International Journal of Advanced Multidisciplinary Research (IJAMR)
ISSN: 2393-8870
www.ijarm.com

Research Article
Cloud storage for Local Hardware and Software management

P.Vignesh*

Department of Computer Applications, Sri Venkateswara Engineering College, Kanchipuram, Tamilnadu, India
*Corresponding Author

Keywords

Cloud storage,
Of local hardware and software management
TPA auditing,

Abstract

Cloud storage enables users to remotely store their data and enjoy the on-demand high quality cloud applications without the burden of local hardware and software management. Though the benefits are clear, such a service is also relinquishing users' physical possession of their outsourced data, which inevitably poses new security risks toward the correctness of the data in cloud. In order to address this new problem and further achieve a secure and dependable cloud storage service,we could not recover the data from the loss that's what in our proposed system introduce some more algorithm like segmentation, encoding and encryption along with TPA auditing using all the above algorithm we can achieve the data confident because we just increases the extraction time when the extraction time is increase automatically the security level also increase.we propose a flexible distributed storage integrity auditing mechanism. The proposed design allows users to audit the cloud storage with very lightweight communication and computation cost. The auditing result not only ensures strong cloud storage correctness guarantee, but also simultaneously achieves fast data error localization, i.e., the identification of misbehaving server.

Introduction

cloud computing is the delivery of computing and storage capacity as a service to a heterogeneous community of end-recipients. The name comes from the use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagram. Cloud computing entrusts services with a user's data, software and computation over a network.

There are three types of cloud computing:

- Infrastructure as a Service (IaaS)
- Platform as a Service (PaaS)
- Software as a Service (SaaS)

Using Infrastructure as a Service, users rent use of servers provided by one or more cloud providers. Using Platform as a Service, users rent use of servers and the system software to use in them. Using Software as a Service, users also rent application software and databases. The cloud providers manage the infrastructure and platforms on which the applications run.

End users access cloud-based applications through a web browser or a light-weight desktop or mobile app while the business software and user's data are stored on servers at a remote location. Proponents claim that cloud computing allows enterprises to get their applications up and running faster, with improved manageability and less maintenance, and enables IT to more rapidly adjust resources to meet fluctuating and unpredictable business demand.

Cloud Computing Security

The cloud infrastructures are much more powerful and reliable than personal computing devices, broad range of both internal and external threats for data integrity still exist. The users may not retain a local copy of outsourced data; there exist various incentives for CSP to behave unfaithfully toward the cloud users regarding the status of their outsourced data. For example, to increase the profit margin by reducing cost, it is possible for CSP to discard rarely accessed data without being detected in a timely fashion. Similarly, CSP may even attempt to hide data loss incidents so as to maintain a reputation. Therefore, although outsourcing data into the cloud is economically attractive for the cost and complexity of long-term large-scale data storage, its lacking of offering strong assurance of data integrity and availability may impede its wide adoption by both enterprise and individual cloud users.In order to achieve the assurances of cloud data integrity and availability and enforce the quality of cloud storage service, efficient methods that enable on-demand data correctness verification on behalf of cloud users. Moving data into the cloud offers great convenience to users since they don't have to care about the complexities of direct hardware management. The pioneer of cloud computing vendors, Amazon Simple Storage Service (S3), and Amazon Elastic Compute Cloud (EC2) are both well-known examples. While these internet-based online services do provide huge amounts of storage space and customizable computing resources, this computing platform shift, however, is eliminating the responsibility of local machines for data maintenance at the same time. As a result, users are at the mercy of their cloud service providers (CSP) for the availability and integrity of their data.

OBJECTIVE:

The main objective users can remotely store their data and enjoy the on-demand high-quality applications and services from a shared pool of configurable computing resources, without the burden of local data storage and maintenance We motivate the public auditing system of data storage security in cloud computing and provide a privacy-preserving auditing protocol. Our scheme enables an external auditor to audit user's cloud data without learning the data content. To the best of our knowledge, our scheme is the first to support scalable and efficient privacy-preserving public storage auditing in cloud. Specifically, our scheme achieves batch auditing where multiple delegated auditing tasks from different users can be performed simultaneously by the TPA in a privacy-preserving manner. We prove the security and justify the performance of our proposed schemes through concrete experiments and comparisons with the state of the art.

SYSTEM SPECIFICATIONS:

2.1 HARDWARE SPECIFICATION

Processor	:	Intel Pentium IV
Clock speed	:	1.8 GHz
RAM	:	256 MB
HDD	:	80 GB
Pointing device	:	Scroll Mouse
Keyboard	:	101 Standard Key-board
Peripherals	:	Printer

2.2 SOFTWARE SPECIFICATION

Core Language	:	Advanced Java (Servlet)
Operating System	:	Windows 7
Database	:	Mysql 5.5
Front End	:	JSP and Html
IDE	:	Spring Source 3.5
Application Engine	:	Amazon EC2

2.3 SOFTWARE DESCRIPTION:

ABOUT JAVA

Java is a computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java applications are typically compiled to bytecode (class file) that can run on any Java virtual machine (JVM) regardless of computer architecture. Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems(which has since merged into Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

The original and reference implementation Java compilers, virtual machines, and class libraries were developed by Sun from 1991 and first released in 1995. In compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java (bytecode compiler), GNU Class path (standard libraries), and Iced Tea-Web (browser plugin for applets). One characteristic of Java is portability, which means that computer programs written in the Java language must run similarly on any hardware/operating-system platform. This is achieved by compiling the Java language code to an intermediate representation called Java bytecode, instead of directly to platform-specific machine code. Java bytecode instructions are analogous to machine code, but they are intended to be interpreted by a virtual machine (VM) written specifically for the host hardware. End-users commonly use a Java Runtime Environment (JRE) installed on their own machine for standalone Java applications, or in a Web browser for Java applets. Standardized libraries provide a generic way to access host-specific features such as graphics, threading, and networking. A major benefit of using bytecode is porting. However, the overhead of interpretation means that interpreted programs almost always run more slowly than programs compiled to native executables would. Just-in-Time (JIT) compilers were introduced from an early stage that compilebytecodes to machine code during runtime.

Java has been tested, refined, extended, and proven by a dedicated community of Java developers, architects and enthusiasts. Java is designed to enable development of portable, high-performance applications for the widest range of computing platforms possible. By making applications available across heterogeneous environments, businesses can provide more services and boost end-user productivity, communication, and collaboration—and dramatically reduce the cost of ownership of both enterprise and consumer applications. Java has become invaluable to developers by enabling them to:

- Write software on one platform and run it on virtually any other platform.
- Create programs that can run within a web browser and access available web services.
- Develop server-side applications for online forums, stores, polls, HTML forms processing, and more.
- Combine applications or services using the Java language to create highly customized applications or services.
- Write powerful and efficient applications for mobile phones, remote processors, microcontrollers, wireless modules, sensors, gateways, consumer products, and practically any other electronic device.

The JRE consists of the Java Virtual Machine (JVM), Java platform core classes, and supporting Java platform libraries. The JRE is the runtime portion of Java software. The Java Plug-in software is a component of the Java Runtime Environment (JRE). The JRE allows applets written in the Java programming language to run inside various browsers. The Java Plug-in software is not a standalone program and cannot be installed separately.The Java Virtual Machine is only one aspect of Java software that is involved in web interaction. The Java Virtual Machine is built right into your Java software download, and helps run Java applications. Java can also be used to develop time critical applications. Real Time Specification for Java (RTSJ) defines and addresses the issues pertaining to the support to be provided by Java for real time environments. The priority based threading model in java provides for real time capabilities.

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM:

Even though cloud storage provide many benefit and high level security but there may be a chances to loss data from the server this will happen both internal attack i.e. cloud owner and external attack i.e. cloud users if the problem accrued in physical process the user cont able to know about the attack because data are persisted in remote machine to identify the misbehavior server this paper introduce new technique called TPA auditing using this service the user can easily know about the misbehavior server because the TPA is not fully dependent cloud server and user it just audit the file based on user requirements and send auditing result to the requested user.

