JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

SECURE COMPUTING OF CLOUD BY HOMOMORPHIC ENCRYPTION AND MULTIPARTY COMPUTATION

[1] Kamujula VishnuvardhanReddy, [2] P.Bhargavi, M.Tech

[1]M. Tech PG Scholar, [2]Asst.Professor

[1,2]Deptartment of CSE,

[1,2] SKR College of Engineering and Technology, Konduru Satram(V), Manubolu(M), SPSR Nellore(Dt), Andhra Pradesh.

Abstract: Recent surveys reveals a powerful attacker which breaks data confidentiality by acquiring cryptographic keys, by means of coercion or backdoors in cryptographic software. Once the encryption key is exposed, the only viable measure to preserve data confidentiality is to limit the attacker's access to the cipher text. Nevertheless, if data is encrypted with existing schemes, an adversary equipped with the encryption key, can still compromise a single server and decrypt the cipher text blocks stored therein. In this paper, we study data confidentiality against an adversary which knows the encryption key and has access to a large fraction of the cipher text blocks. To this end, we propose Bastion, a novel and efficient scheme that guarantees data confidentiality even if the encryption key is leaked and the adversary has access to almost all cipher text blocks.

Keywords: Data Confidentiality, Key Exposure, Dispersed Storage.

1.INTRODUCTION

Cloud Computing offers a number of benefits and services to its customers who pay the use of hardware and software resources (servers hosted in data centres, applications, software...) on demand which they can access via internet without the need of expensive computers or a large storage system capacity and without paying any equipment maintenance fees. But these cloud providers must provide guarantees on the protection of privacy and sensitive data stored in their data centre's shared between multiple clients using the concept of virtualization.

Cloud Computing has emerged as an important paradigm that has attracted considerable attention in both industry and academia. Cloud Computing already existed under different names like "outsourcing" and "server hosting." But the poor performance of processors used, slow Internet connections and the exorbitant costs of the materials used, do not allow the use of services and storage spaces. However, recent advances in current technology (through virtualization) paved the way for these operations with faster processing. Cloud Computing security challenges and it's also an issue to many researchers; first priority was to focus on security which is the biggest concern of organizations that are considering a move to the cloud. The use of cloud computing brings a lot of advantages including reduced costs, easy maintenance and re provisioning of resources. The first real use of the concept of cloud computing was in 2002 by the company Amazon Web Services, when it leased its resources to companies during periods off celebrations (when there was no peak usage of its IT) on demand.

□ Characteristics and Services Models

□ Characteristics

The salient characteristics of cloud computing based on the definitions provided by the National Institute of Standards and Terminology (NIST) are outlined below:

- On-demand self-service: A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service's provider.
- Broad network access: Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, laptops, and PDAs).
- Resource pooling: The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location-independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or data center). Examples of resources include storage, processing, memory, network bandwidth, and virtual machines.
- Rapid elasticity: Capabilities can be rapidly and elastically provisioned, in some cases automatically, to quickly scale out and rapidly released to quickly scale in. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time.

- Measured service: Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be managed, controlled, and reported providing transparency for both the provider and consumer of the utilized service.

Services Models

Cloud Computing comprises three different service models, namely Infrastructure-as- a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS). The three service models or layer are completed by an end user layer that encapsulates the end user perspective on cloud services. The model is shown in figure below. If a cloud user accesses services on the infrastructure layer, for instance, she can run her own applications on the resources of a cloud infrastructure and remain responsible for the support, maintenance, and security of these applications herself. If she accesses a service on the application layer, these tasks are normally taken care of by the cloud service provider.

1.1 MOTIVATION

The motivation of our project is to securely computing of cloud by using homomorphic encryption techniques.

1.2 AIM

The main aim of this work is to provide a security to data stored in a cloud. Homomorphic Encryption method in cloud computing is presented in this paper as a solution to increase the security of the data. By using this method, a client can perform an operation on encrypted data without being decrypted which is the same result as the computation applied to decrypted data. Here multiple keys are generated to encrypting the uploaded files in a cloud. The project mainly done on a public cloud. Because the public cloud is globally accessible but less secure. So by using this encryption technique we can provide some more security.

