx">AsstManagerLogin</a></li>

<li class="last"><a href="User.aspx">UserLogin</a></li>

</ul>

<br class="clearfix" />

</div>

<%--<div id="splash">

<img class="pic" src="images/pic01.jpg" width="870" height="230" alt="" />

</div>--%>

<br class="clearfix" />

<form id="form1" runat="server">

<div>

<asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">

</asp:ContentPlaceHolder>

</div>

</form>

<br class="clearfix" />

</div>

<div id="footer">

&copy; Group Data Sharing <a href="#" rel="nofollow">Website</a>.

</div>

</body>

</html>

**Admin Login**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.SqlClient;

public partial class AdminLogin : System.Web.UI.Page

{

SqlConnection con = new SqlConnection(@"Data Source=.\SQLEXPRESS;AttachDbFilename=|DataDirectory|\Dataindb.mdf;Integrated Security=True;User Instance=True");

SqlCommand cmd;

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void Button1\_Click(object sender, EventArgs e)

{

if (TextBox1.Text == "admin" & TextBox2.Text == "admin")

{

Response.Redirect("Adminhome.aspx");

}

else

{

Response.Write("<Script> alert('Password Mismatch!') </Script>");

}

}

protected void Button2\_Click(object sender, EventArgs e)

{

TextBox1.Text = "";

TextBox2.Text = "";

}

}

**Manager Login**

<%@ Page Title="" Language="C#" MasterPageFile="~/Home.master" AutoEventWireup="true" CodeFile="ManagerLogin.aspx.cs" Inherits="ManagerLogin" %>

<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">

<p>

<br />

</p>

<table style="width: 100%">

<tr>

<td style="width: 279px">

&nbsp;</td>

<td colspan="2">

<strong>

<asp:Label ID="Label1" runat="server" Text="Manager Login"></asp:Label>

</strong>

</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

&nbsp;</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

<asp:Label ID="Label2" runat="server" Text="User Name"></asp:Label>

</td>

<td>

<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

<asp:Label ID="Label3" runat="server" Text="Password"></asp:Label>

</td>

<td>

<asp:TextBox ID="TextBox2" runat="server" TextMode="Password"></asp:TextBox>

</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

<asp:HyperLink ID="HyperLink1" runat="server" NavigateUrl="~/NewMangager.aspx">New Manager</asp:HyperLink>

</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

<asp:Button ID="Button1" runat="server" Text="Login" onclick="Button1\_Click" />

&nbsp;<asp:Button ID="Button2" runat="server" Text="Clear" onclick="Button2\_Click" />

</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

&nbsp;</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

&nbsp;</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

&nbsp;</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

&nbsp;</td>

<td>

&nbsp;</td>

</tr>

</table>

<p>

</p>

<p>

</p>

<p>

</p>

</asp:Content>

**AssistantManager Login**

<%@ Page Title="" Language="C#" MasterPageFile="~/Home.master" AutoEventWireup="true" CodeFile="AsstManagerLogin.aspx.cs" Inherits="AsstManagerLogin" %>

<asp:Content ID="Content1" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">

<p>

<br />

</p>

<p>

</p>

<table style="width: 100%">

<tr>

<td style="width: 279px">

&nbsp;</td>

<td colspan="2">

<strong>

<asp:Label ID="Label1" runat="server" Text="Asst.Manager Login"></asp:Label>

</strong>

</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

&nbsp;</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

<asp:Label ID="Label2" runat="server" Text="User Name"></asp:Label>

</td>

<td>

<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

<asp:Label ID="Label3" runat="server" Text="Password"></asp:Label>

</td>

<td>

<asp:TextBox ID="TextBox2" runat="server" TextMode="Password"></asp:TextBox>

</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

<asp:HyperLink ID="HyperLink1" runat="server" NavigateUrl="~/NewAsst.aspx">New Asst.Manager</asp:HyperLink>

</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

<asp:Button ID="Button1" runat="server" Text="Login" onclick="Button1\_Click" />

&nbsp;<asp:Button ID="Button2" runat="server" Text="Clear" onclick="Button2\_Click" />