DISADVANTAGE

- ➢ The internal and external affects the data integrity in cloud computing.
- ➢ The identification of misbehaving server is not possible.
- ➢ The Error report of other verification technique report is binary.
- ➢ The other system is not cost effective and it is large scale data storage and it will not provide full guarantee to the user's data.
- ➢ The users can easily attack the data by dynamic operations.

3.2 PROPOSED SYSTEM:

By using TPA Auditing the user can find the misbehavior server but still the problem is exist i.e. we could not recover the data from the loss that's what in our proposed system introduce some more algorithm link segmentation, encoding and encryption along with TPA auditing using all the above algorithm we can achieve the data confident because we just increases the extraction time when the extraction time is increase automatically the security level also increase.

ADVANTAGES

- ➢ It achieves the integration of storage correctness and data error localization, i.e., the identification of misbehaving server(s).
- ➢ It supports secure and efficient dynamic operations on data blocks, including: update, delete, and append.
- ➢ It is highly efficient, lightweight communication and computation cost.
- ➢ It is resilient again malicious data modification attack, and even server colluding attacks.

- ➢ For achieving high security we proposed new algorithm, like segmentation, encryption and encoding through this techniques we can achieve the data confident in cloud environment

FEASIBILITY STUDY

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Technical Feasibility
- Operation Feasibility
- Economical Feasibility

Technical Feasibility

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipments have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access and data security?
-

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a browser based user interface for audit workflow. Thus it provides an easy access to the users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hard requirements for the development of this project are not many and are already available or are available as free as open source. The work for the project is done with the current equipment and existing software technology. Necessary bandwidth exists for providing a fast feedback to the users irrespective of the number of users using the system.

Operational Feasibility

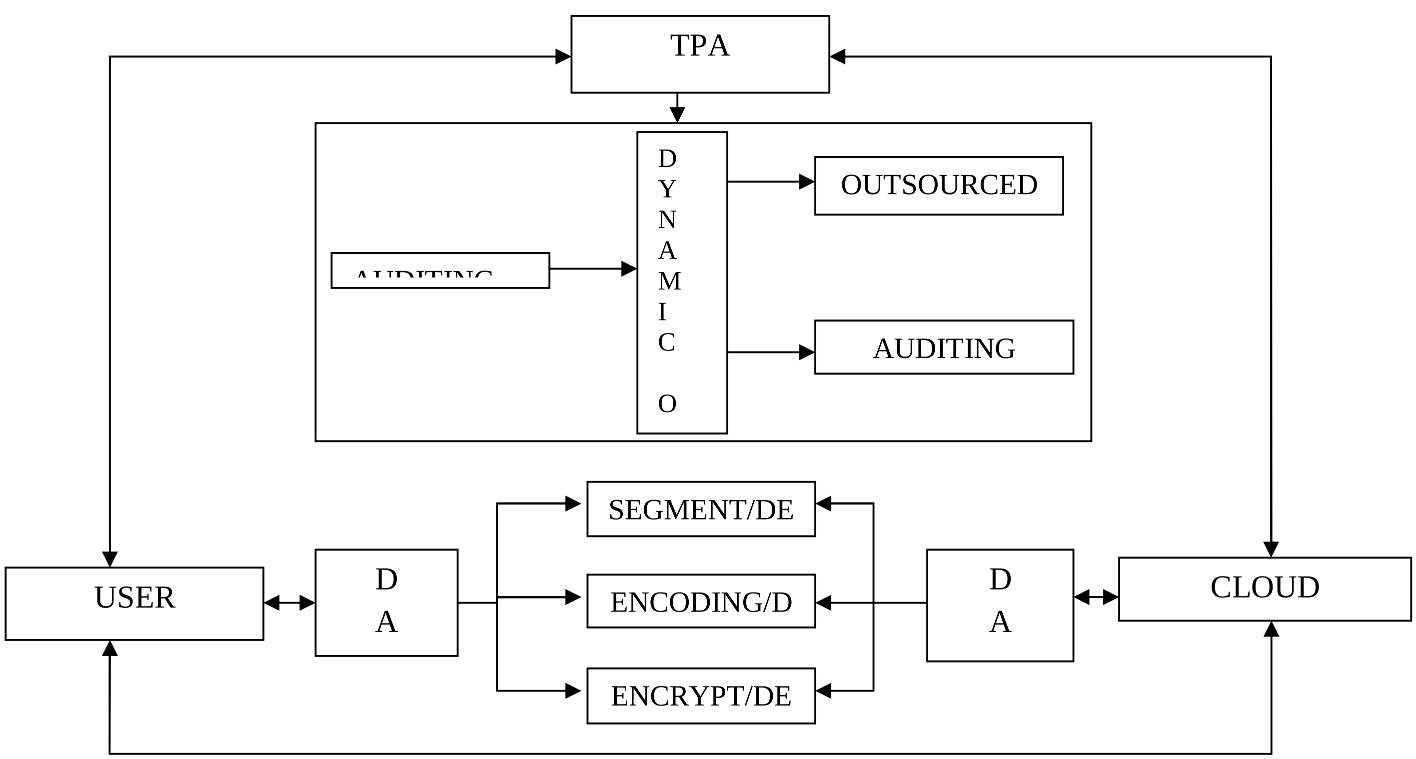
The analyst considers the extent the proposed system will fulfill his departments. That is whether the proposed system covers all aspects of the working system and whether it has considerable improvements. We have found that the proposed "Secure transaction" will certainly have considerable improvements over the existing system.

3.3.3 Economic Feasibility

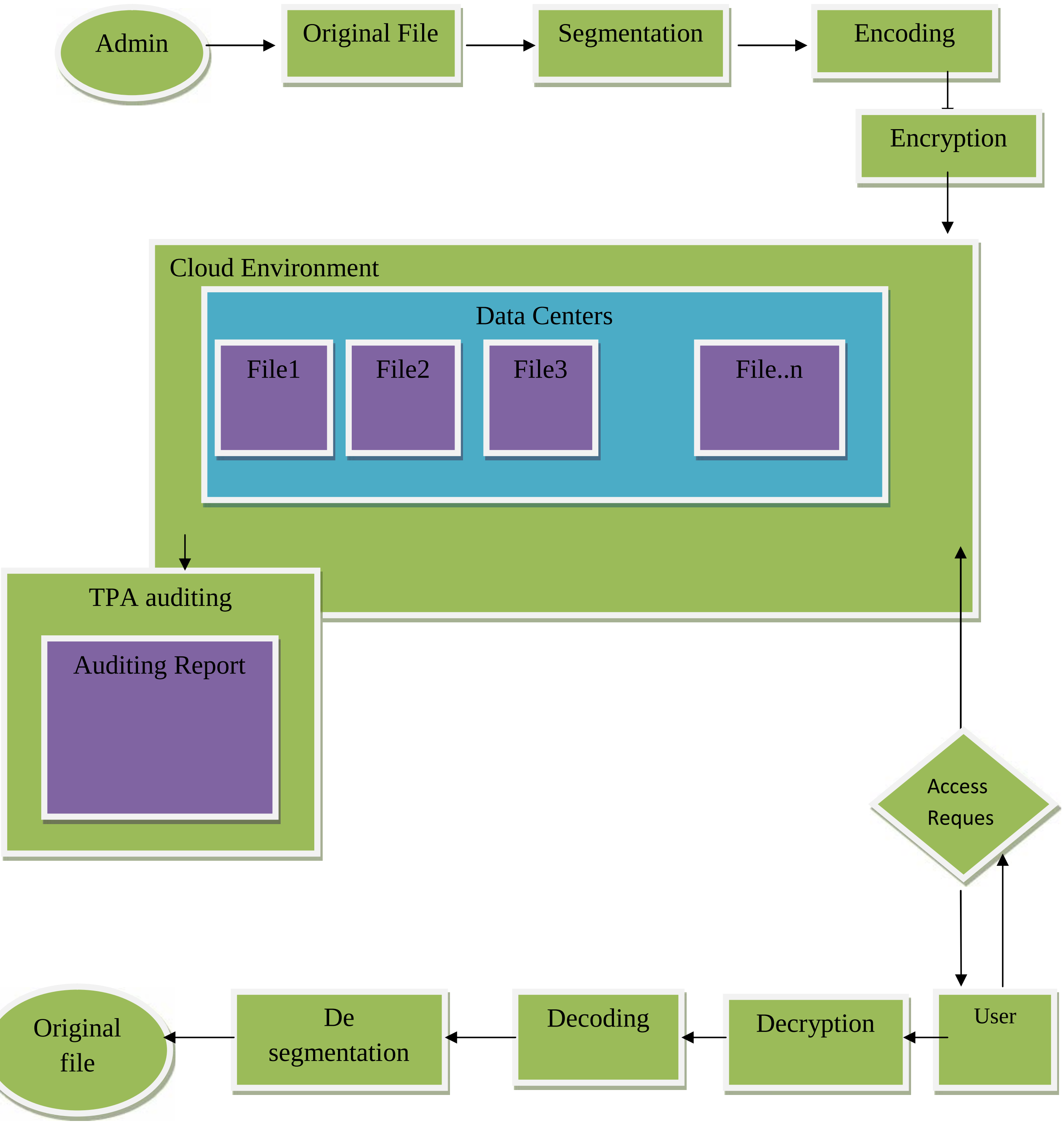
The proposed system is economically feasible because the cost involved in purchasing the hardware and the software are within approachable. Working in this system need not required a highly qualified professional. The operating-environment costs are marginal. The less time involved also helped in its economical feasibility.

