Aim of the project

This project aim to set up a secure layer for storing, retrieving and transfer of data across multiple users with data privacy , content privacy and user identity intact . proposed secrecy control to let cloud servers to control clients 'get' to help without knowing their character data.

The advocated plans can secure clients protection against every single expert. half way data revealed in secrecy control and no evidence showed in secrecy control.

1.1 Problem Solving

With the development of new computing paradigm, cloud computing becomes the most notable one, which provides convenient, on-demand services from a shared pool of configurable computing resources. Therefore, an increasing number of companies and individuals prefer to outsource their data storage to cloud server. Despite the tremendous economic and technical advantages, unpredictable security and privacy concerns become the most prominent problem that hinders the widespread adoption of data storage in public cloud infrastructure. Encryption is a fundamental method to protect data privacy in remote storage. However, how to effectively execute keyword search for plaintext becomes difficult for encrypted data due to the unreadability of ciphertext. Searchable encryption provides mechanism to enable keyword search over encrypted data for the file sharing system, such

Multi-owner multiuser scenario, fine-grained search authorization is a desirable function for the data owners to share their private data with another authorized user. The outsourced decryption method allows user to recover the message with ultra-lightweight decryption. However, the cloud server might return wrong half-decrypted information as a result of malicious attack or system malfunction. Thus, it is an important issue to guarantee the correctness of outsourced decryption in public key encryption with keyword search (PEKS) system. The authorized entities may illegally leak their secret key to a third party for profits. Suppose that a patient someday suddenly finds out that a secret key corresponding his electronic medical data is sold on e-Bay. Such despicable behaviour seriously threatens the patient’s data privacy. Even worse, if the private electronic health data that contain serious health disease is abused by the insurance company or the patient’s employment corporation, the patient would be declined to renew the medical insurance or labour contracts. The intentional secret key leakage seriously undermines the foundation of authorized access control and data privacy protection. Thus, it is extremely urgent to identify the malicious user or even prove it in a court of justice. In attribute-based access control system, the secret key of user is associated with a set of attributes rather than individual’s identity. As the search and THE EFFICIENT AUTHORIZATION SEARCH SYSTEM FOR SECURE CLOUD STORAGE Dept of BCA BIHE, Davangere. decryption authority can be shared by a set of users who own the same set of attributes, it is hard to trace the original key owner. Providing traceability to a fine-grained search authorization system is critical and not considered in previous searchable encryption systems. More importantly, in the original definition of PEKS scheme, key generation centre (KGC) generates all the secret keys in the system, which inevitably leads to the key escrow problem. That is, the KGC knows all the secret keys of the users and thus can unscrupulously search and decrypt on all encrypted files, which is a significant threat to data security and privacy. Besides, the key escrow problem brings another problem when traceability ability is realized in PEKS. If a secret key is found to be sold and the identity of secret key’s owner (i.e., the traitor) is identified, the strait or may claim that the secret key is leaked by KGC. There is no technical method to distinguish who is the true traitor if the key escrow problem is not solve.

1.2 OBJECTIVES

We include functions like:

- Security and privacy concerns
- Key generation centre (KGC) generates all the secret keys in the system, which inevitably leads to the key escrow problem
- Searchable Encryption
- ABE (Attribute-based encryption)
- Traitor Tracing

2.1 EXISTING SYSTEM

Secure search over encrypted report data is crucial in cloud computing to guarantee the data privacy and usability. To prevent unauthorised data usage, fine-grained access control is necessary in multi-user system. However, authorized user may intentionally leak the secret key for financial benefit. increasing number of companies and individuals prefer to outsource their data storage to cloud server. Despite the tremendous economic and technical advantages, unpredictable security and privacy concerns become the most prominent problem that hinders the widespread adoption of data storage in public cloud infrastructure.

2.2 Proposed System

EF-TAMKSVOD achieves fine-grained data access authorization and supports multiple keyword subset search. In the encryption phase, a keyword set KW is extracted from the file, and both of KW and the file are encrypted. An access policy is also enforced to define the authorized types of users. In the search phase, the data user specifies a keyword set KW0 and generates a trapdoor TKW0 using his secret key. In the test phase, if the attributes linked with user’s secret key satisfy the file’s access policy and KW0 (embedded in the trapdoor) is a subset of KW (embedded in the ciphertext), the corresponding file is deemed as a match file and returned to the data user. The order of keywords in KW0 can be arbitrarily changed, which does not affect the search result. EF-TAMKS-VOD supports flexible system extension, which accommodates flexible number of attributes. The attributes are not fixed in the system initialization phase and the size of attribute set is not restricted to polynomially bound, so that new attribute can be added to the system at any time. Moreover, the size of public parameter does not grow with the number of attributes. No matter how many attributes are supported in the system, no additional communication nor storage costs is brought to EF-TAMKS-VOD. This feature is desirable for the cloud system for its ever-increasing user volume.

3.1 H/W System Configuration: -

Processor	:	Intel (R) Pentium (R)
Speed	:	1.1 Ghz
RAM	:	2GB
Hard Disk	:	57 GB
Key Board	:	Standard Windows Keyboard
Mouse	:	Two or Three Button Mouse
Monitor	:	SVGA

3.2 S/W System Configuration

❖	Operating System	:	Windows 8/7/95/98/2000/XP
❖	Application Server	:	Tomcat5.0/6.X/8.X
❖	Front End	:	HTML, Java, Jsp
❖	Scripts	:	JavaScript.
❖	Server side Script	:	Java Server Pages.
❖	Database Connectivity	:	Mysql.
❖	Java Version	:	jdk 1.8

4.1 FUNCTIONAL REQUIREMENTS

Functional Requirements:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

4.2 NON-FUNCTIONAL REQUIREMENTS

The major non functional of the system are as follows

Useability

The system is designed with completely automated process hence there is no or less user intervention

Reliability

The system is more reliable because of the qualities that are inherited from the chosen platform java. The code built by using java is more reliable.

Performance

This system is developing in the high level languages and using the advanced front end and back end technologies it will give response to the end user on client system within very less time.

Supportability

The system is designed to be the cross platform supportable. The system is supported on a wise range of hardware and any software platform, which is having JVM , break into the system

Implementation

The system is implemented in web environment using struts framework. The apache tomcat Used as the web server and windows XP professional is used as the platform.

Module Description

1. Key generation centre
2. Cloud server
3. Data owner
4. Data user

1. Key generation centre

KGC is responsible to generate the public parameter for the system and the public/secret key pairs for the users. Once the user's secret key is leaked for profits or other purposes, KGC runs trace algorithm to find the malicious user. After the traitor is traced, KGC sends user revocation request to cloud server to revoke the user's search privilege.

2. Cloud server

Cloud server has tremendous storage space and powerful computing capability, which provides on-demand service to the system. Cloud server is responsible to store the data owner's encrypted files and respond on data user's search query.

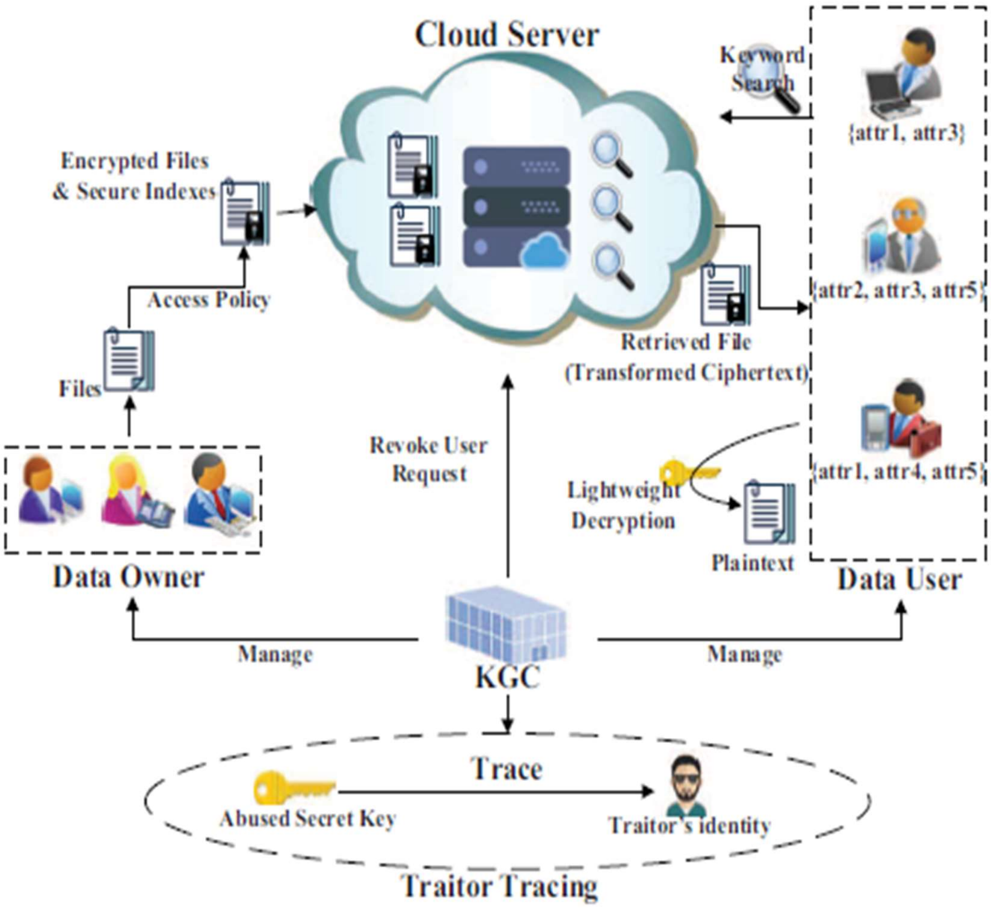
3.Data owner

Data owner utilizes the cloud storage service to store the files. Before the data outsourcing, the data owner extracts keyword set from the file and encrypts it into secure index. The document is also encrypted to ciphertext. During the encryption process, the access policy is specified and embedded into the ciphertext to realize fine-grained access control.