</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

&nbsp;</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

&nbsp;</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

&nbsp;</td>

<td>

&nbsp;</td>

</tr>

<tr>

<td style="width: 279px">

&nbsp;</td>

<td style="width: 142px">

&nbsp;</td>

<td>

&nbsp;</td>

<td>

&nbsp;</td>

</tr>

</table>

<p>

</p>

<p>

</p>

</asp:Content>

**UserFileApproved**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data;

using System.Data.SqlClient;

using System.Net.Mail;

using System.Net;

public partial class MUserApproved : System.Web.UI.Page

{

SqlConnection con = new SqlConnection(@"Data Source=.\SQLEXPRESS;AttachDbFilename=|DataDirectory|\Dataindb.mdf;Integrated Security=True;User Instance=True");

SqlCommand cmd;

protected void Page\_Load(object sender, EventArgs e)

{

bind();

}

string altermail;

protected void lnkView\_Click(object sender, EventArgs e)

{

GridViewRow grdrow = (GridViewRow)((LinkButton)sender).NamingContainer;

string id = grdrow.Cells[0].Text;

con.Open();

cmd = new SqlCommand("select \* from usertb where id='"+id+"' ", con);

SqlDataReader dr = cmd.ExecuteReader();

if (dr.Read())

{

altermail = dr["Email"].ToString();

}

con.Close();

Random rr=new Random ();

int key=rr.Next (1111,9999);

string to = altermail;

string from = "sampletest685@gmail.com";

// string subject = "Key";

// string body = TextBox1.Text;

string password = "mailtest2";

using (MailMessage mm = new MailMessage(from, to))

{

mm.Subject = "Login Key";

mm.Body = "Your Login Keys " + key;

//if (fuAttachment.HasFile)

//{

//string FileName = Server.MapPath(ff);

//mm.Attachments.Add(new Attachment(FileName));

//}

mm.IsBodyHtml = false;

SmtpClient smtp = new SmtpClient();

smtp.Host = "smtp.gmail.com";

smtp.EnableSsl = true;

NetworkCredential NetworkCred = new NetworkCredential(from, password);

smtp.UseDefaultCredentials = true;

smtp.Credentials = NetworkCred;

smtp.Port = 587;

smtp.Send(mm);

ClientScript.RegisterStartupScript(GetType(), "alert", "alert('Email has sent.');", true);

}

con.Open();

cmd = new SqlCommand("update usertb set Status='" + RadioButtonList1.SelectedItem.Text + "',keys='" + key + "' where id='" + id + "' ", con);

cmd.ExecuteNonQuery();

con.Close();

bind();

}

public static string Publickey(int length)

{

const string chars = "1234567890";

var random = new Random();

return new string(Enumerable.Repeat(chars, length)

.Select(s => s[random.Next(s.Length)]).ToArray());

}

protected void lnkView\_Click1(object sender, EventArgs e)

{

GridViewRow grdrow = (GridViewRow)((LinkButton)sender).NamingContainer;

string id = grdrow.Cells[0].Text;

con.Open();

cmd = new SqlCommand("update usertb set Status='Inactive' where id='" + id + "' ", con);

cmd.ExecuteNonQuery();

con.Close();

//Delete

con.Open();

cmd = new SqlCommand("Delete from usertb where id='" + id + "' ", con);

cmd.ExecuteNonQuery();

con.Close();

bind();

}

private void bind()

{

cmd = new SqlCommand("select \* from usertb Where Status='Waiting' ", con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataTable dt = new DataTable();

da.Fill(dt);

GridView1.DataSource = dt;

GridView1.DataBind();

cmd = new SqlCommand("select \* from usertb where Status !='Waiting'", con);

SqlDataAdapter da1 = new SqlDataAdapter(cmd);

DataTable dt1 = new DataTable();

da1.Fill(dt1);

GridView2.DataSource = dt1;

GridView2.DataBind();

cmd = new SqlCommand("select \* from usertb where Status !='Waiting'", con);