SYSTEM DESIGN

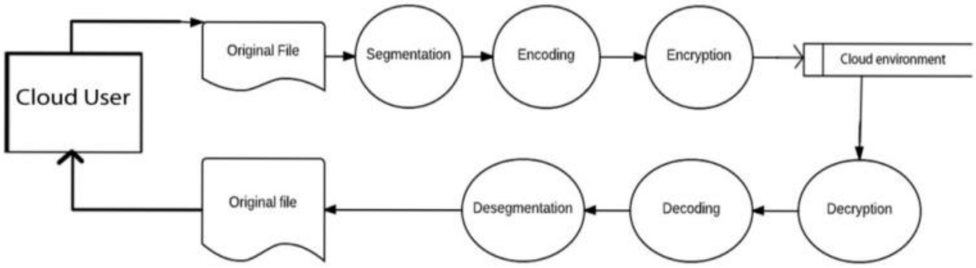
DETAILED SYSTEM DESIGN:



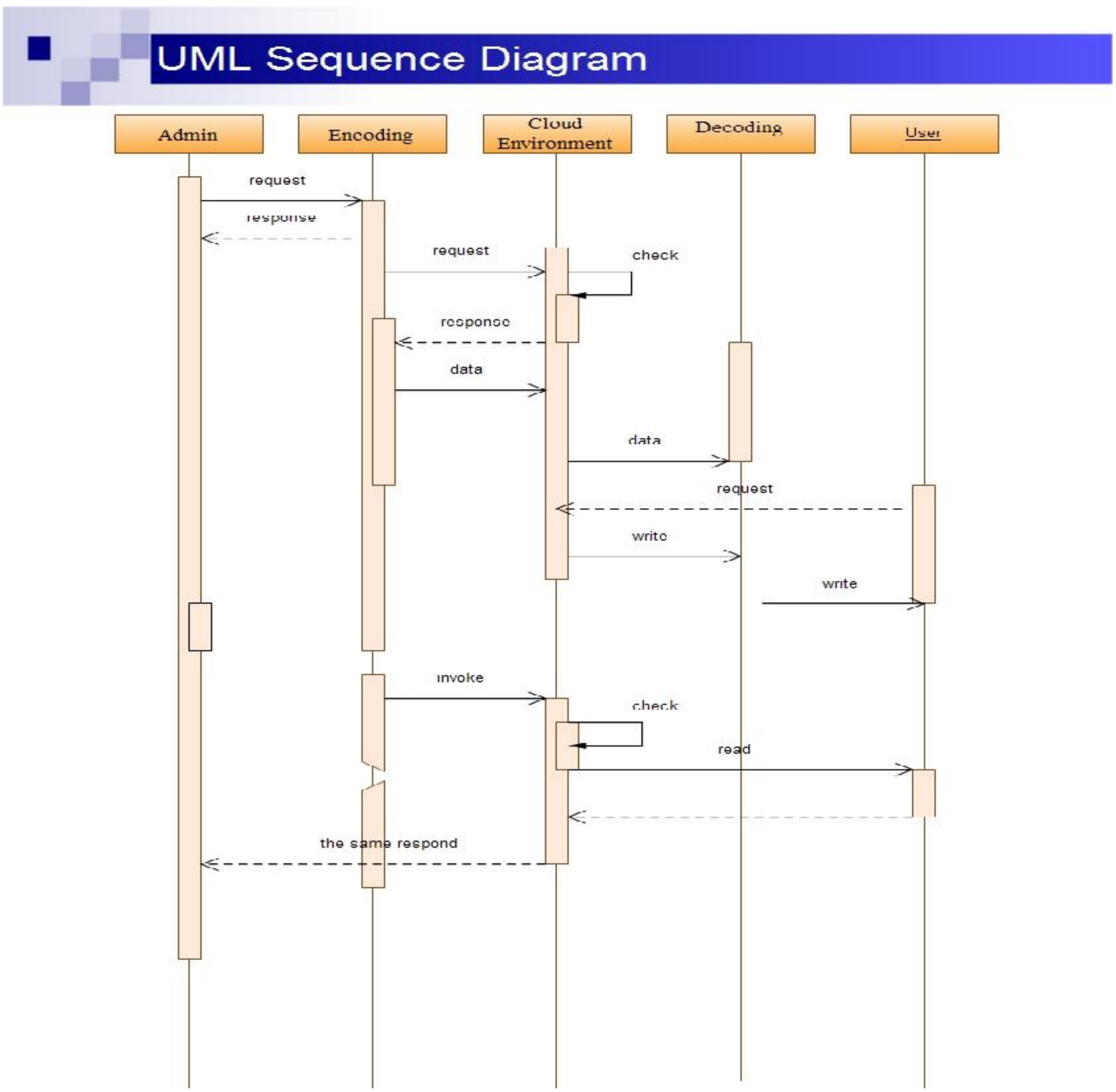
4.1.2 SYSTEM ARCHITECTURE



4.1.3 DATA FLOW DIAGRAM



4.1.4 SEQUENCE DIAGRAM



4.1.5 DATABASE DESIGN

4.2 MODULE DESCRIPTION:

MODULES

1. Authentication
2. Security Process
   - ✓ Segmentation
   - ✓ myEncoding
   - ✓ Encryption
3. Data Access Operation
4. Outsourced Data information
5. Auditing

4.2.1 AUTHENTICATION

It is the act of confirming the truth of an attribute of a datum or entity. This might involve confirming the identity of a person or software program, tracing the origins of an artifact, or ensuring that a product is what it's packaging and labeling claims to be. Authentication often involves verifying the validity of at least one form of identification. User enters the cloud with the registered username and password. It also provides option for creating an account to use the cloud storage. Once created the person can do the dynamic operations with the newly created account. The authentication can be obtained with certain cloud service providers like Amazon EC2, Windows Azure etc. It is approved with certain credit card being provided to the service providers for money transaction.

4.2.2 SECURITY PROCESS

The security process involves all the mechanisms involved to protect the data from being getting corrupted or being misused by a third person. The data is subjected to three security processes namely sementation/ desegmentation, encoding/ decoding and encryption/ decryption.

SEGMENTATION:

User store files in cloud server. For providing security, segment the file which is stored in different location. Segmentation, file size is splits up and save in different location. While download the uploaded file, cloud server desegment the file which was segmented.

ENCODING:

Base-64 algorithm is used for encoding. By encoding more security is provided for the file which is stored in cloud server. Base64 is a group of similar binary-to-text encoding schemes that represent binary data in an ASCII string format by translating it into a radix-64 representation. The Base64 term originates from a specific MIME content transfer encoding.