1.3 OBJECTIVE

The objective of this work is to enhance the security of cloud computing. We should be able to encrypt data and send it to the cloud. After it is send, we should be able to make computation on theencrypted data and the result of this computation is an encrypted data too. If we decrypt the result of our computation, then we should get back the plain text version of the result. The question is, why we assume it enhances the cloud computing? We could just encrypt data, send it to cloud. When we need to make computation, query these data to our computers and decrypt it, make computation, then send it back to the cloud if needed.

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet). The name comes from the common use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams. Cloud computing entrusts remote services with a user's data, software and computation. Cloud computing consists of hardware and software resources made available on the Internet as managed third-party services. These services typically provide access to advanced software applications and high-end networks of server computers.

The goal of cloud computing is to apply traditional supercomputing, or high performance computing power, normally used by military and research facilities, to perform tens of trillions of computations per second, in consumer-oriented applications such as financial portfolios, to deliver personalized information, to provide data storage or to power large, immersive computer games. The cloud computing uses networks of large groups of servers typically running low-cost consumer PC technology with specialized connections to spread data processing chores across them. This shared IT infrastructure contains large pools of systems that are linked together. Often, virtualization techniques are used to maximize the power of cloud computing.

2. LITERATURE SURVEY

1 .“Secret-Sharing Schemes: A Survey,”

AUTHORS: A. Beimel,

A secret-sharing scheme is a method by which a dealer distributes shares to parties such that only authorized subsets of parties can reconstruct the secret. Secretsharing schemes are important tools in cryptography and they are used as a building box in many secure protocols, e.g., general protocol for multiparty computation, Byzantine agreement, threshold cryptography, access control, attribute-based encryption, and generalized oblivious transfer. In this survey, we will describe the most important constructions of secret-sharing schemes, explaining the connections between secret-sharing schemes and monotone formulae and monotone span programs. The main problem with known secretsharing schemes is the large share size: it is exponential in the number of parties. We conjecture that this is unavoidable. We will discuss the known lower bounds on the share size. These lower bounds are fairly weak and there is a big gap between the lower and upper bounds. For linear secret-sharing schemes, which is a class of schemes based on linear algebra that contains most known schemes, superpolynomial lower bounds on the share size are known. We will describe the proofs of these lower bounds. We will also present two results connecting secret-sharing schemes for a Hamiltonian access structure to the NP vs. co NP problem and to a major open problem in cryptography – constructing oblivious-transfer protocols from one-way functions.

2. Using Erasure Codes Efficiently for Storage in a Distributed System,”

AUTHORS: M. K. Aguilera, R. Janakiraman, and L. Xu,

Erasure codes provide space-optimal data redundancy to protect against data loss. A common use is to reliably store data in a distributed system, where erasurecoded data are kept in different nodes to tolerate node failures without losing data. In this paper, we propose a new approach to maintain ensure-encoded data in a distributed system. The approach allows the use of space efficient k-of-n erasure codes where n and k are large and the overhead n-k is small. Concurrent updates and accesses to data are highly optimized: in common cases, they require no locks, no two-phase commits, and no logs of old versions of data. We evaluate our approach using an implementation and simulations for larger systems.

3. “Security amplification by composition: The case of doublyiterated, ideal ciphers,”

AUTHORS: W. Aiello, M. Bellare, G. D. Crescenzo, and R. Venkatesan,

One concern in using cloud storage is that the sensitive data should be confidential We investigate, in the Shannon model, the security of constructions corresponding to double and (two-key) triple DES. That is, we consider Fk1 (Fk2 ()) and Fk1 (F 1 k2 (Fk1 ())) with the component functions being ideal ciphers. This models the resistance of these constructions to \generic" attacks like meet in the middle attacks. sense. We compute a bound on the probability of breaking the double cipher as a function of the number of computations of the base cipher made, and the number of examples of the composed cipher seen, and show that the success probability is the square of that for a single key cipher meet in the middle is the best possible generic attack against the double cipher. local revocable group signature and identity-based broadcast encryption with constant size cipher text and private keys. To realize our concept, we equip the broadcast encryption with the dynamic cipher text update feature, and give formal security guarantee against adaptive chosen-cipher text decryption and update attacks.