SqlDataAdapter da11 = new SqlDataAdapter(cmd);

DataTable dt11 = new DataTable();

da11.Fill(dt11);

GridView3.DataSource = dt11;

GridView3.DataBind();

}

protected void RadioButtonList1\_SelectedIndexChanged(object sender, EventArgs e)

{

}

}

**UserFileDownload**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data;

using System.Data.SqlClient;

using System.Security.Cryptography;

using System.IO;

using System.Net;

using System.Text;

public partial class UserFileDownload : System.Web.UI.Page

{

SqlConnection con = new SqlConnection(@"Data Source=.\SQLEXPRESS;AttachDbFilename=|DataDirectory|\Dataindb.mdf;Integrated Security=True;User Instance=True");

SqlCommand cmd;

string filename;

string passs = @"myKey123";

protected void Page\_Load(object sender, EventArgs e)

{

bind();

}

protected void lnkView\_Click(object sender, EventArgs e)

{

GridViewRow grdrow = (GridViewRow)((LinkButton)sender).NamingContainer;

string id = grdrow.Cells[0].Text;

if (TextBox1.Text == "")

{

Response.Write("<Script> alert('Please Enter Key') </Script>");

}

else

{

con.Open();

cmd = new SqlCommand("select \* from filetb where id='" + id + "' and Keys='" + TextBox1.Text + "'", con);

SqlDataReader dr1 = cmd.ExecuteReader();

if (dr1.Read())

{

//string aaa = dr1["FilePath"].ToString();

filename = dr1["FileName"].ToString();

string filePath1 = Server.MapPath("~/Encrypt/" + filename);

string filePath2 = Server.MapPath("~/Decrypt/" + filename);

DecryptFile(filePath1, filePath2);

string aaa = "~/Decrypt/" + filename;

if (aaa != string.Empty)

{

string filePath = aaa;

Response.ContentType = "doc/docx";

Response.AddHeader("Content-Disposition", "attachment;filename=\"" + aaa + "\"");

Response.TransmitFile(Server.MapPath(filePath));

Response.End();

}

}

else

{

Response.Write("<Script> alert('File Key Mismatch') </Script>");

}

con.Close();

Response.Write("<Script> alert('" + id + "') </Script>");

}

bind();

}

private void DecryptFile(string inputFile, string outputFile)

{

{

string keyss = TextBox1.Text;

string password = passs;

UnicodeEncoding UE = new UnicodeEncoding();

byte[] key = UE.GetBytes(password);

FileStream fsCrypt = new FileStream(inputFile, FileMode.Open);

RijndaelManaged RMCrypto = new RijndaelManaged();

CryptoStream cs = new CryptoStream(fsCrypt,

RMCrypto.CreateDecryptor(key, key),

CryptoStreamMode.Read);

FileStream fsOut = new FileStream(outputFile, FileMode.Create);

int data;

while ((data = cs.ReadByte()) != -1)

fsOut.WriteByte((byte)data);

fsOut.Close();

cs.Close();

fsCrypt.Close();

}

}

private void bind()

{

cmd = new SqlCommand("select \* from userfiletb where Status='waiting' and UserName='" + Session["uname"].ToString() + "'", con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataTable dt = new DataTable();

da.Fill(dt);

GridView1.DataSource = dt;

GridView1.DataBind();

cmd = new SqlCommand("select \* from userfiletb where Status='Approved' and UserName='" + Session["uname"].ToString() + "' ", con);

SqlDataAdapter da1 = new SqlDataAdapter(cmd);

DataTable dt1 = new DataTable();

da1.Fill(dt1);

GridView2.DataSource = dt1;

GridView2.DataBind();