Base64 encoding schemes are commonly used when there is a need to encode binary data that need to be stored and transferred over media that are designed to deal with textual data. This is to ensure that the data remain intact without modification during transport. Base64 is commonly used in a number of applications including email via MIME, and storing complex data in XML.

ENCRYPTION:

The algorithm is required to be royalty-free for use worldwide and offer security of a sufficient level to protect data. It was to be easy to implement in hardware and software, as well as in restricted environment and offer good defenses against various attack techniques. The entire selection process was fully open to public scrutiny and comment, it being decided that full visibility would ensure the best possible analysis of the designs.

High-level description of the algorithm

1. KeyExpansion—round keys are derived from the cipher key using Rijndael's key schedule. AES requires a separate 128-bit round key block for each round plus one more.
2. InitialRound
   1. AddRoundKey—each byte of the state is combined with a block of the round key using bitwise xor.
3. Rounds
   1. SubBytes—a non-linear substitution step where each byte is replaced with another according to a lookup table.
   2. ShiftRows—a transposition step where each row of the state is shifted cyclically a certain number of steps.
   3. MixColumns—a mixing operation which operates on the columns of the state, combining the four bytes in each column.

4. AddRoundKey

4. Final Round (no MixColumns)
    1. SubBytes
    2. ShiftRows
    3. AddRoundKey

4.2.3 DATA ACCESS OPERATION

Dynamic operations like delete, download, append and modifications of files done in cloud server based on user access. The file can be added to the cloud for keeping it safe in the cloud storage. The file added is stored by adapting all the security processes defined. The file can be downloaded from the cloud after proper authentication being made. Any modifications made will be stored in the cloud made by the user. If any unknown person enters the cloud without proper authentication and misuses the file in the cloud, it must be managed by the TPA and must be alerted for the user to notice it. These operations are done to communicate with the cloud and thus making all the transfer of files or any information into the cloud for storing and retrieving it in a safe way.

4.2.4 OUTSOURCED DATA INFORMATION

The metadata refers to data about data. It is descriptive information about a particular data set, object, or resource, including how it is formatted, and when and by whom it was collected. Metadata summarizes basic information about data, which can make finding and working with particular instances of data easier. For example, author, date created and date modified and file size are examples of very basic document metadata. Having the ability to filter through that metadata makes it much easier for someone to locate a specific document. The information provided in a document might have further descriptions that explain the origin or context of the information itself. Metadata (metacontent) are defined as the data providing information about one or more aspects of the data, such as:

1. Means of creation of the data
2. Purpose of the data
3. Time and date of creation
4. Creator or author of the data
5. Location on a computer network where the data were created

4.2.5 AUDITING

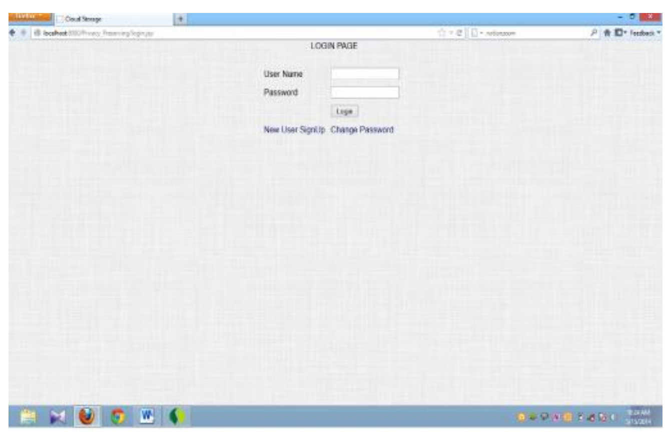
Auditing is a systematic and independent examination of data, statements, records, operations and of the data residing on the cloud for a stated purpose. Auditing main purpose or object is to find the opinion of an auditor about correctness and reliability. Audits should always be an independent evaluation that will include some degree of quantitative and qualitative analysis. These standards assure third parties or external users that they can rely upon the auditor's opinion. In any auditing the auditor perceives and recognizes the data for examination, collects evidence, evaluates the same and on this basis formulates the judgment which is communicated through the audit report. The TPA (Third Party Auditor) is the one who audits the cloud system. The TPA provides an audit report to the user for verifying the users' data whether had been misused or not by any third person. The audit result will indicate an error if there is any change in the data or any unauthorized access of the data from the cloud.

IMPLEMENTATION

INPUT DESIGN:

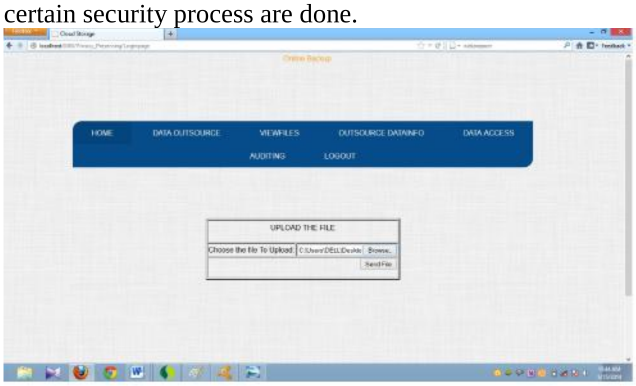
The dynamic and open characteristics of the cloud computing, continuing malicious attacks happen frequently. Combining the idea of trusted cloud, a trust-based defensive system model for cloud computing has been constructed to guarantee the cloud security. Through real-timely monitoring, users' behavior evidences have been obtained and standardized. It gradually determines the weights of behavior evidences, achieves quantitative assessment of behavioral trust to provide great security defense for users, multiple detection engines have been used to conduct a comprehensive inspection of suspicious files and integrated decisions have been made.

The cloud system is a live drive as shown in figure where the data is stored in the cloud through online with the help of web browsers. They can be accessed from anywhere, where there is internet access in the entire world. The cloud storage is not made in the same place where the access is done but can be stored anywhere around the world.



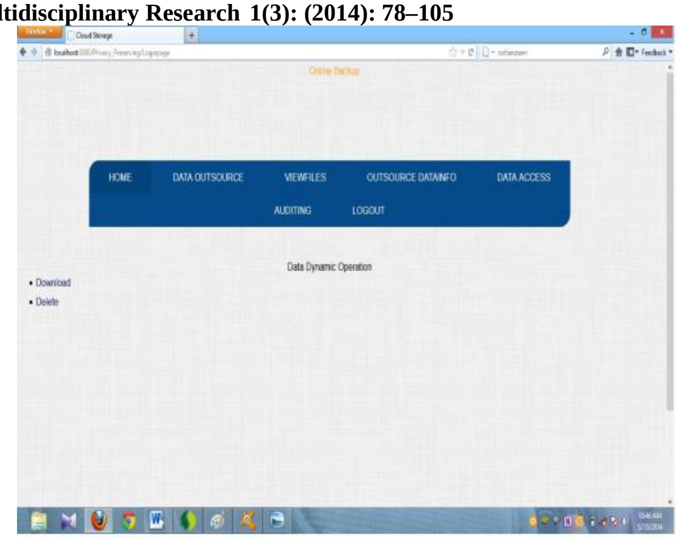
4.3.2 DATAOUTSOURCE:

Using a storage service to store file uploads provides an order of magnitude, scalability, reliability, and speed gain than just storing files on a local file system. The file to be stored is to be chosen to upload in the cloud as shown in figure . The file will be segmented and further security processes are made, thus avoiding the unnecessary access of the file from the cloud. The data will be stored in the cloud but in different data centers that was created in the cloud for security purpose. There are totally six data centers being created for placing the data in different location after performing segmentation. The cloud storage is not done in a particular place or country but can be done in different places. The data is not just stored in one place but are stored in two different places. Hence when any disastrous occurs, the data from one place get destroyed or eradicated the data from other place will be used. Thus avoids data loss. The data will not be retrieved by any unauthorized person since certain security process are done.