4.Data user

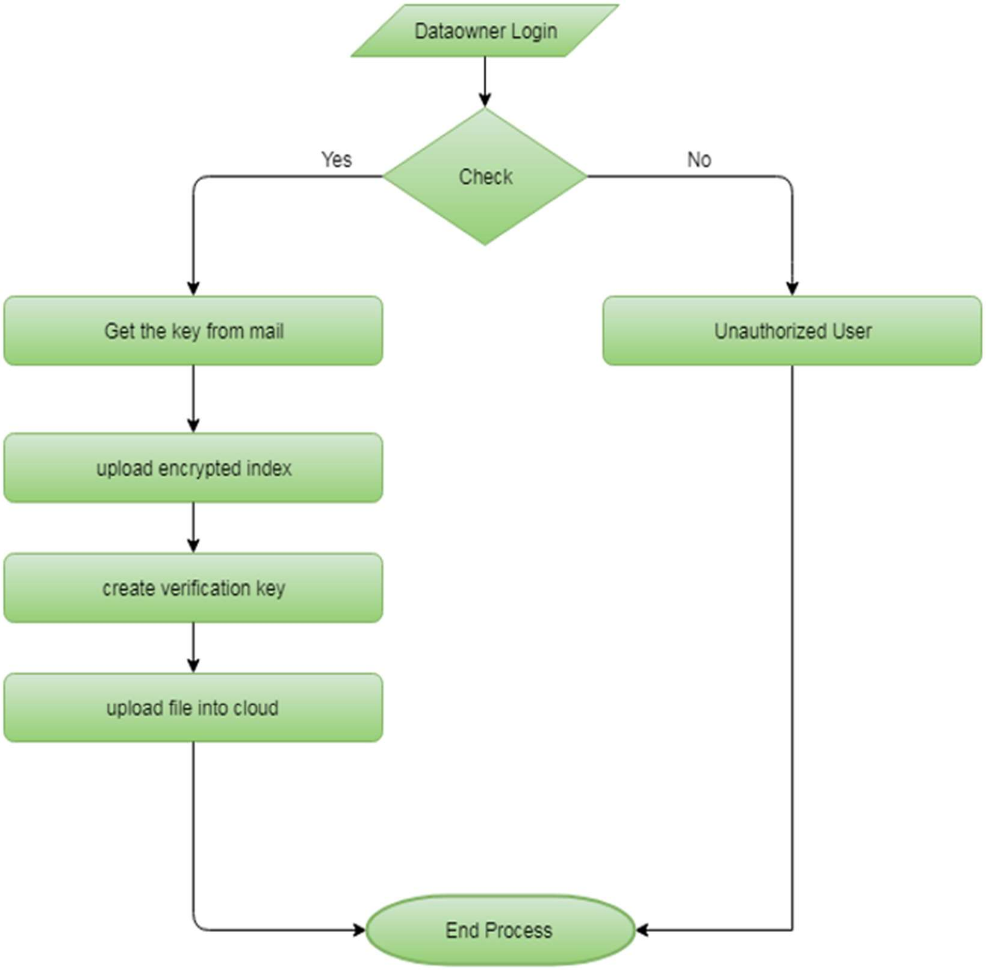
Each data user has attribute set to describe his characteristics, such as professor, computer science college, dean, etc. The attribute set is embedded into user’s secret key. Using the secret key, data user is able to search on the encrypted files stored in the cloud, i.e., chooses a keyword set that he wants to search. Then, the keyword is encrypted to a trapdoor using user’s secret key. If the user’s attribute set satisfies the access policy defined in the encrypted files, the cloud server responds on user’s search query and finds the match files. Otherwise, the search query is rejected. After the match files are returned, the user runs decryption algorithm to recover the plaintext.