}

}

**SCREENSHOTS:**

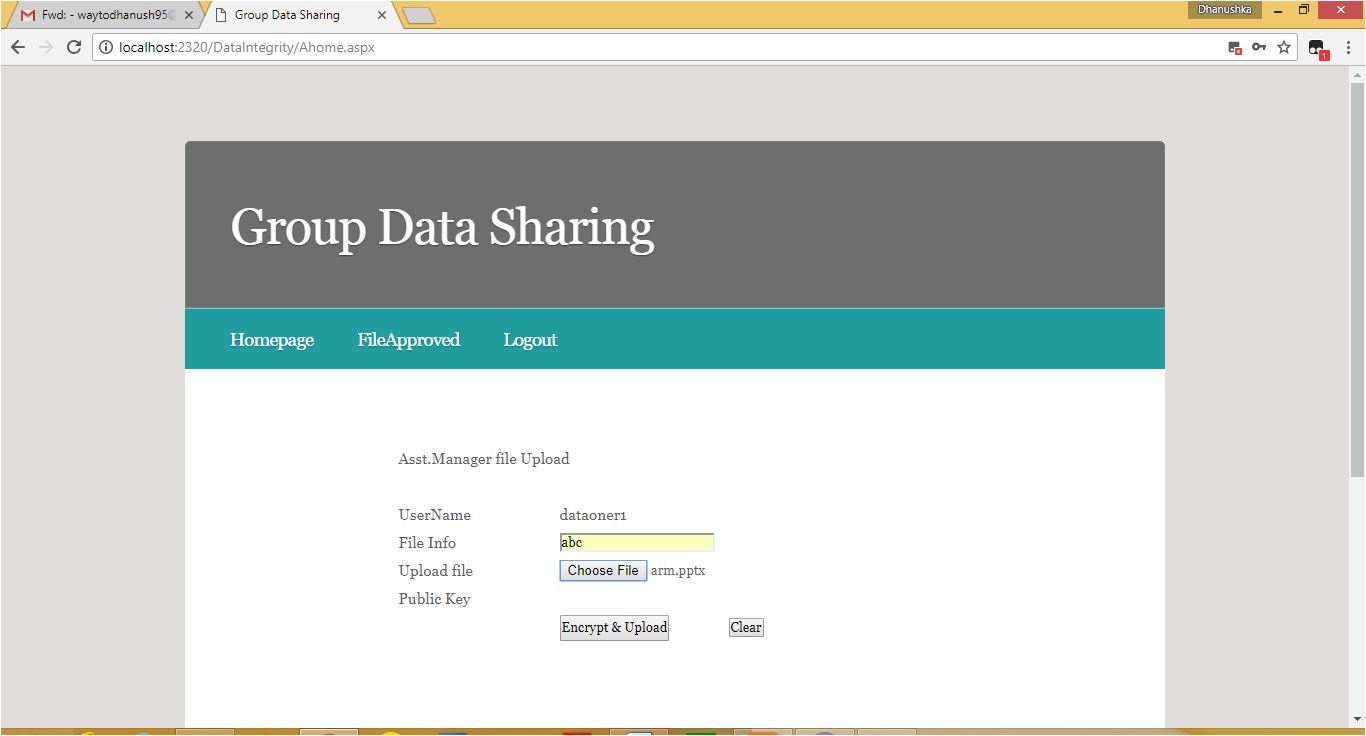


FIGURE 6.1 DATA ENCRYPTION

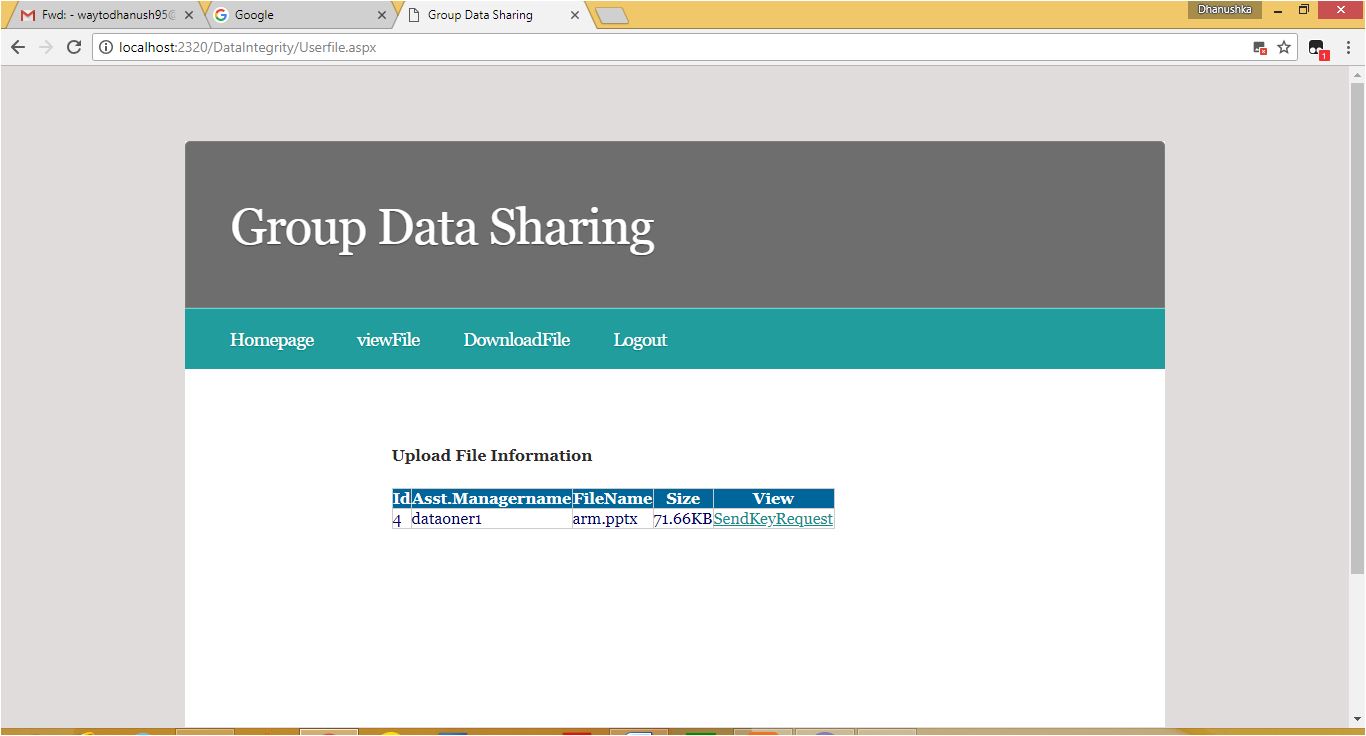


FIGURE 6.2 REQUEST FILE

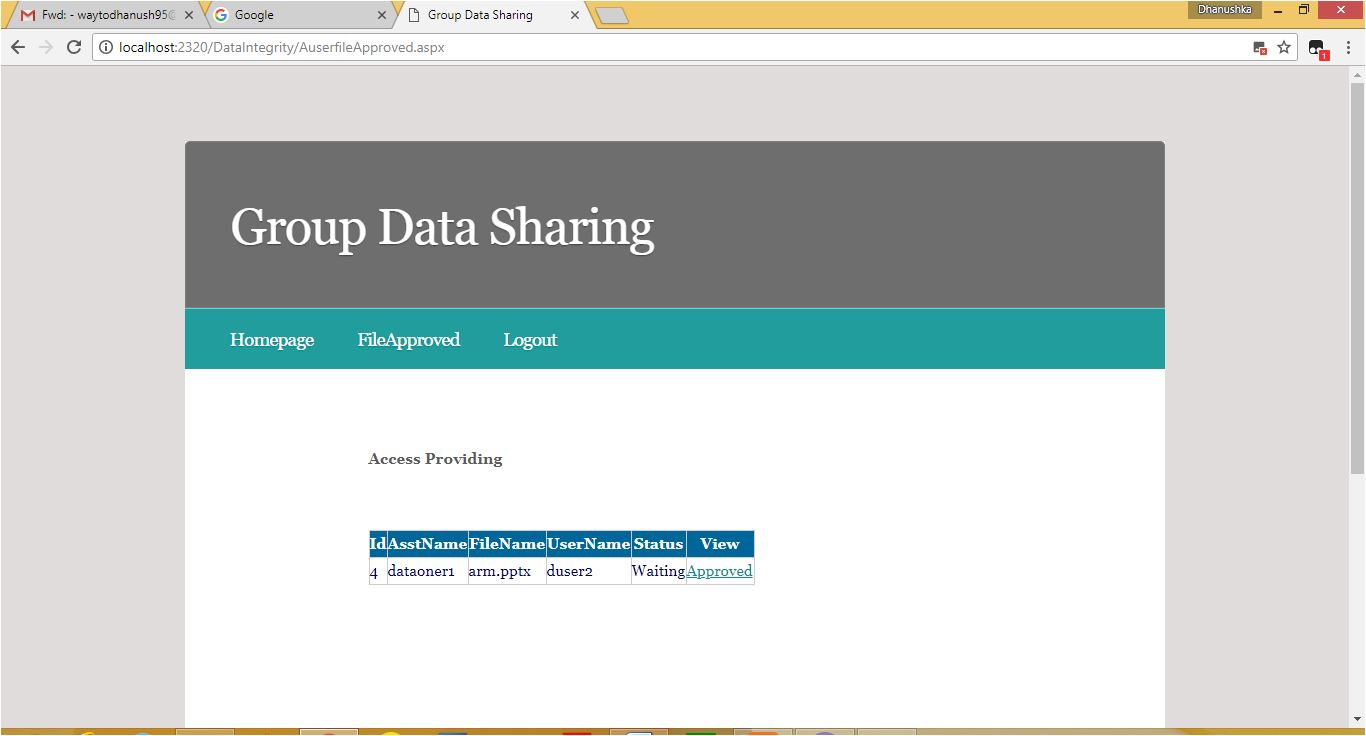


FIGURE 6.3 ACCESS PRIVILEDGES

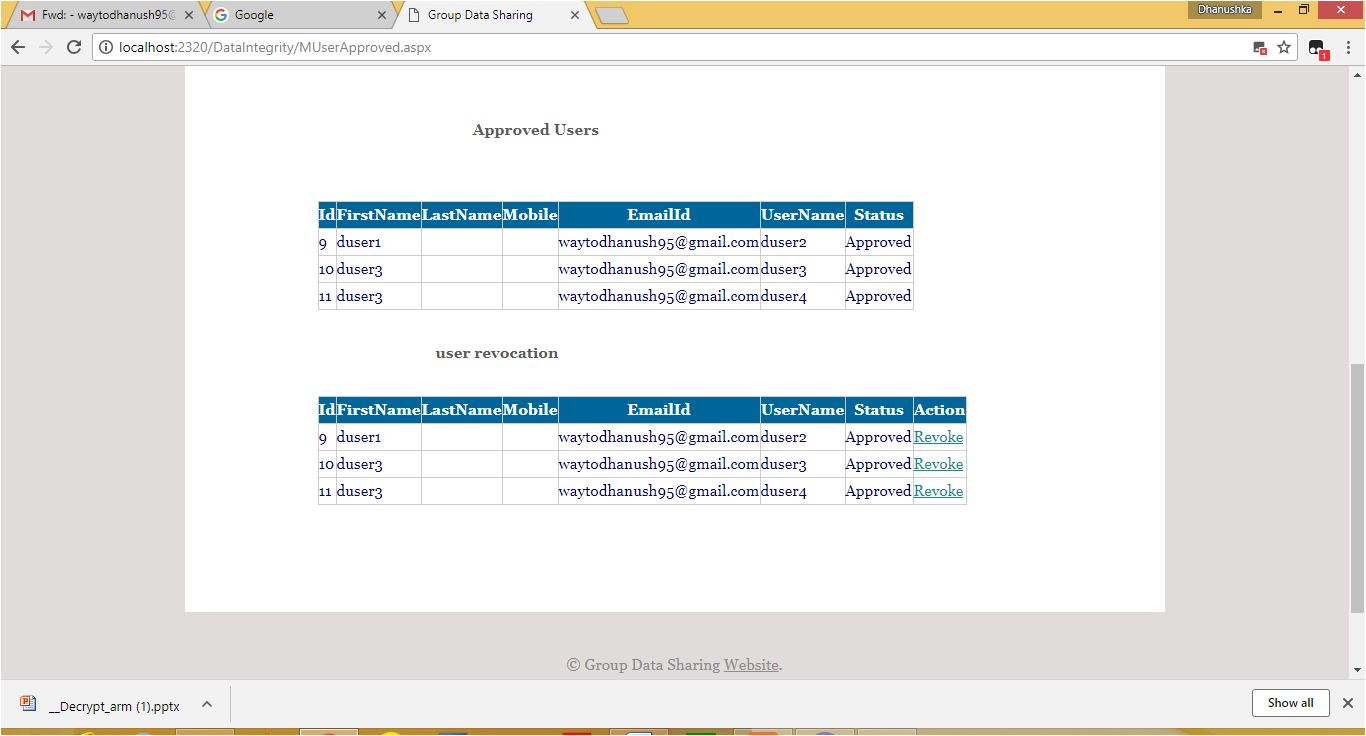


FIGURE 6.4 USER REVOCATION

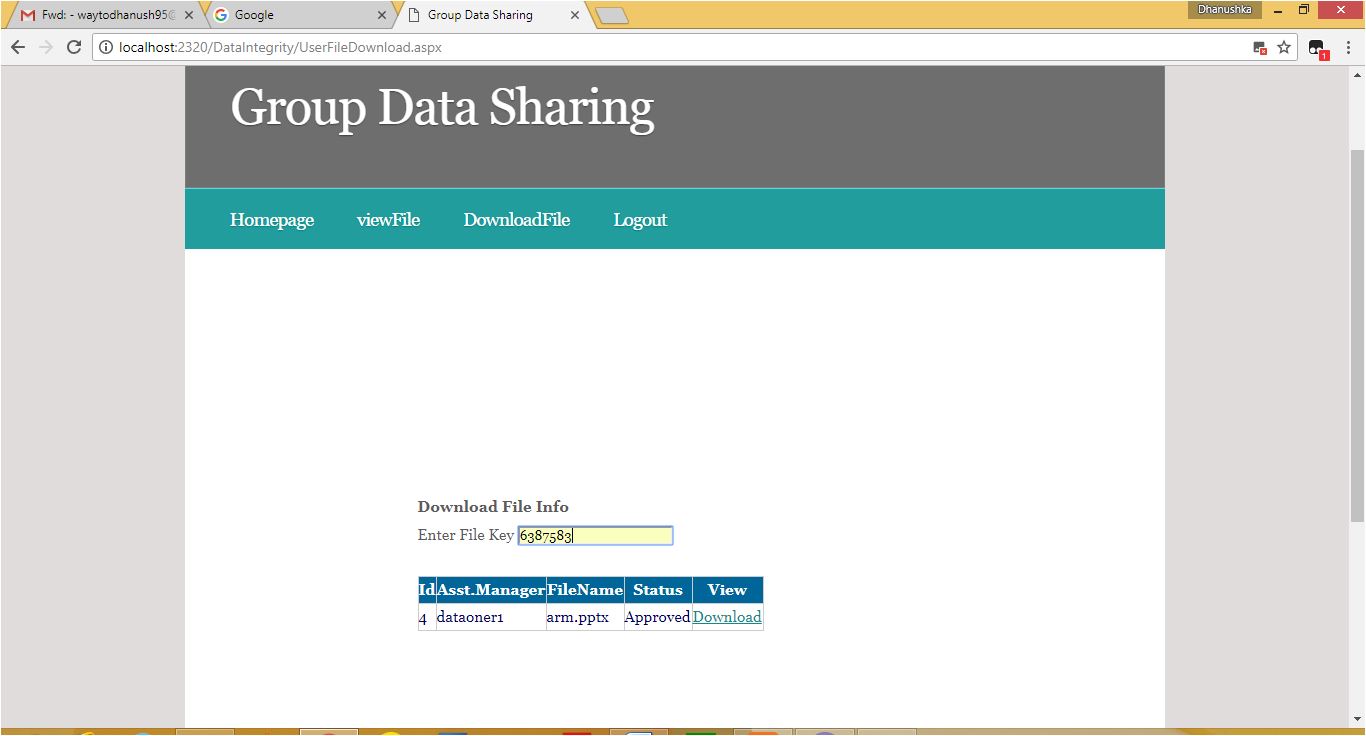


FIGURE 6.4 DATA DECRYPTION

**CHAPTER 7**

**CONCLUSION AND FUTURE WORK**

* 1. **CONCLUSION**

This work designs a secure data sharing scheme, for dynamic clusters in a un trusted cloud. A user is able to share data with others in the group without revealing identity privacy to the cloud. Supports