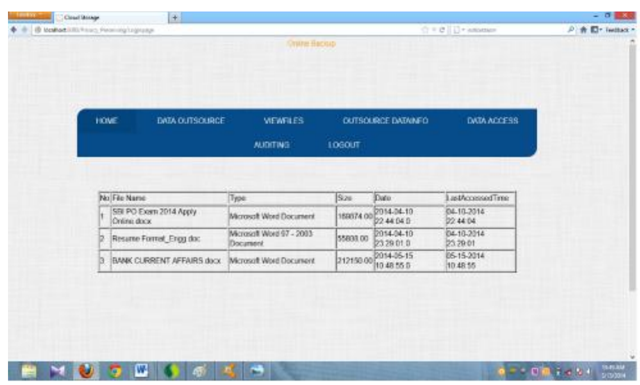
4.3.3 DATA ACCESS OPERATIONS:

Data access operations like delete, download of files done in cloud server based on user access as shown in figure . The file can be added to the cloud for keeping it safe in the cloud storage. The file added is stored by adapting all the security processes defined. The file can be downloaded from the cloud after proper authentication being made. Any modifications made will be stored in the cloud made by the user. If any unknown person enters the cloud without proper authentication and misuses the file in the cloud , it must be managed by the TPA and must be alerted for the user to notice it. The data access operations are those involved or invoked to make operations on the data uploaded or stored in the cloud, also any new uploading of data can also be done. These operations can be made use of by the authorized persons. Thus proper authentication is interfaced with it to have a proper data storage. Any improper entry detected will be indicated in the auditing report.



4.3.4 OUTSOURCED DATA INFORMATION :

Outsourced is the information or data in cloud that is not exactly the data itself but its data about the data. In a broader way you can view metadata as an encyclopedia of the data warehouse. Outsourced datainfo in cloud contains information about the various data or files used or stored in cloud as shown in figure . The outsurced data information is provided by the TPA (Third Party Auditor) and thus allows the user to verify the data or files stored in the cloud based on the creation time , accessed time etc. The outsourced data information thus generated is used for generating the auditing report. The auditing report makes use of the data created and accessed by the user. Thus any misuse of the file can be obtained with the outsourced data information. The change in the size of the file can also be used to indicate the errorness in the data stored in the cloud. The data about the data in the cloud is to be generated for security purposes. The security provided will be analyzed only through the auditing report which is generated with the use of outsourced data report.



SOURCE CODE:
DATAOUTSOURCE.Java

```
public class Fileupload extends HttpServlet {

        private static final long serialVersionUID = 1L;

publicFileupload() {

super();

      }

        protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
{

doPost(request,response);

        }

        protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException,
IOException {

                HttpSessionUserId=request.getSession();

                ArrayList<Long> al=new ArrayList<Long>();

                String userid=UserId.getAttribute("userid")!=null?UserId.getAttribute("userid").toString():" ";

                if(userid.equals(" ")){

                        RequestDispatcher   rd2= request.getRequestDispatcher("FileUpload.jsp");

                request.setAttribute("msg", "You are not logged in");

rd2.forward(request, response);

                }

                else

                {

                int rem=0;

        int size=0;

int File1=0;

int File2=0;

int File3=0;

int File4=0;

int File5=0;
```

```
String[]
storage={"DataCenter1","DataCenter2","DataCenter3","DataCenter4","DataCenter5","DataCenter6",};
        String contentType = request.getContentType();
    if(contentType != null &&contentType.indexOf("multipart/form-data") >= 0)
        {
    DataInputStream in = new DataInputStream(request.getInputStream());
    intformDataLength = request.getContentLength();
    bytedataBytes[] = new byte[formDataLength];
    intbyteRead = 0;

    for(inttotalBytesRead = 0; totalBytesRead<formDataLength; totalBytesRead
+= byteRead)
    byteRead = in.read(dataBytes, totalBytesRead, formDataLength);
            String file = new String(dataBytes);
            String saveFile = file.substring(file.indexOf("filename=\"") + 10);
    saveFile = saveFile.substring(0, saveFile.indexOf("\n"));
    saveFile = saveFile.substring(saveFile.lastIndexOf("\\") + 1, saveFile.indexOf("\""));
    out.println(saveFile);
    intlastIndex = contentType.lastIndexOf("=");
            String boundary = contentType.substring(lastIndex + 1, contentType.length());
    intpos = file.indexOf("filename=\"");
    pos = file.indexOf("\n", pos) + 1;
    pos = file.indexOf("\n", pos) + 1;
    pos = file.indexOf("\n", pos) + 1;
    intboundaryLocation = file.indexOf(boundary, pos) - 4;
    intstartPos = file.substring(0, pos).getBytes().length;
    intendPos = file.substring(0, boundaryLocation).getBytes().length;
    FileOutputStreamfileOut = new FileOutputStream(saveFile);
```

```
fileOut.write(dataBytes, startPos, endPos - startPos);
    fileOut.flush();
    fileOut.close();
        Connection connection = null;
         String connectionURL = "jdbc:mysql://localhost:3306/towards";
    ResultSetrs = null;
    PreparedStatementpsmnt = null;
         String name = null;
    try
  {
    int count = 0;
    int count1 = 0;
    Class.forName("com.mysql.jdbc.Driver").newInstance();
    connection = DriverManager.getConnection(connectionURL, "mysql", "mysql");
           Statement st = connection.createStatement();
           File f = new File(saveFile);
    psmnt = connection.prepareStatement("insert into fileupload(no,path,username,image) values(?,?,?,?)");
    FileInputStreamfis = new FileInputStream(f);
    psmnt.setInt(1, count++);
    psmnt.setString(2, saveFile);
    psmnt.setString(3, userid);
    psmnt.setBinaryStream(4, fis, (int)f.length());
    int s = psmnt.executeUpdate();
    if(s!=0){}
    count++;
           String n = f.getName();
    if(count != 0)
           {
```

```
File file2 = new File(saveFile);

        longfilesize = file2.length();

                String filename = file2.getName();

                Calendar currentDate = Calendar.getInstance();

        SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd HH-mm-ss");

                String dateNow = formatter.format(currentDate.getTime());

        JFileChooserfiletype = new JFileChooser();

                String fileTypeName = filetype.getTypeDescription(file2);

        long l=file2.lastModified();

        SimpleDateFormatsdf = new SimpleDateFormat("MM-dd-yyyyHH:mm:ss");

                count1 = st.executeUpdate((new StringBuilder("insert into
storage(filename,type,size,added,username,lastaccessedtime,downloadtime)values('")).append(filename).append("','").append(file
TypeName).append("','").append(filesize).append("','").append(dateNow).append("','").append(userid).append("','").append(sdf.fo
rmat(l)).append("','").append("0").append("')").toString());

                String query = (new StringBuilder("select * from fileupload where
path='")).append(n).append("'").toString();

        rs = st.executeQuery(query);

        if(rs.next())

                {

        InputStreamfullimage = rs.getBinaryStream(4);

        name = rs.getString(2);

        byte b[] = new byte[fullimage.available()];

        fullimage.read(b);

        OutputStream outs = new Base64OutputStream(new FileOutputStream((new
StringBuilder("c:/CloudStorage1/"+userid+"/")).append(name).toString()));

        outs.write(b);

        outs.close();

        size=b.length;

        rem=size%6;

size=size/6;

    File1=0+size;
```

```
File2=File1+size;
File3=File2+size;
File4=File3+size;
File5=File4+size+rem;
KeyGenerator kgen1 = KeyGenerator.getInstance("AES");
kgen1.init(128);
SecretKey key1 = kgen1.generateKey();
AESEncrypter encrypter1 = new AESEncrypter(key1);
KeyGenerator kgen2 = KeyGenerator.getInstance("AES");
kgen2.init(128);
SecretKey key2 = kgen2.generateKey();
AESEncrypter encrypter2 = new AESEncrypter(key2);
KeyGenerator kgen3 = KeyGenerator.getInstance("AES");
kgen3.init(128);
SecretKey key3 = kgen3.generateKey();
AESEncrypter encrypter3 = new AESEncrypter(key3);
KeyGenerator kgen4 = KeyGenerator.getInstance("AES");
kgen4.init(128);
SecretKey key4 = kgen4.generateKey();
AESEncrypter encrypter4 = new AESEncrypter(key4);
KeyGenerator kgen5 = KeyGenerator.getInstance("AES");
kgen5.init(128);
SecretKey key5 = kgen5.generateKey();
AESEncrypter encrypter5 = new AESEncrypter(key5);
KeyGenerator kgen6 = KeyGenerator.getInstance("AES");
kgen6.init(128);
SecretKey key6 = kgen6.generateKey();
AESEncrypter encrypter6 = new AESEncrypter(key6);
```

```
OutputStream outs1 = new Base64OutputStream(new FileOutputStream((new
StringBuilder("c:/"+storage[0]+"/")).append("1"+name).toString()));

outs1.write(b, 0,size);



OutputStream outs2 = new Base64OutputStream(new FileOutputStream((new
StringBuilder("c:/"+storage[1]+"/")).append("2"+name).toString()));

outs2.write(b, File1,size);

OutputStream outs3 = new Base64OutputStream(new FileOutputStream((new
StringBuilder("c:/"+storage[2]+"/")).append("3"+name).toString()));

outs3.write(b, File2,size);

OutputStream outs4 = new Base64OutputStream(new FileOutputStream((new
StringBuilder("c:/"+storage[3]+"/")).append("4"+name).toString()));

outs4.write(b, File3,size);

OutputStream outs5 = new Base64OutputStream(new FileOutputStream((new
StringBuilder("c:/"+storage[4]+"/")).append("5"+name).toString()));

outs5.write(b, File4,size);

OutputStream outs6 = new Base64OutputStream(new FileOutputStream((new
StringBuilder("c:/"+storage[5]+"/")).append("6"+name).toString()));

outs6.write(b, File5,size);

outs1.close();

outs2.close();

outs3.close();

outs4.close();

outs5.close();

outs6.close();

encrypter1.encrypt(new FileInputStream("c:/"+storage[0]+"/"+"1"+name), new
FileOutputStream("c:/"+storage[0]+"/Encryption/"+"1"+name));

encrypter2.encrypt(new FileInputStream("c:/"+storage[1]+"/"+"2"+name), new
FileOutputStream("c:/"+storage[1]+"/Encryption/"+"2"+name));

encrypter3.encrypt(new FileInputStream("c:/"+storage[2]+"/"+"3"+name), new
FileOutputStream("c:/"+storage[2]+"/Encryption/"+"3"+name));

encrypter4.encrypt(new FileInputStream("c:/"+storage[3]+"/"+"4"+name), new
FileOutputStream("c:/"+storage[3]+"/Encryption/"+"4"+name));
```

```
encrypter5.encrypt(new FileInputStream("c:/"+storage[4]+"/"+"5"+name), new
FileOutputStream("c:/"+storage[4]+"/Encryption/"+"5"+name));

encrypter6.encrypt(new FileInputStream("c:/"+storage[5]+"/"+"6"+name), new
FileOutputStream("c:/"+storage[5]+"/Encryption/"+"6"+name));

                File Datacenter;

int a=0;

intaa=0;

for(int k=0;k<6;k++)

                  {

            Datacenter=new File((new StringBuilder("c:/"+storage[aa++]+"/")).append(aa+name).toString());

            al.add(Datacenter.length());

}}}

            try{

                        Class.forName("com.mysql.jdbc.Driver").newInstance();

                        connection = DriverManager.getConnection(connectionURL, "mysql", "mysql");

                                    Statement st5 = connection.createStatement();

                        psmnt = connection.prepareStatement("insert into matrix(filename,username,v1,v2,v3,v4,v5,v6)
values(?,?,?,?,?,?,?,?)");

            psmnt.setString(1, name);

                        psmnt.setString(2,userid);

                        psmnt.setLong(3,al.get(0));

                        psmnt.setLong(4,al.get(1));

                        psmnt.setLong(5,al.get(2));

                        psmnt.setLong(6,al.get(3));

                        psmnt.setLong(7,al.get(4));

                        psmnt.setLong(8,al.get(5));

                        intss = psmnt.executeUpdate();

                       }

            catch(Exception e)

                      {
```

```
System.out.println(e);
                    }
          try{
                      File ff;
                     ArrayList<String>l=new ArrayList<String>();
                                         for(int k=1;k<=6;k++)
                               {
                     ff=new File("c:/DataCenter"+k+"/"+k+name);
                                                   long t=ff.lastModified();
                                                             SimpleDateFormatsm=new SimpleDateFormat("MM-dd-yyyyHH:mm:ss");
                                                             l.add(sm.format(t));
                     Class.forName("com.mysql.jdbc.Driver").newInstance();
                     connection = DriverManager.getConnection(connectionURL, "mysql", "mysql");
                                Statement st5 = connection.createStatement();
                     psmnt = connection.prepareStatement("insert into
datacenterinfo(username,filename,datacenter1,datacenter2,datacenter3,datacenter4,datacenter5,datacenter6)
values(?,?,?,?,?,?,?,?)");
                     psmnt.setString(1,userid);
                     psmnt.setString(2,name);
                     psmnt.setString(3,l.get(0));
                     psmnt.setString(4,l.get(1));
                     psmnt.setString(5,l.get(2));
                     psmnt.setString(6,l.get(3));
                     psmnt.setString(7,l.get(4));
                     psmnt.setString(8,l.get(5));
                     intss = psmnt.executeUpdate();
                    }
          catch(Exception e)
                    {
```

```
System.out.println(e);
                    }
          if(count1 != 0)
                       {
          RequestDispatcher   rd1 = request.getRequestDispatcher("Link.jsp");
          rd1.forward(request, response);
                       }
                    }
          catch(Exception e)
                    {
          System.out.println(e);
          }     }
}       }
```