Architecture

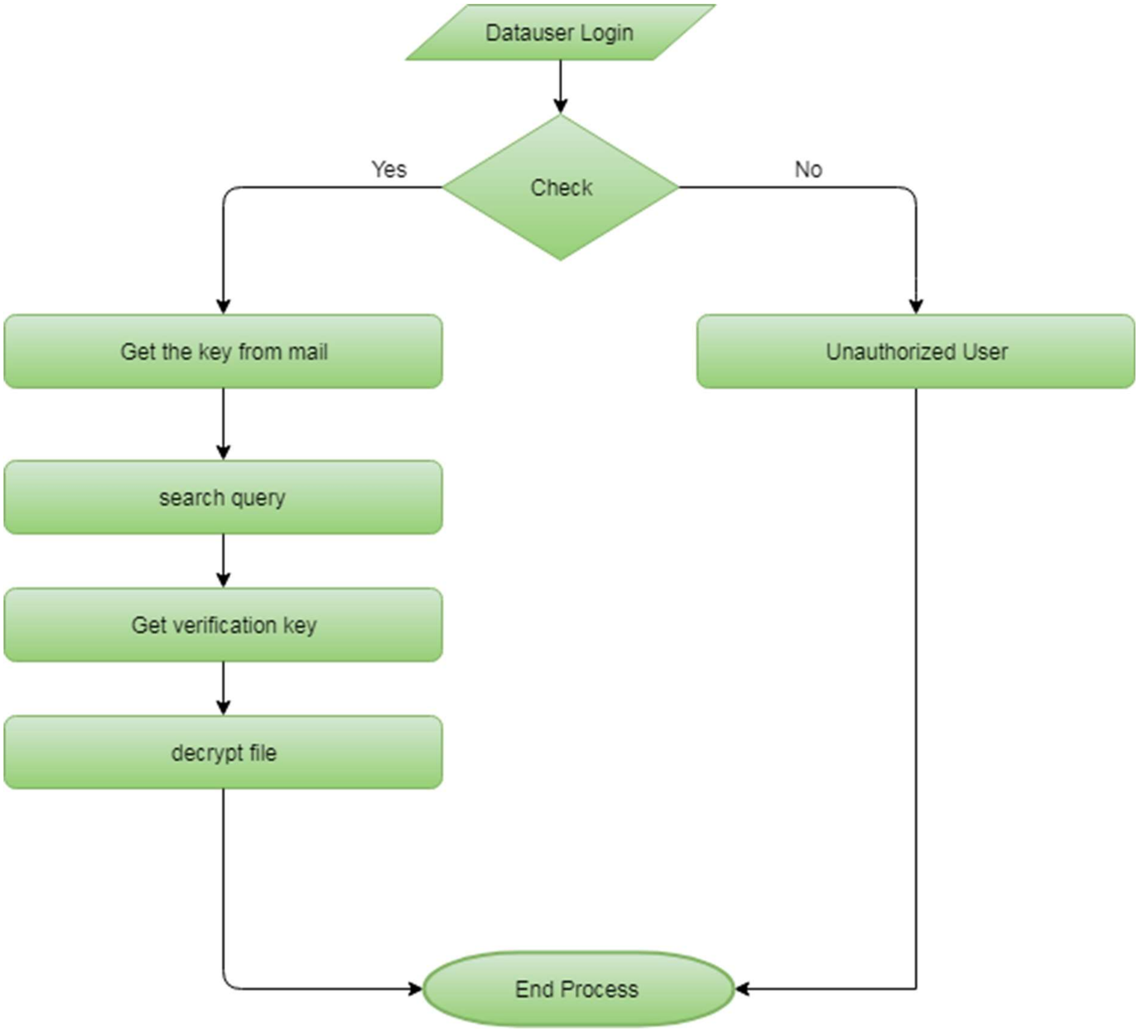


Data Flow Diagram

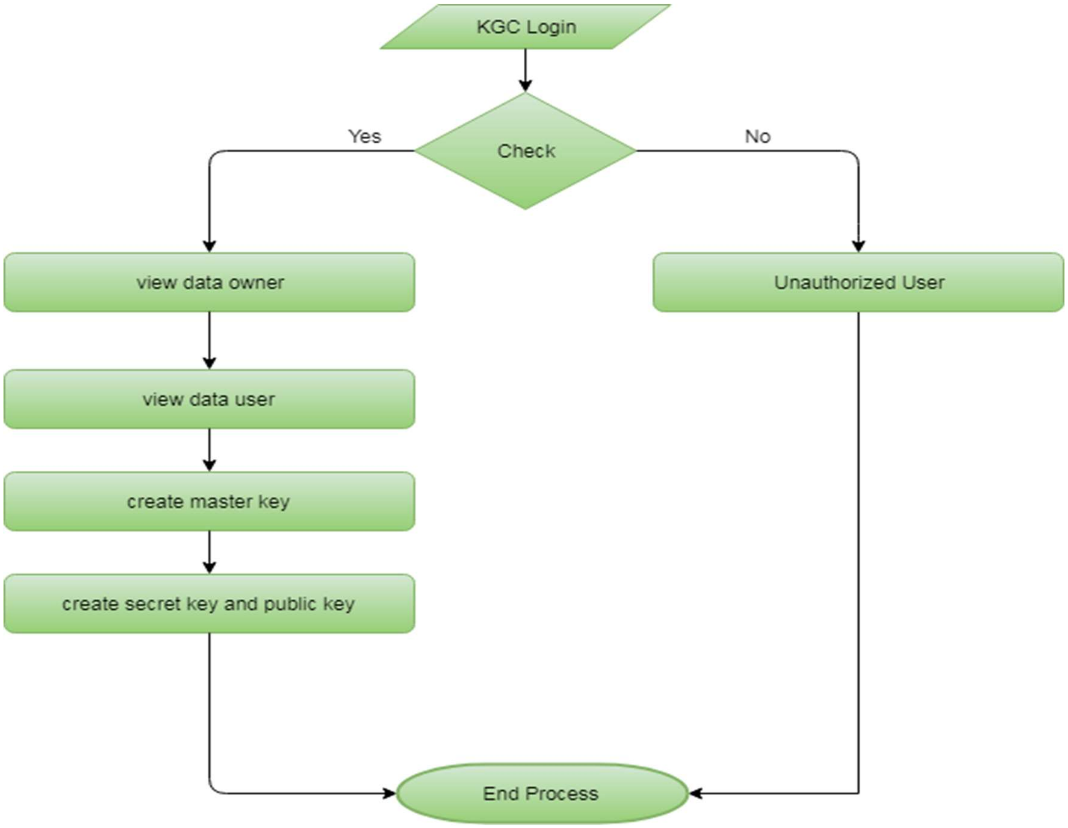
Data owner



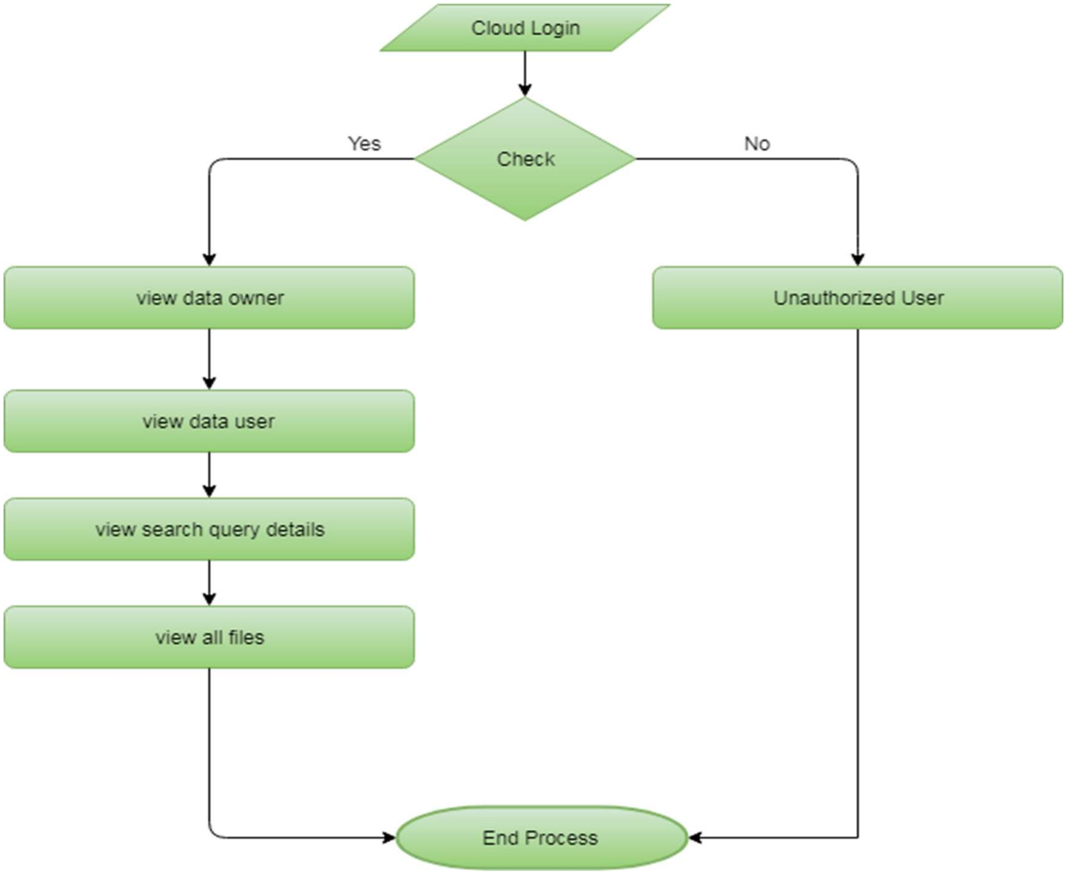
Data User


Datauser Login
Check
Yes
No
Get the key from mail
Unauthorized User
search query
Get verification key
decrypt file
End Process

KEY GENERATION CENTER



Cloud


Cloud Login
Check
Yes
No
view data owner
view data user
view search query details
view all files
Unauthorized User
End Process

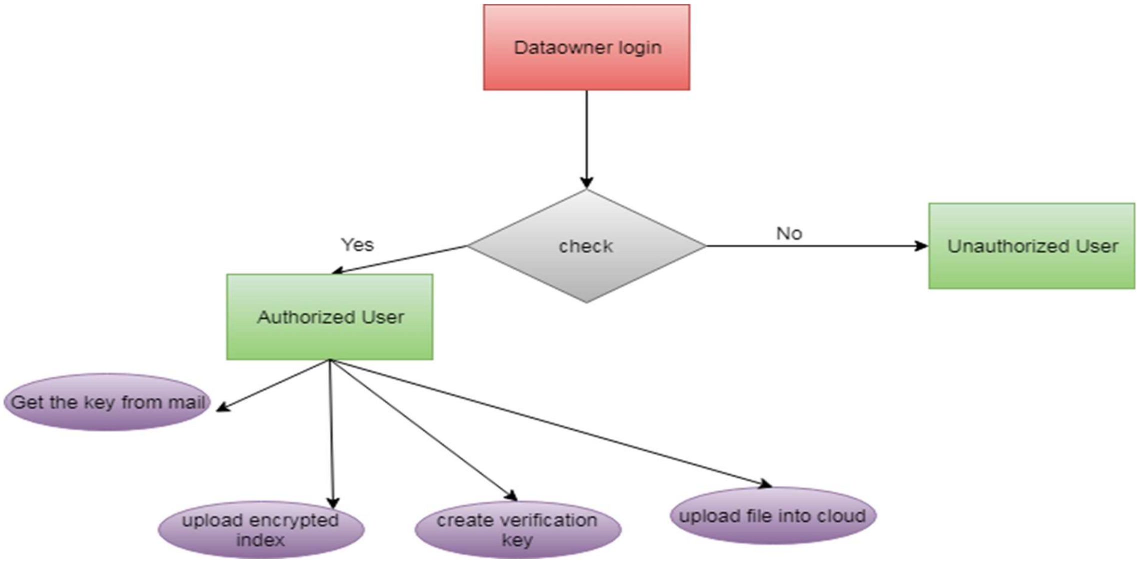
5.3 DATABASE TABLE

Id	Username	Password	Email	Dob	Place	Country	Access_type
1	bhuvana	bhuvana1	C90@gmail.com	2019-01-18	Banglore	India	Srm hospital
2	karan	Karan1	karan@gmail.com	2019-012-2019	Mysore	India	Professor
3	pavan	Pavan1	pavan@gmail.com	2019-03-21	Dharwad	India	Doctor
4	shankri	Shankri1	shankri@gmail.com	2019-04-17	hubli	India	Teacher
5	kumar	Kumar1	kumar@gmail.com	2019-05--18	bellary	India	

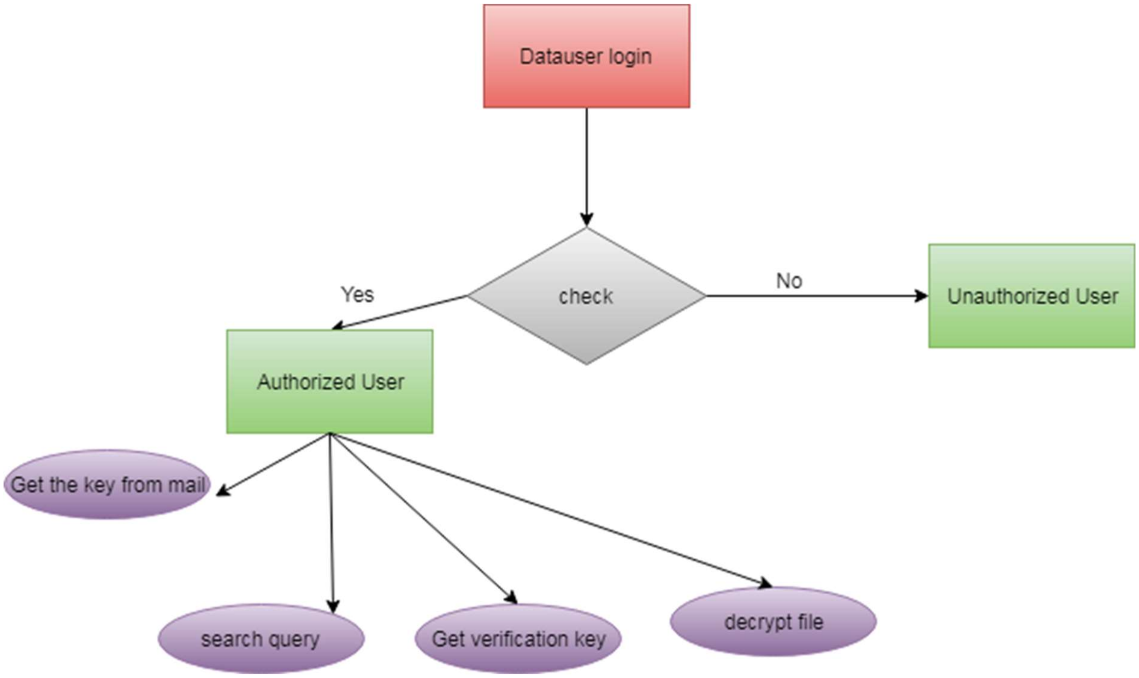
Public_key	Attribute
C44Uis	49553
S65Id	65368
H69gS	76389
F40dE	96392
C39sJ	53783
D25iN 32678	