secure data sharing in dynamic groups using Multi-level secure file sharing. Efficient user revocation achieved through public revocation list without updating the private keys of the remaining users.

**7.2 FUTURE WORK**

In future, for solving the reliability and scalability issues we further introduce the backup group manager. In case of any failures of group manager the backup group manager handles those problems. So that the reliability and scalability increases.

**7.3 REFEREENCES**

1. Y. Yu, M. H. Au, G. Ateniese, X. Huang, W. Susilo, Y. Dai, and G. Min, “Identity-based remote data integrity checking with perfect data privacy preserving for cloud storage,” IEEE Trans. Inform. Forensic Secur., vol. 12, no.4, pp. 767-778, Apr. 2017, DOI. 10.1109/TIFS.2016.261583.
2. H. Yan, J. Li, J. Han, and Y. Zhang, “A novel efficient remote data possession checking protocol in cloud storage,” IEEE Trans. Inform. Forensic Secur., vol. 12, no. 1, pp. 78-88, Jan. 2017, DOI. 10.1109/TIFS.2016.2601070.
3. Z. M. Zhu and R. Jiang, “A Secure Anti-Collusion Data Sharing Scheme for Dynamic Groups in the Cloud,” IEEE Trans. Parallel Distrib. Syst., vol. 27, no.1, pp. 40-50, Jan. 2016, DOI. 10.1109/TPDS.2015.2388446.