OUTSOURCEDATA.Java

```
public class MetaInfo extends HttpServlet {
          private static final long serialVersionUID = 1L;
publicMetaInfo() {
super();
   }
          protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
{
                              doPost(request,response);
          }
          protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException,
IOException {
                    String Filename=request.getParameter("menu");
                    PrintWriter out=response.getWriter();
                    HttpSessionfname=request.getSession();
                    HttpSessionftype=request.getSession();
```

```
HttpSessionfsize=request.getSession();
            HttpSessionfownername=request.getSession();
            HttpSessionfdatacenter=request.getSession();
            HttpSessionfdate=request.getSession();
            try{
                Class.forName("com.mysql.jdbc.Driver");
                Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/towards","mysql","mysql");
                Statement st=con.createStatement();


                ResultSetrs=st.executeQuery("select * from storage where filename='"+Filename+"'");
                out.println("<html><body><center>File Information</center><table border=2 align=center>");
                if(rs.next())
                {
                    out.println("<tr><td>FileName</td><td><a
href=Metainfodownload.jsp>"+rs.getString(2)+"</a></td></tr>");
                    out.println("<tr><td>FileType</td><td>"+rs.getString(3)+"</td></tr>");
                    out.println("<tr><td>FileSize(in Kb)</td><td>"+rs.getString(4)+"</td></tr>");
                    out.println("<tr><td>FileUploaded Date</td><td>"+rs.getString(5)+"</td></tr>");
                    out.println("<tr><td>OwnerName</td><td>"+rs.getString(6)+"</td></tr>");
                    out.println("<tr><td>LastAccessedTime</td><td>"+rs.getString(7)+"</td></tr>");


                    fname.setAttribute("FILENAME", rs.getString(2));
                    ftype.setAttribute("FILETYPE", rs.getString(3));
                    fsize.setAttribute("FILESIZE", rs.getString(4));
                    fdate.setAttribute("FILEDATE", rs.getString(5));
                    fownername.setAttribute("FILEOWNERNAME", rs.getString(6));
                    fdatacenter.setAttribute("DATACENTER", rs.getString(7));
                }
```