5.4 ER Diagram

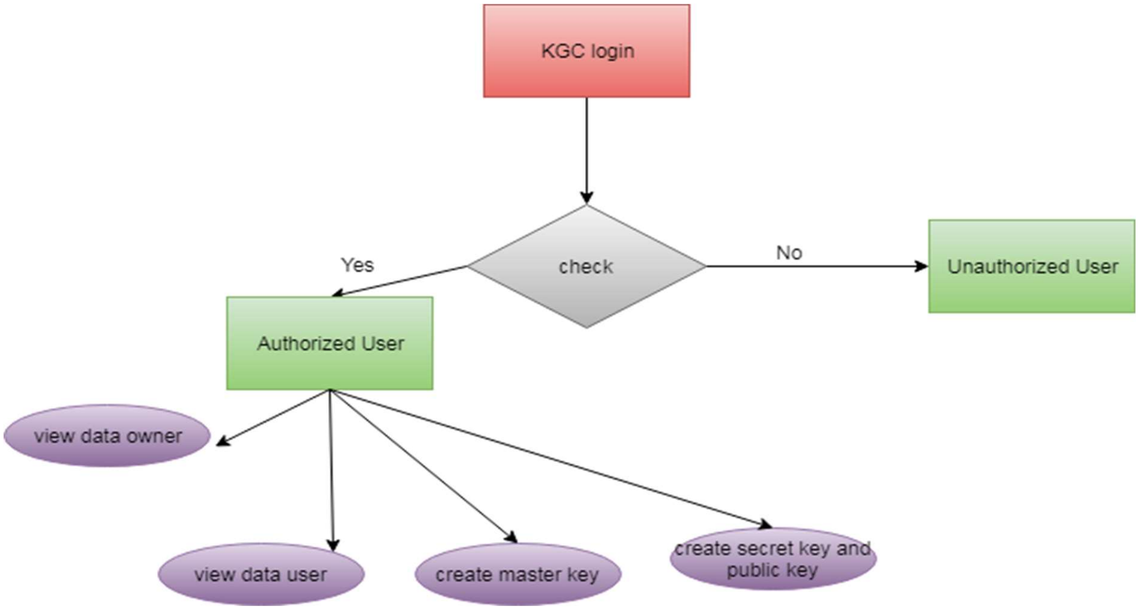
Data owner



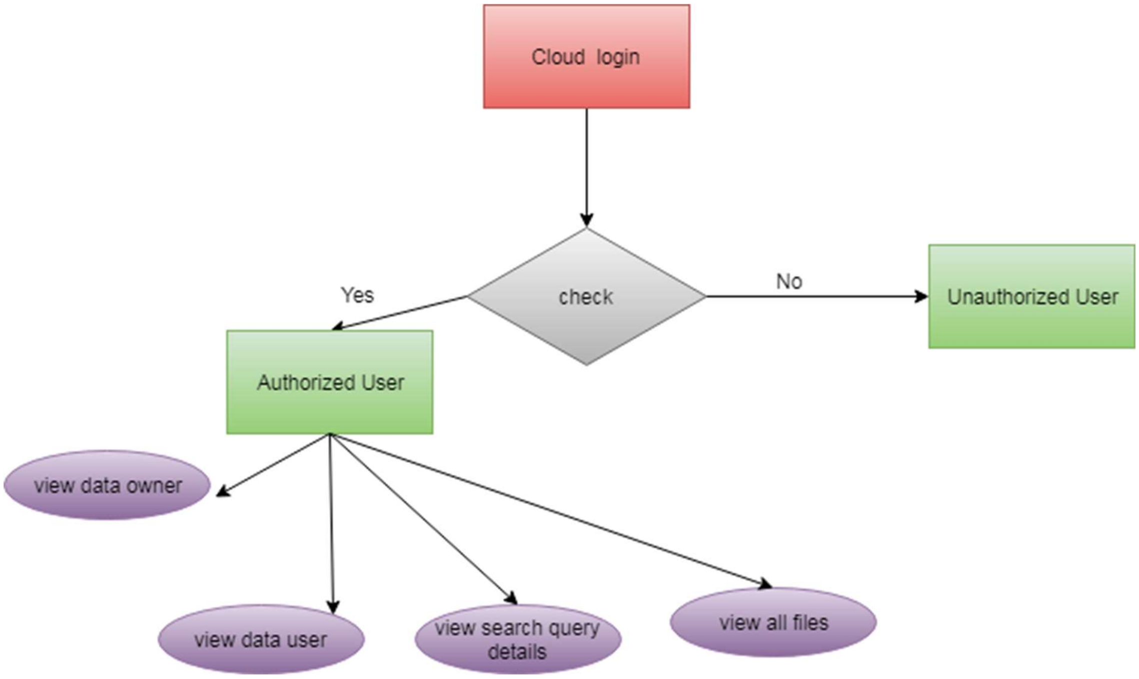
Data User



KGC (Key Generation Centre)