4. Jiawei Yuan and S. Yu, “Public integrity auditing for dynamic data sharing with multiuser modification,” IEEE Trans. Inform. Forensic Secur., vol. 10, no. 8, pp. 1717-1726, Aug. 2015, 10.1109/TIFS.2015.2423264.
5. B. Wang, B. Li, and H. Li, “Panda: public auditing for shared data with efficient user revocation in the cloud,” IEEE Trans. Serv. Comput., vol. 8, no. 1, pp-92-106, Jan. 2015, DOI. 10.1109/TSC.2013.2295611.
6. J. Y. Hwang, L. Chen, H. S. Cho, and D. Nyang, “Short dynamic group signature scheme supporting controllable linkability,” IEEE Trans. Inform Forensic Secur., vol. 10, no. 6, pp. 1109-1124, Jun. 2015, DOI. 10.1109/TIFS.2015.2390497.
7. C. Liu, J. Chen, L.T. Yang, X. Zhang, C. Yang, R. Ranjan, and R. Kotagiri, “Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates,” IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 9, pp. 2234-2244, Sep. 2014, DOI. 10.1109/TPDS.2013.191
8. Boyang Wang, B. Li, and H. Li, “Oruta: privacy-preserving public auditing for shared data in the cloud,” IEEE Trans. Cloud Comput., vol. 2, no. 1, pp. 43-56, Jan. 2014, DOI. 10.1109/TCC.2014.2299807
9. B. Y. Wang, B. C. Li, and H. Li, “Knox: Privacy-Preserving Auditing for Shared Data with Large Groups in the Cloud,” Proc. ACNS, Singapore, 2012, pp 507-525
10. S. Kamara, K. Lauter, “Cryptographic cloud storage,” Proc. FC, Tenerife, Canary Islands, Spain, 2010, pp. 136-149.