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

DATA ACCESS

DOWNLOAD.Java

```
public class Download extends HttpServlet {
        private static final long serialVersionUID = 1L;


public Download() {
super();
   }
        protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
{
                doPost(request,response);
        }
        protected void processRequest(HttpServletRequest request, HttpServletResponse response)
                throwsServletException, IOException
                  {
                ArrayList<String>al=new ArrayList<String>();
                        HttpSessionUserId=request.getSession();
                        String userid=UserId.getAttribute("userid").toString();
                        ArrayList<Long>oldtime=new ArrayList<Long>();
                        String[]
storage={"DataCenter0","DataCenter1","DataCenter2","DataCenter3","DataCenter4","DataCenter5","DataCenter6"};
                ServletOutputStream op =null;
                     String s = request.getParameter("menu");
                     String original_filename = s;
```

```
String name="";
        longtstart=System.currentTimeMillis();
        try{
          File delfile=new File("c:/backup/"+s+"");
        if(delfile.exists())
          {
                delfile.delete();
          }
        FileOutputStreamfout=(new FileOutputStream("c:/backup/"+s+"",true));
              File fff=null;

 InputStream fin;
        int l=0;
                        for(int i=1;i<=6;i++)
                          {
                                name=i+s;
                                String src="c:/"+storage[i]+"/"+name+"";
                                fff=new File(src);
                                oldtime.add(fff.lastModified());
                                byte b[] = new byte[(int) fff.length()];
                        for(fin = new Base64InputStream(new FileInputStream(fff)); fin != null && (l= fin.read(b))
!= -1;)
                        {
                                fout.write( b, 0, l);
                        }
                                fout.flush();
                          }
                          String filename = "c:/backup/"+s+"";
                File f = new File(filename);
```

```
int length = 0;

op = response.getOutputStream();

        ServletContext context = getServletConfig().getServletContext();

        String mimetype = context.getMimeType(filename);

        response.setContentType(mimetype == null ? "application/octet-stream" :mimetype);

        response.setContentLength((int)f.length());

        response.setHeader("Content-Disposition", (new StringBuilder("attachment;
filename=\"")).append(original_filename).append("\"").toString());

        bytebbuf[] = new byte[filename.length()];

        InputStream in;

                for(in = new FileInputStream(f); in != null && (length = in.read(bbuf)) != -1;)



                {

                        op.write(bbuf, 0, length);

                }

        op.flush();

        in.close();

        op.close();

fout.close();

long tend=System.currentTimeMillis();

doubletseconds=((double)(tend-tstart))/(double)1000;

    }

catch(Exception e)

{

System.out.println(e);  }

try{

            Calendar currentDate = Calendar.getInstance();

SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd HH-mm-ss");
```

```
String dateNow = formatter.format(currentDate.getTime());

Class.forName("com.mysql.jdbc.Driver");

Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/towards","mysql","mysql");

Statement st=con.createStatement();

Statement st1=con.createStatement();

ResultSetrs=st1.executeQuery("select * from datacenterinfo where
filename='"+s+"'");

intkk=st.executeUpdate("update storage set downloadtime='"+dateNow+"' where
filename='"+s+"'");

}

catch(Exception e)

{

System.out.println(e);

}

}

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException,
IOException {

processRequest(request, response);

}

}
```

DELETE.Java

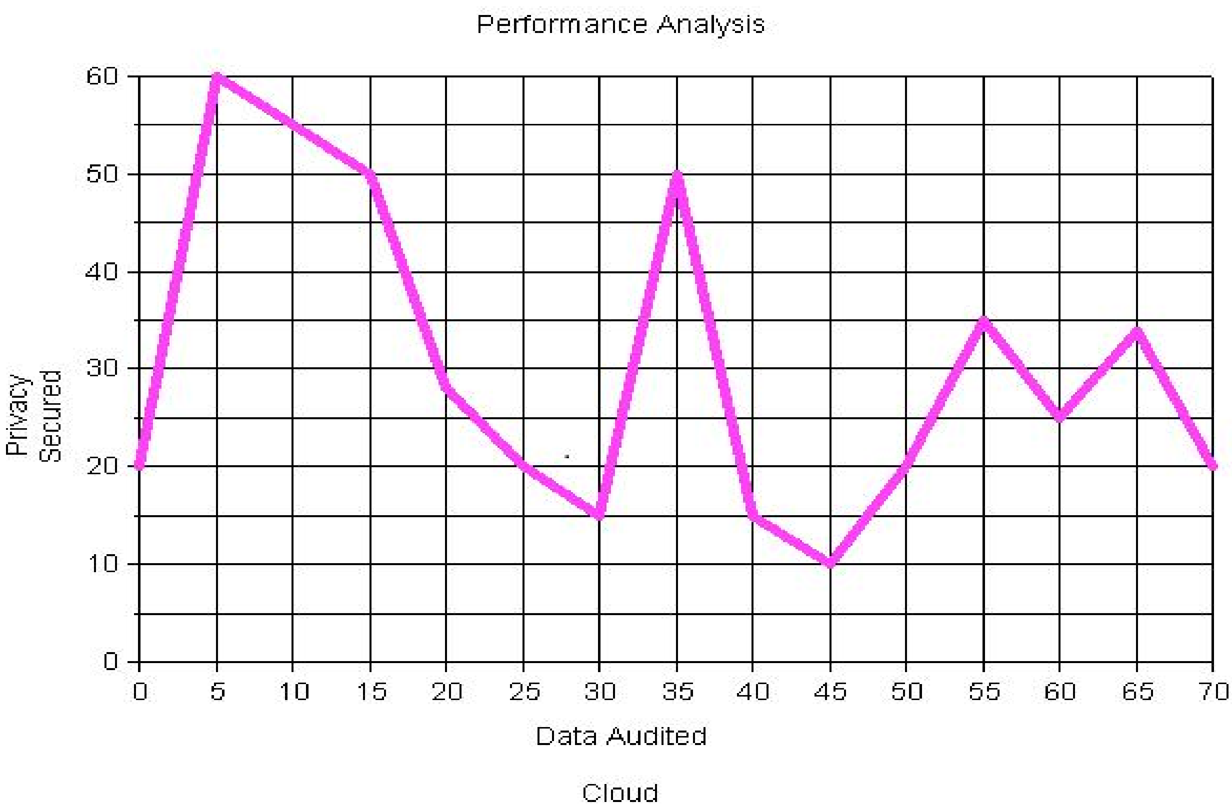
```
public class Delete extends HttpServlet {

private static final long serialVersionUID = 1L;

public Delete() {

super();

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
{

doPost(request,response);
```

```
}

        protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException,
IOException {

                HttpSessionUserId=request.getSession();

                String userid=UserId.getAttribute("userid").toString();

                String[]
storage={"DataCenter0","DataCenter1","DataCenter2","DataCenter3","DataCenter4","DataCenter5","DataCenter6"};

    String s = request.getParameter("menu");

    String name="";

    File dfile;

try{

        File delfile=new File("c:/CloudStorage1/"+userid+"/"+s+"");


delfile.delete();

                for(int i=1;i<=6;i++)

                  {

                        name=i+s;

                        String src="c:/"+storage[i]+"/"+name+"";

                        dfile=new File(src);

                        dfile.delete();

                          }

                                Class.forName("com.mysql.jdbc.Driver");

                                Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/towards","mysql","mysql");

                                Statement st=con.createStatement();

                                int k=st.executeUpdate("delete from storage where filename='"+s+"' and
username='"+userid+"'");

                                int k1=st.executeUpdate("delete from fileupload where path='"+s+"' and
username='"+userid+"'");

                                int k2=st.executeUpdate("delete from matrix where filename='"+s+"' and
username='"+userid+"'");
```

```
        int k3=st.executeUpdate("delete from datacenterinfo where filename='"+s+"' and username='"+userid+"'");

            RequestDispatcherrd=request.getRequestDispatcher("delete.jsp");

            request.setAttribute("Dmsg", "File Was Deleted Successfully");

            rd.forward(request, response);

        }

catch(Exception e)

    {

System.out.println(e);

    }

    }

}
```

RESULTS

Graph is one of the efficient method to display the result visually. The graph helps to evaluate the throughput performance. The performance analysis means the total efficiency of the system. The performance analysis is used to represent, how efficiently the system works. The performance analysis is represented by the data audited by the TPA (Third Party Auditor) in the Y-axis and privacy secured in the X-axis.