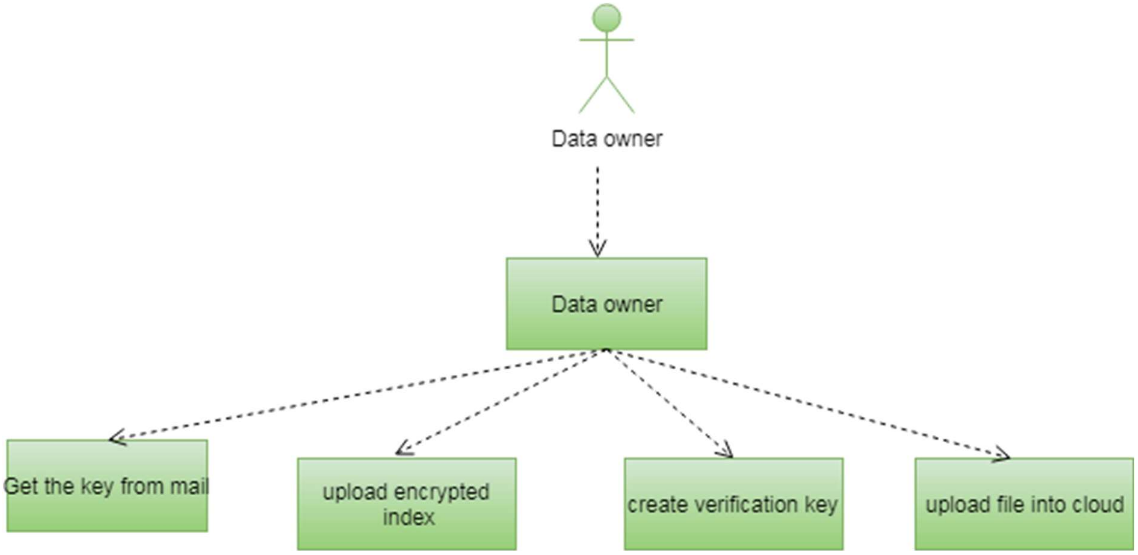


Cloud

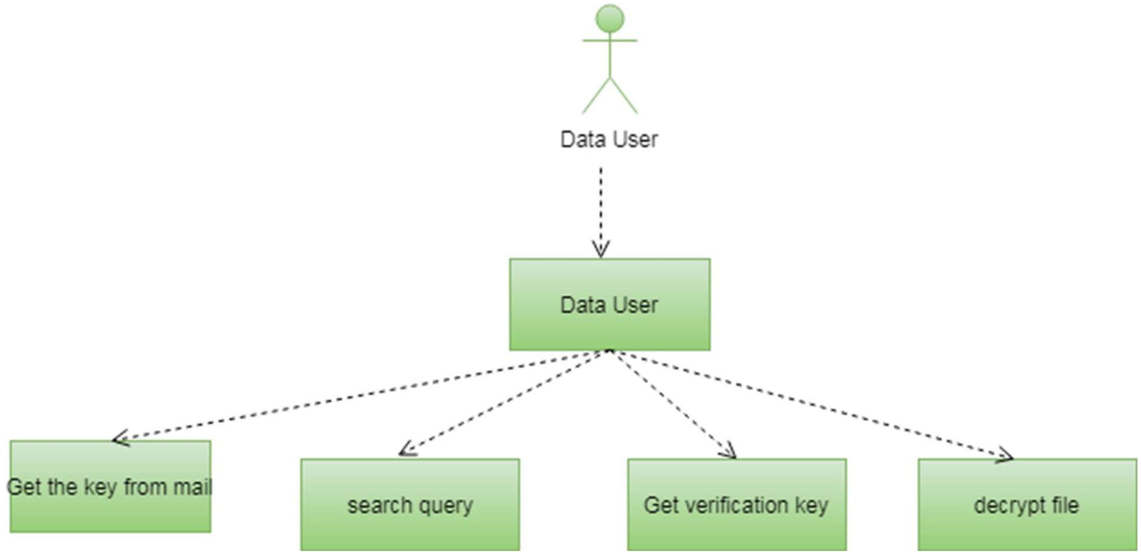


5.5 Use case Diagram

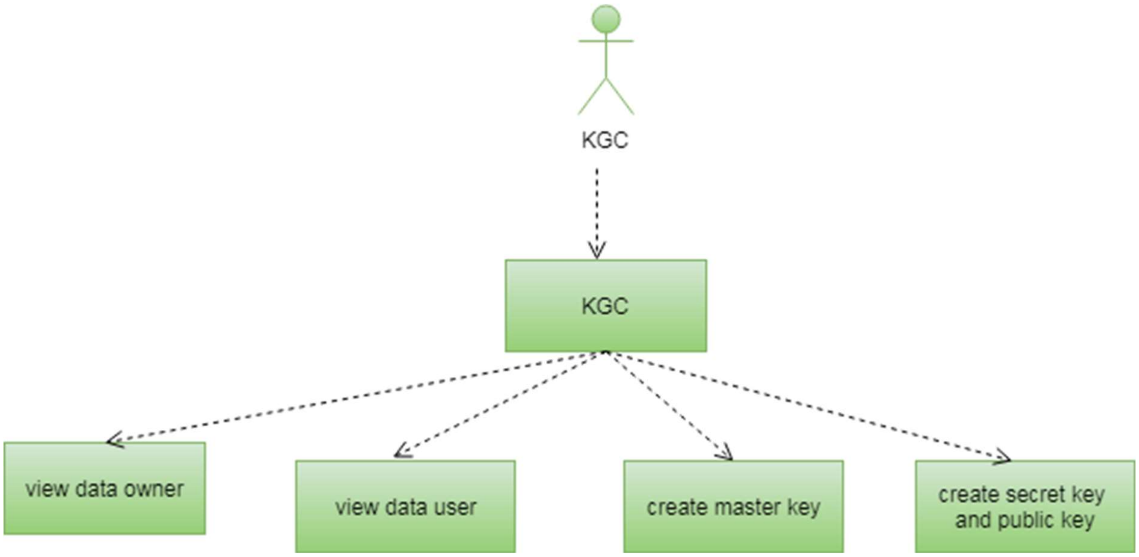
Data owner



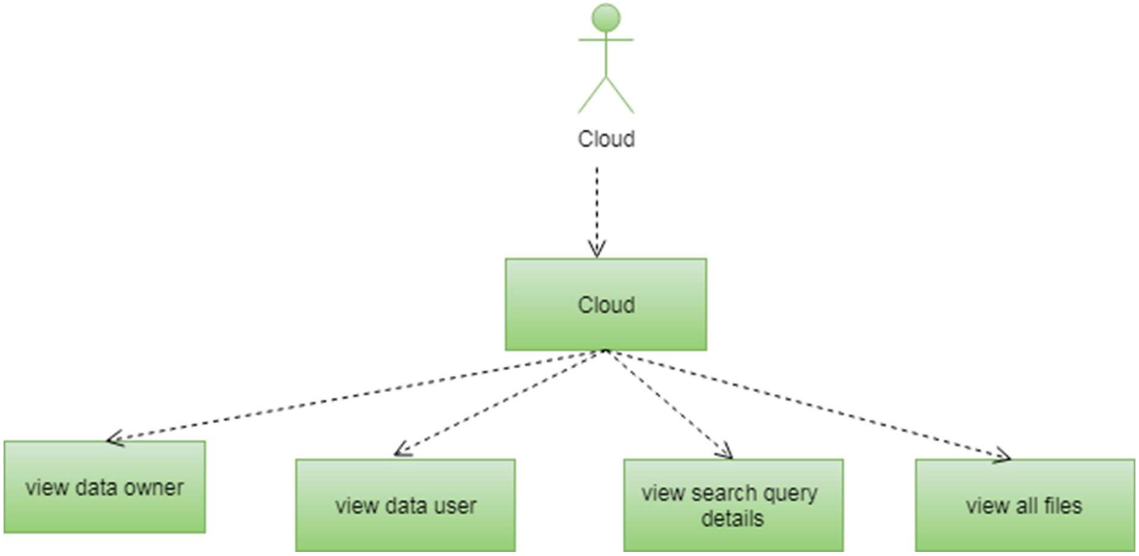
Data User



KGC (Key Generation Centre)


KGC
KGC
view data owner
view data user
create master key
create secret key and public key

Cloud


Cloud
Cloud
view data owner
view data user
view search query details
view all files

6.1 Code Used

```
<!--A Design by W3layouts
Author: W3layout
Author URL: http://w3layouts.com
License: Creative Commons Attribution 3.0 Unported
License URL: http://creativecommons.org/licenses/by/3.0/
-->
<!DOCTYPE html>
<html lang="en">

<head>
<title>Efficient Traceable </title>
<!-- Meta tags -->
<meta name="viewport" content="width=device-width, initial-scale=1" />
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta name="keywords" content="Heaven Booking Form Responsive Widget, Audio
and Video players, Login Form Web Template, Flat Pricing Tables, Flat Drop-
Downs, Sign-Up Web Templates, Flat Web Templates, Login Sign-up Responsive
Web Template, Smartphone Compatible Web Template, Free Web Designs for
Nokia, Samsung, LG, Sony Ericsson, Motorola Web Design"
/>
<script type="application/x-javascript">
addEventListener("load", function() { setTimeout(hideURLbar, 0); }, false);
function hideURLbar(){ window.scrollTo(0,1); }
</script>
<!-- Meta tags -->
<!--stylesheets-->
<link href="css/style.css" rel='stylesheet' type='text/css' media="all">
<!--//style sheet end here-->

<link href="//fonts.googleapis.com/css?family=Barlow:300,400,500,600"
rel="stylesheet">
</head>

<body>
```

```
<h2 class="header-w3ls">
Efficient Traceable Authorization Search System for Secure Cloud
Storage</h2>
<div class="left_column">
<a href="index.html">DATA OWNER</a>
<a href="user.jsp">DATA USER</a>
<a href="kgc.jsp">KGC</a>
<a href="cloud.jsp">CLOUD</a>
</div>
<!--login form-->
<div class="hotel-bothside">
<h2>DATA OWNER REGISTER</h2>
<form action="owner_registerdb.jsp">
<div class="mid-cls">

<div class="hotel-left-w3ls">
<!-- <h2>Personal Details</h2>

<div class="main">
<div class="form-left-to-w3l">

<input type="text" name="name" placeholder="Name" required="">
<div class="clear"></div>
</div>
<div class="form-right-to-w3ls">

<input type="text" name="last name" placeholder="Last Name" required="">
<div class="clear"></div>
</div>

</div>
<div class="main">

<div class="form-left-to-w3l">

<input type="email" name="email" required="" placeholder="Email">
</div>
<div class="form-right-to-w3ls ">


<input type="text" name="phone number" placeholder="Phone Number"
required="">
</div>
</div>
<div class="form-add-to-w3ls add">

<input type="text" name="address" placeholder="Street Address" required="">
<div class="clear"></div>
</div>


<div class="main">
<div class="form-left-to-w3l">

<input type="text" name="city" placeholder="City" required="">
<div class="clear"></div>
</div>
<div class="form-right-to-w3ls">
<input type="text" name="state" placeholder="State" required="">
```

```
<div class="clear"></div>
</div>
</div>
<div class="main">
<div class="form-left-to-w3l">

<input class="pin-bottom" type="text" name="Pin code" placeholder="Pin
code" required="">
<div class="clear"></div>
</div>
<div class="form-right-to-w3ls">

<select class="form-control country-buttom">
<option value="">
Select Country</option>
<option value="category2">Oman</option>
<option value="category1">Australia</option>
<option value="category3">America</option>
<option value="category3">London</option>
<option value="category3">Goa</option>
<option value="category3">Canada</option>
<option value="category3">Srilanka</option>
</select>
</div>
</div>

<div class="clear"></div> -->
<img src="images/register2.png" width="300" height="300" style="margin-
left:40px;margin-top:100px;">
</div>
<div class="hotel-right-w3ls" style="margin-top:50px;">


<!--  <div class="main">
<div class="form-left-to-w3l">
<input type="date" name="dateofbirth" id="dateofbirth"
placeholder="Arrival-Date" required="">

</div>
<div class="form-right-to-w3ls">
<input type="date" name="dateofbirth" id="dateofbirth"
placeholder="Departure-Date" required="">

</div>
</div> -->
<!--  <div class="main">
<div class="form-right-to-w3ls">

<select class="form-control buttom">
<option value="">
No Of Adult</option>
<option value="category2">2</option>
<option value="category1">3</option>

</select>
</div>
<div class="form-right-to-w3ls">

<select class="form-control buttom">
```

```
<option value="">
No Of Kids</option>
<option value="category2">0</option>
<option value="category1">1</option>
<option value="category3">2</option>

</select>
</div>
</div> -->
<!--  <div class="main">
<div class="form-right-to-w3ls">

<select class="form-control buttom">
<option value="">
No Of Night At Hotel</option>
<option value="category2">2 Nights</option>
<option value="category1">3 Nights</option>
<option value="category1">And More</option>
</select>
</div>
<div class="form-right-to-w3ls">

<select class="form-control buttom">
<option value="">
Room Preference</option>
<option value="category2">single</option>
<option value="category1">Double</option>
<option value="category3">Suite</option>

</select>
</div>
</div> -->
<!--  <div class="form-control-w3l">

<textarea name="Message" placeholder="Any Special Request..."
required=""></textarea>
</div> -->
<div class="main">
<div class="form-left-to-w3l">

<input type="text" name="name" placeholder="Username" required=""
style="width:350px;">
<div class="clear"></div>
</div>


</div>
<div class="main">

<div class="form-left-to-w3l">

<input type="password" name="password" required="" placeholder="Password"
style="width:350px;">
</div>

</div>
<div class="main">

<div class="form-left-to-w3l">
```

```
<input type="email" name="email" required="" placeholder="Email"
style="width:350px;">
</div>

</div>
<div class="main">
<div class="form-left-to-w3l">
<input type="date" name="dateofbirth" id="dateofbirth" placeholder="Date of
Birth" required="" style="width:350px;">

</div>
</div>
<div class="main">
<div class="form-left-to-w3l">

<input type="text" name="place" placeholder="Place" required=""
style="width:350px;">
<div class="clear"></div>
</div>


</div>
<div class="main">
<div class="form-left-to-w3l">

<input type="text" name="country" placeholder="Country" required=""
style="width:350px;">
<div class="clear"></div>
</div>


</div>
<div class="clear"></div>
<div class="btnn" style="margin-left:-70px;">
<button type="submit" style="width:350px;">Register</button><br>
</div>

</div>
</div>
</form>
</div>


</body>

</html>
```

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.1 TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle,

although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

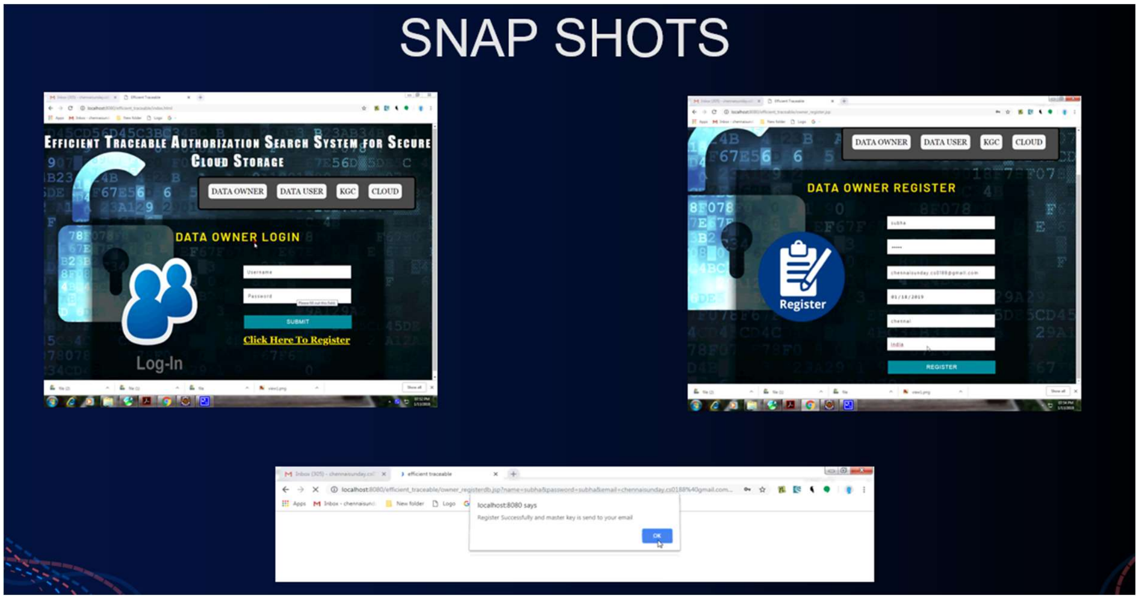
The task of the integration test is to check that components or software applications, e.g., components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.



SNAP SHOTS

subha

*****

chennaisunday.cs0188@gmail.com

01/18/2019

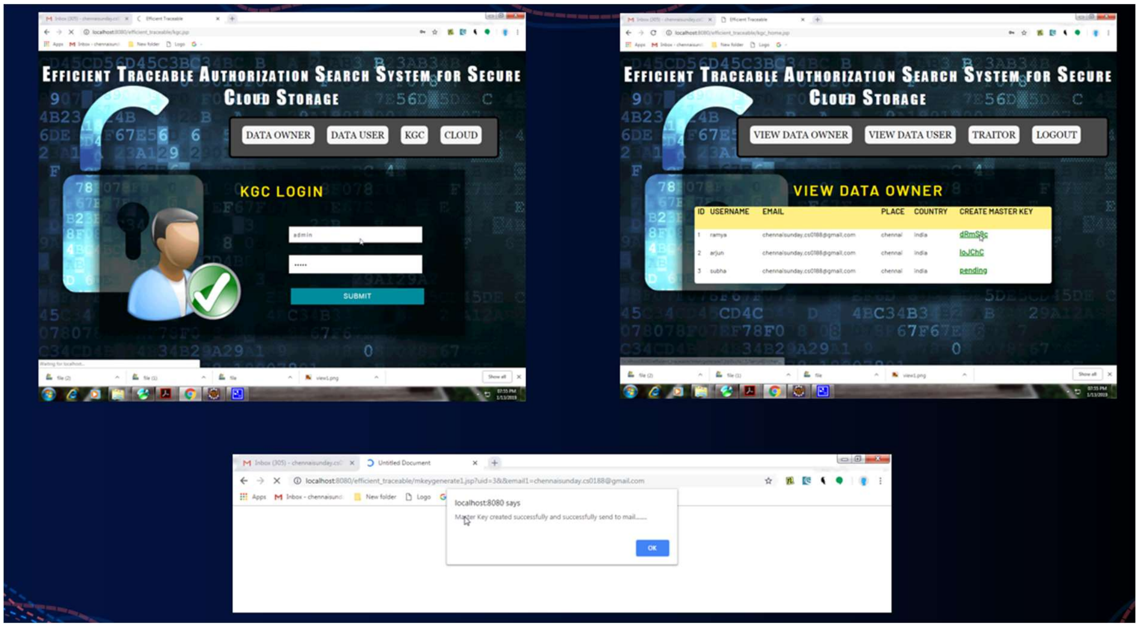
chennai

india

localhost:8080/efficient_traceable/owner_registerdb.jsp?name=subha&password=subha&email=chennaisunday.cs0188%40gmail.com...

localhost:8080 says

Register Successfully and master key is send to your email



admin

*****

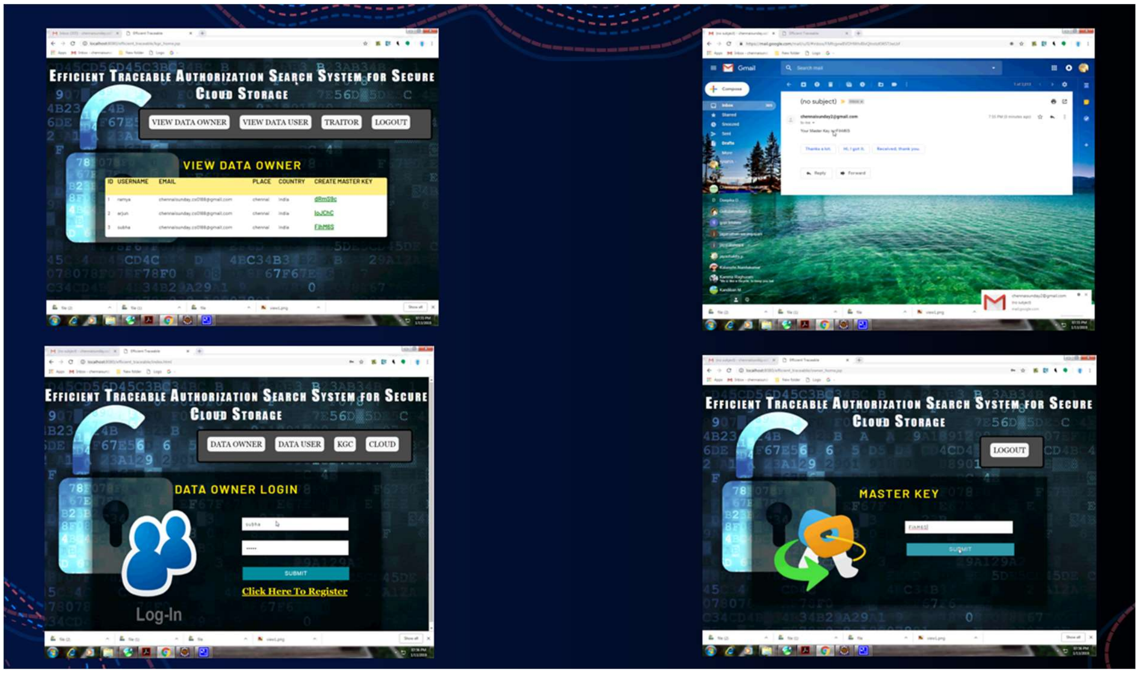
VIEW DATA OWNER

ID	USERNAME	EMAIL	PLACE	COUNTRY	CREATE MASTER KEY
1	ramya	chennaisunday.cs0188@gmail.com	chennai	india	dRmS8c
2	arjun	chennaisunday.cs0188@gmail.com	chennai	india	loJChC
3	subha	chennaisunday.cs0188@gmail.com	chennai	india	pending

localhost:8080/efficient_traceable/mkeygenerate1.jsp?uid=3&&email1=chennaisunday.cs0188@gmail.com

localhost:8080 says

Master Key created successfully and successfully send to mail......