SYSTEM TESTING

6.1 INTRODUCTION

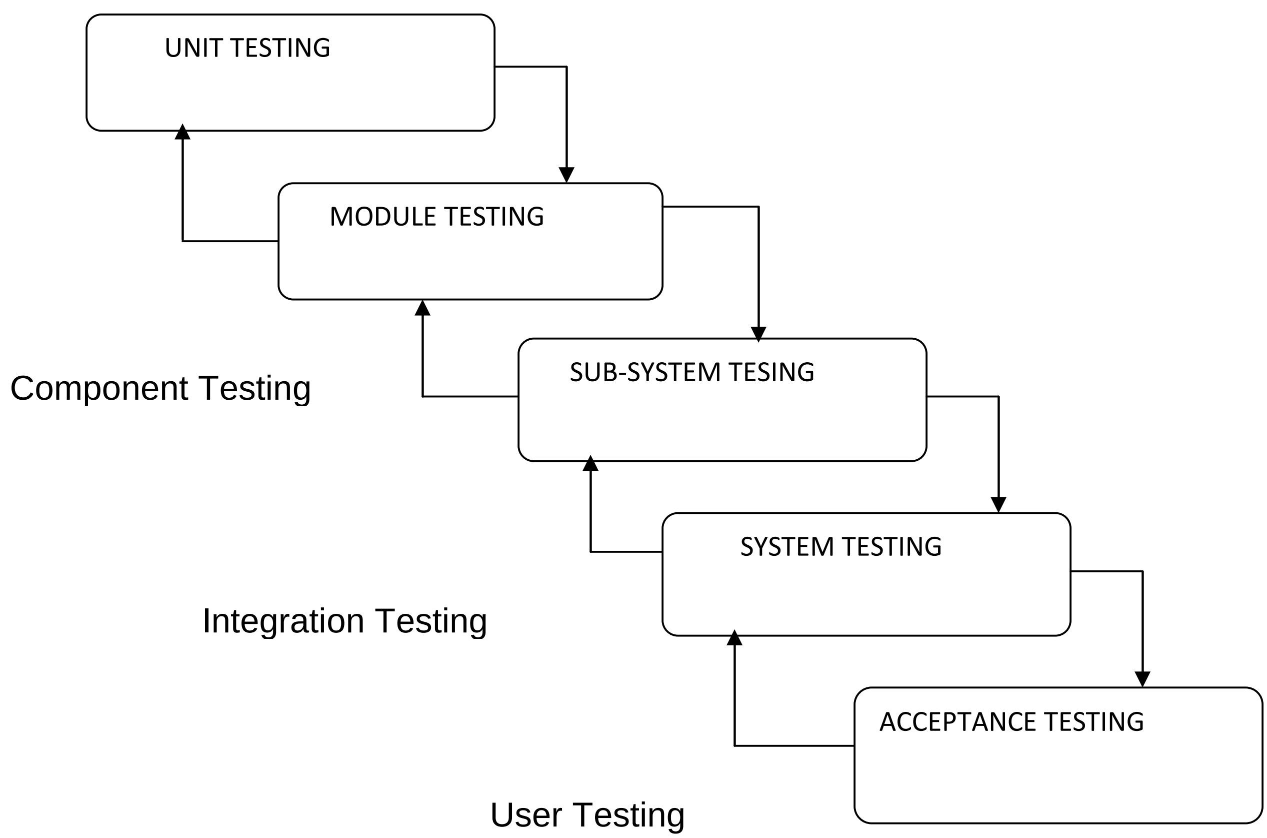
Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

6.2 STRATEGIC APPROACH TO SOFTWARE TESTING

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing progress is done by moving outward along the spiral to integration testing, where the focus is on the design and the construction of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.



6.3 Unit Testing

Unit testing focuses verification effort on the smallest unit of software design, the module. The unit testing we have is white box oriented and some modules the steps are conducted in parallel.

6.3.1 WHITE BOX TESTING

This type of testing ensures that

1. All independent paths have been exercised at least once
2. All logical decisions have been exercised on their true and false sides
3. All loops are executed at their boundaries and within their operational bounds
4. All internal data structures have been exercised to assure their validity.

To follow the concept of white box testing we have tested each form .we have created independently to verify that Data flow is correct, All conditions are exercised to check their validity, All loops are executed on their boundaries.

6.3.2 BASIC PATH TESTING

Established technique of flow graph with Cyclomatic complexity was used to derive test cases for all the functions. The main steps in deriving test cases were:

Use the design of the code and draw correspondent flow graph.

Determine the Cyclomatic complexity of resultant flow graph, using formula:

V(G)=E-N+2 or

V(G)=P+1 or

V(G)=Number Of Regions

Where V(G) is Cyclomatic complexity,

E is the number of edges,

N is the number of flow graph nodes,

P is the number of predicate nodes.

6.3.3 CONDITIONAL TESTING

In this part of the testing each of the conditions were tested to both true and false aspects. And all the resulting paths were tested. So that each path that may be generate on particular condition is traced to uncover any possible errors.

6.3.4 DATA FLOW TESTING

This type of testing selects the path of the program according to the location of definition and use of variables. This kind of testing was used only when some local variable were declared. The *definition-use chain* method was used in this type of testing. These were particularly useful in nested statements.

6.3.5 LOOP TESTING

In this type of testing all the loops are tested to all the limits possible. The following exercise was adopted for all loops:

1. All the loops were tested at their limits, just above them and just below them.
2. All the loops were skipped at least once.
3. For nested loops test the inner most loop first and then work outwards.
4. For concatenated loops the values of dependent loops were set with the help of connected loop.

CONCLUSION AND FUTURE ENHANCEMENT

A privacy-preserving public auditing system for data storage security in cloud computing is proposed. The Homomorphic Linear Authenticator (HLA) is utilized and random masking is used to guarantee that the TPA would not learn any knowledge about the data content stored on the cloud server during the efficient auditing process, which not only eliminates the burden of cloud user from the tedious and possibly expensive auditing task, but also alleviates the users' fear of their outsourced data leakage. Considering TPA may concurrently handle multiple audit sessions from different users for their outsourced data files, further extend the privacy-preserving public auditing protocol into a multiuser setting, where the TPA can perform multiple auditing tasks in a batch manner for better efficiency. The data is protected through security processes which include segmentation, encoding and encryption. Thus the data in the cloud is said to have certain protection schemes and also found to be effective by monitoring the data access by the TPA and makes the stored data to be protected effectively.

The security process followed is to be implemented on the cloud for further protection of the data than those provided by the cloud service providers and as a future extension it is also insisted to further impose any other security process to be implemented on the cloud for any unauthorized access and misuse of the data. It is now to leave the full-fledged implementation of the mechanism on commercial public cloud as an important future extension, which is expected to robustly cope with very large scale data and thus encourage users to adopt cloud storage services more confidently.

REFERENCE

[1] Cong Wang, Sherman S.M. Chow, Qian Wang, KuiRen, and Wenjing Lou, "Privacy-Preserving Public Auditing for Secure Cloud Storage," - IEEE Transactions on Computers, vol.62, no. 2, February 2013.

[2] Qian Wang, Cong Wang, KuiRen, Wenjing Lou, and Jin Li, "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing," - IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 5, May 2011.

[3] R. Nithiavathy, "Data Integrity and Data Dynamics with Secure Storage Service in Cloud," -Proceedings of the 2013

International Conference on Pattern Recognition, Informatics and Mobile Engineering, February 21-22.

[4] Jun Feng, Yu Chen, Douglas H. Summerville, “A Fair Multi-Party Non-Repudiation Scheme for Storage Clouds,” -IEEE Trans. Service Computing, vol. 5, no. 2, 220-232, Apr.-June 2011.

[5] AyadBarsoum and Anwar Hasan, “Enabling Dynamic Data and Indirect Mutual Trust for Cloud Computing Storage Systems,” - IEEE Trans. Parallel and Distributed Systems, vol. 22, no.5, pp. 847-859, May 2012.

[6] Jens-Matthias Bohli, Nils Gruschka, Meiko Jensen, Luigi Lo Iacono, and Ninja Marnau, “Security and Privacy-Enhancing Multicloud Architectures”, - IEEE Transactions on Dependable and Secure Computing, vol. 10, no. 4, july /august 2013.