VIEW DATA OWNER

ID	USERNAME	EMAIL	PLACE	COUNTRY	CREATE MASTER KEY
1	ramya	chennaisunday.cs0188@gmail.com	chennai	india	dRmS9c
2	arjun	chennaisunday.cs0188@gmail.com	chennai	india	IoJChC
3	subha	chennaisunday.cs0188@gmail.com	chennai	india	FihM8S

(no subject)

chennaisunday2@gmail.com

Your Master Key is:FihM8S

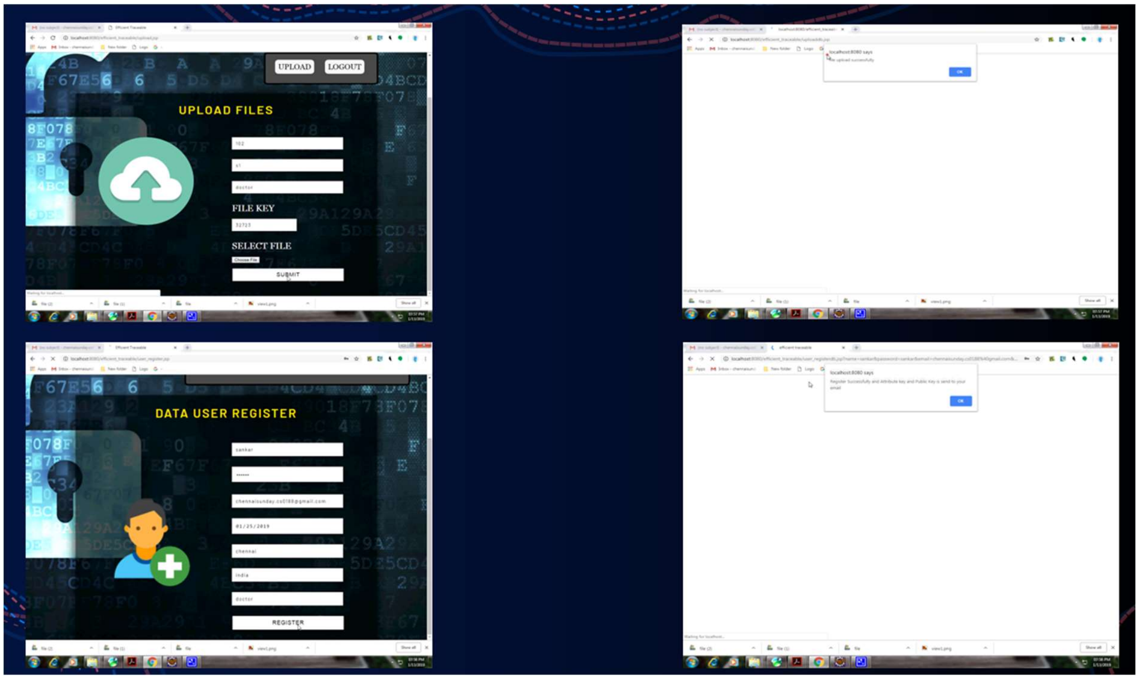
DATA OWNER LOGIN

subha

*****

MASTER KEY

FihM8S



UPLOAD FILES

102

s1

doctor

FILE KEY

32723

localhost:8080 says

file upload successfully

DATA USER REGISTER

sankar

******

chennaisunday.cs0188@gmail.com

01/25/2019

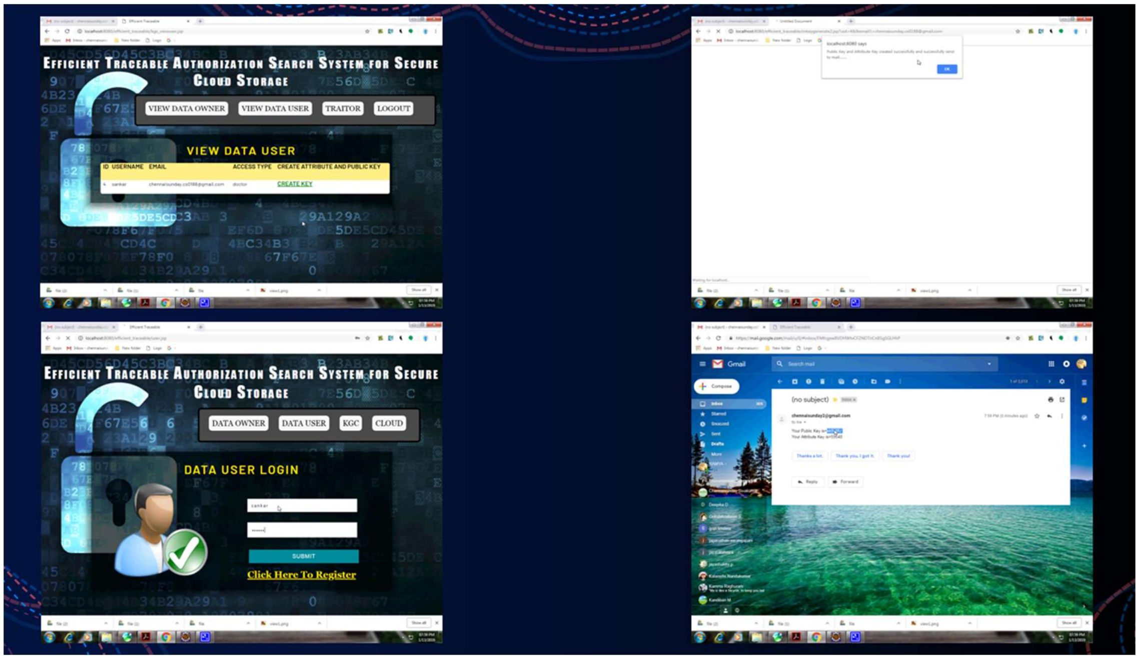
chennai

india

doctor

localhost:8080 says

Register Successfully and Attribute key and Public Key is send to your email



ID	USERNAME	EMAIL	ACCESS TYPE	CREATE ATTRIBUTE AND PUBLIC KEY
4	sankar	chennaisunday.cs0188@gmail.com	doctor	CREATE KEY

localhost:8080 says

Public Key and Attribute Key created successfully and successfully send to mail......

(no subject)

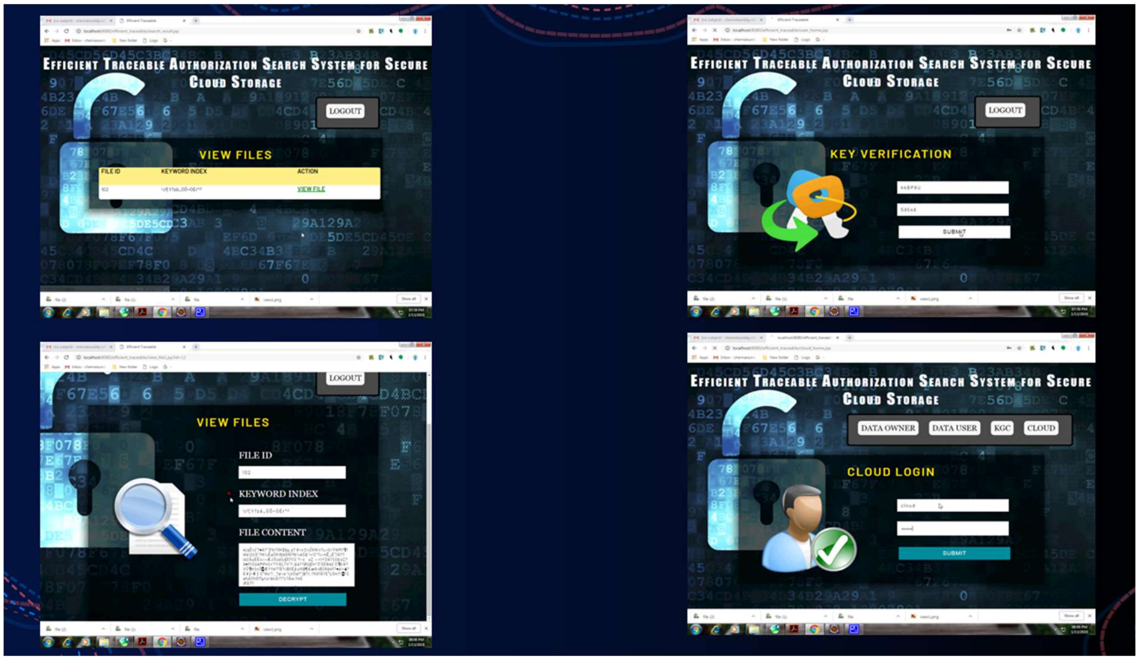
chennaisunday2@gmail.com

to me

Your Public Key is=[illegible]

Your Attribute Key is=59548

sankar



FILE ID	KEYWORD INDEX	ACTION
102	½¶ÝY?źA,ÒÖ-ÒÉr*²	VIEW FILE

KEY VERIFICATION

AABP9U

59548

VIEW FILES

FILE ID

102

KEYWORD INDEX

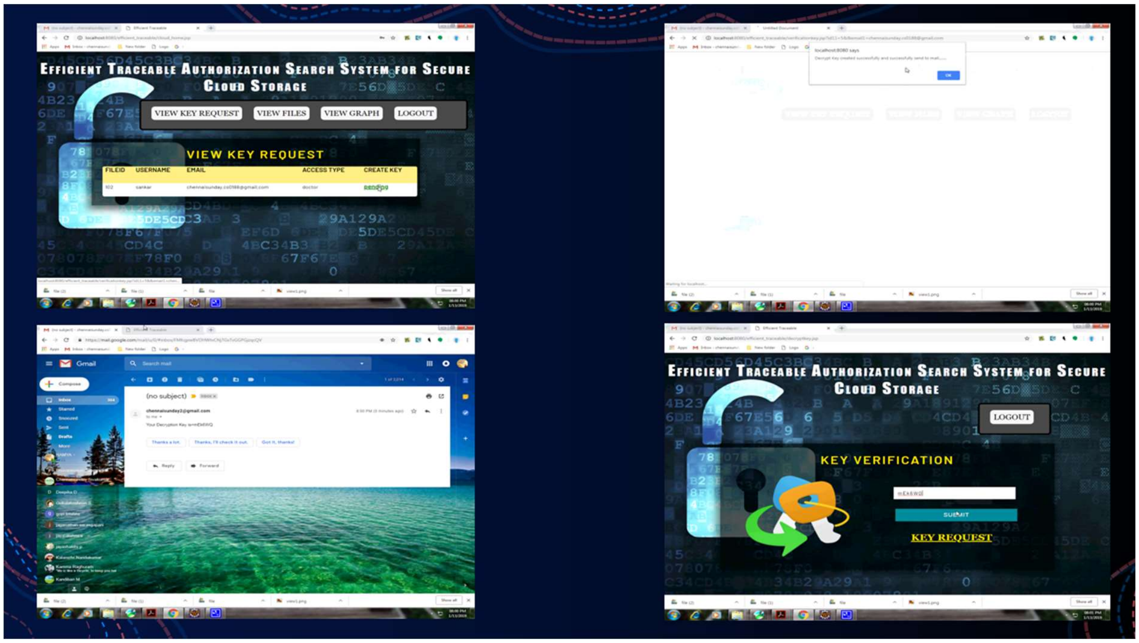
½¶ÝY?źA,ÒÖ-ÒÉr*²

FILE CONTENT

[illegible]

CLOUD LOGIN

cloud



Efficient Traceable Authorization Search System for Secure Cloud Storage

VIEW KEY REQUEST

FILEID	USERNAME	EMAIL	ACCESS TYPE	CREATE KEY
102	sankar	chennaisunday.cs0188@gmail.com	doctor	pending

localhost:8080 says

Decrypt Key created successfully and successfully send to mail......

(no subject)

chennaisunday2@gmail.com

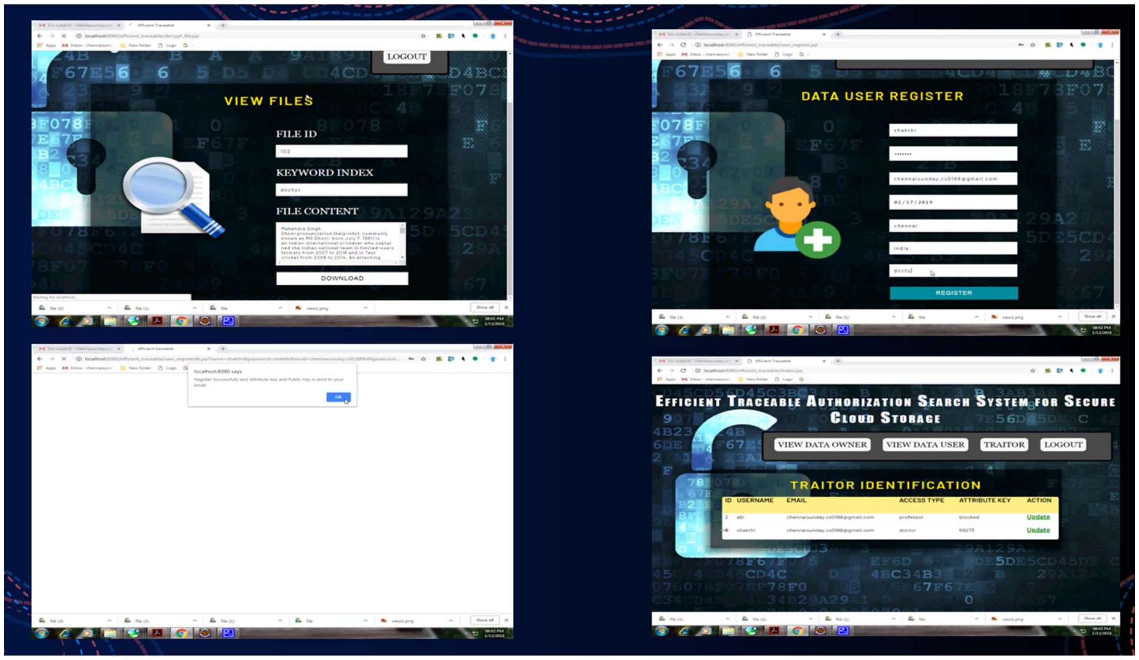
to me

Your Decryption Key is=mEkEWQ

Efficient Traceable Authorization Search System for Secure Cloud Storage

KEY VERIFICATION

mEkEWQ



VIEW FILES

FILE ID

102

KEYWORD INDEX

doctor

FILE CONTENT

Mahendra Singh Dhoni pronunciation (help·info); commonly known as MS Dhoni; born July 7, 1981) is an Indian international cricketer who captained the Indian national team in limited-overs formats from 2007 to 2016 and in Test cricket from 2008 to 2014. An attacking

DATA USER REGISTER

shakthi

•••••••

chennaisunday.cs0188@gmail.com

01/17/2019

chennai

india

docto

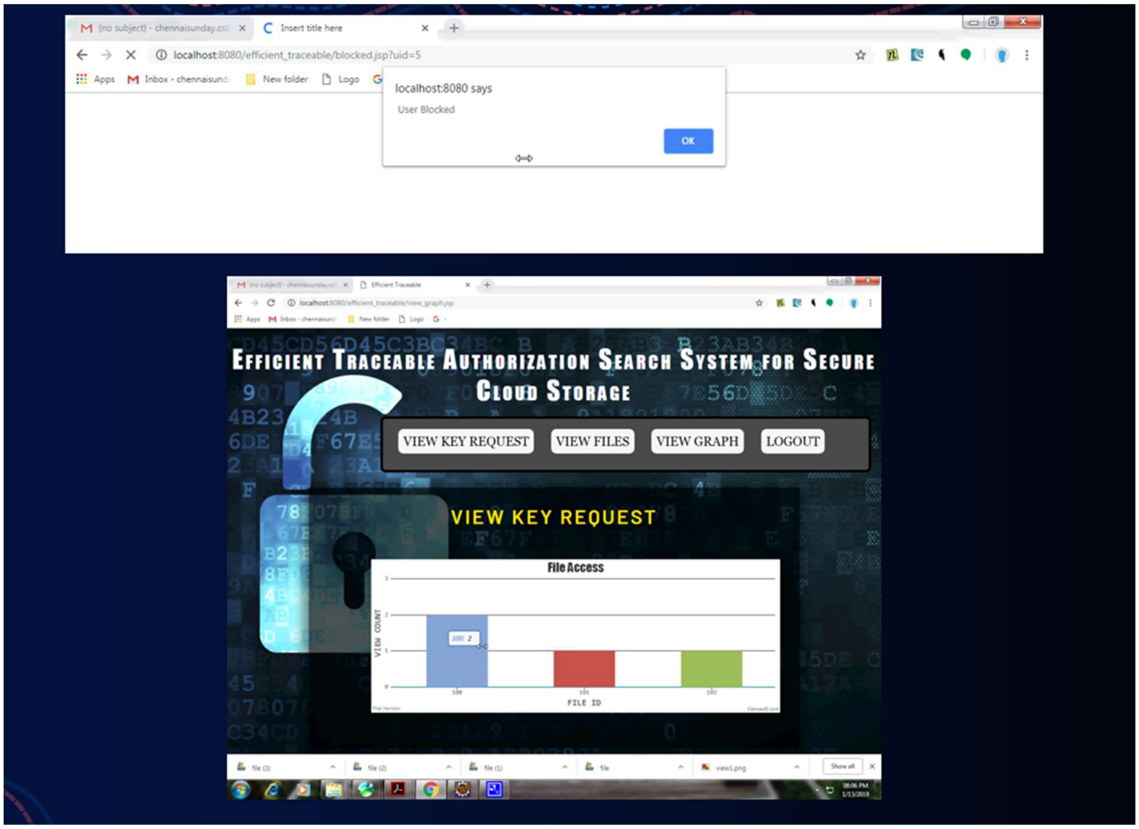
localhost:8080 says

Register Successfully and Attribute key and Public Key is send to your email

Efficient Traceable Authorization Search System for Secure Cloud Storage

TRAITOR IDENTIFICATION

ID	USERNAME	EMAIL	ACCESS TYPE	ATTRIBUTE KEY	ACTION
2	abi	chennaisunday.cs0188@gmail.com	professor	blocked	Update
6	shakthi	chennaisunday.cs0188@gmail.com	doctor	69273	Update



localhost:8080/efficient_traceable/blocked.jsp?uid=5

localhost:8080 says

User Blocked

OK

localhost:8080/efficient_traceable/view_graph.jsp

EFFICIENT TRACEABLE AUTHORIZATION SEARCH SYSTEM FOR SECURE CLOUD STORAGE

VIEW KEY REQUEST

The enforcement of access control and the support of keyword search are important issues in secure cloud storage system. In this work, we defined a new paradigm of searchable encryption system, and proposed a concrete construction. It supports flexible multiple keywords subset search, and solves the key escrow problem during the key generation procedure. Malicious user who sells secret key for benefit can be traced. The decryption operation is partly outsourced to cloud server and the correctness of half-decrypted result can be verified by data user. The performance analysis and simulation show its efficiency in computation and storage overhead. Experimental results indicate that the computation overhead at user's terminal is significantly reduced, which greatly saves the energy for resource-constrained devices of users.

BIBILOGRAPHY

[1]. Q. Chai and G. Gong, “Verifiable symmetric searchable encryption for semi honest- bu curious cloud servers,” in Communications (ICC), 2012 IEEE International Conference on. IEEE, 2012, pp. 917– 922. [2]. S. Hou, T. Uehara, S. Yiu, L. C. Hui, and K. Chow, “Privacy preserving multiple keyword search for confidential investigation of remote forensics,” in Multimedia Information Networking and Security (MINES), 2011 Third International Conference on. IEEE, 2011, pp. 595–599. [3]. N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, “Privacy-preserving multikey word ranked search over encrypted cloud data,” Parallel and Distributed Systems, IEEE Transactions on, vol. 25, no. 1, pp. 222–233, 2014. [4]. B. Wang, S. Yu, W. Lou, and Y. T. Hou, “Privacy-preserving multi-keyword fuzzy search over encrypted data in the cloud,” in INFOCOM, 2014 Proceedings IEEE. [5]. L. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, “A break in the clouds: towards a cloud definition,” ACM SIGCOMM Computer Communication Review, vol. 39, no. 1, pp. 50–55,2008.