4. “The security of all-or-nothing encryption: Protecting against exhaustive key search,”

AUTHORS: A. Desai

We investigate the all-or-nothing encryption paradigm which was introduced by Rivest as a new mode of operation for block ciphers. The paradigm involves composing an all- or-nothing transform (AONT) with an ordinary encryption mode. The goal is to have secure encryption modes with the additional property that exhaustive key-search attacks on them are slowed down by a factor equal to the number of blocks in the cipher text.

We give a new notion concerned with the privacy of keys that provably captures this key-search resistance property. We suggest a new characterization of AONTs and establish that the resulting all-ornothing encryption paradigm yields secure encryption modes that also meet this notion of key privacy. A consequence of our new characterization is that we get more efficient ways of instantiating the all-or-nothing encryption paradigm. We describe a simple block-cipher-based AONT and prove it secure in the Shannon Model of a block cipher. We also give attacks against alternate paradigms that were believed to have the above key search resistance property.

5. “Deniable encryption with negligible detection probability,”

AUTHORS: M. Dürmuth and D. M. Freeman,

Deniable encryption, introduced in 1997 by Canetti, Dwork, Naor, and Ostrovsky, guarantees that the sender or the receiver of a secret message is able to “fake” the message encrypted in a specific cipher text in the presence of a coercing adversary, without the adversary detecting that he was not given the real message. To date, constructions are only known either for weakened variants with separate “honest” and “dishonest” encryption algorithms, or for single-algorithm schemes with nonnegligible detection probability. We propose the first sender-deniable public key encryption system with a single encryption algorithm and negligible detection probability. We describe a generic interactive construction based on a public key bit encryption scheme that has certain properties, and we give two examples of encryption schemes with these properties, one based on the quadratic residuosity assumption and the other on trapdoor permutations.

3.SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

To the best of our knowledge, this is the first work that addresses the problem of securing data stored in multicloud storage systems when the cryptographic material is exposed. In the following, we survey relevant related work in the areas of deniable encryption, information dispersal, all-or-nothing transformations, secret-sharing techniques, and leakage-resilient cryptography. Existing AON (all or nothing) encryption schemes, however, require at least two rounds of block cipher encryptions on the data: one preprocessing round to create the AONT, followed by another round for the actual encryption. Notice that these rounds are sequential, and cannot be parallelized. This results in considerable often un acceptable overhead to encrypt and decrypt large files. The disadvantages of existing system is,

- Security is not provided very efficiently
- This requires two rounds so that time will be consumed and mostly results are not perfect.

3.2 PROPOSED SYSTEM

Here we proposed Homomorphic Encryption And Multi-Party Computation which ensures the confidentiality of encrypted data even when the adversary has the encryption key, and all but two cipher text blocks. Homomorphic Encryption is most suitable for settings where the cipher text blocks are stored in multi-cloud storage systems. In these settings, the adversary would need to acquire the encryption key, and to compromise all servers, in order to recover any single block of plaintext. Homomorphic encryption is the conversion of data into cipher text that can be analyzed and worked with as if it were still in its original form.

Homomorphic encryptions allow complex mathematical operations to be performed on encrypted data without compromising the encryption. Now a days technology has been improved a lot. so by using technology a powerful attacker breaks data confidentiality by acquiring cryptographic keys, by means of coercion or backdoors in cryptographic software. Once the encryption key is exposed, the only viable measure to preserve data confidentiality is to limit the attacker’s access to the cipher text. By using algorithms and encryption techniques we can provide security to the data in a cloud. Homomorphic algorithm uses in the project to improve the security of adata in a cloud. When compare with previous encryption techniques like RSA, AES etc, the homomorphic encryption technique provides more security. So the over view of the project is to provide information and security in a cloud.

Homomorphic encryption allows data to be encrypted and outsourced to commercial cloud environments for research and data-sharing purposes while protecting user or patient data privacy. The advantages of proposed system is,

- Here security has improved
- Performance is also increased.

3.3 SYSTEM REQUIREMENTS

3.3.1 SOFTWARE REQUIREMENTS

- System : Pentium IV 2.4 GHz.
- Hard Disk : 40 GB
- Floppy Drive : 1.44 Mb.
- Monitor : 15 VGA Colour.
- Ram : 512 Mb.

3.3.2 HARDWARE REQUIREMENTS

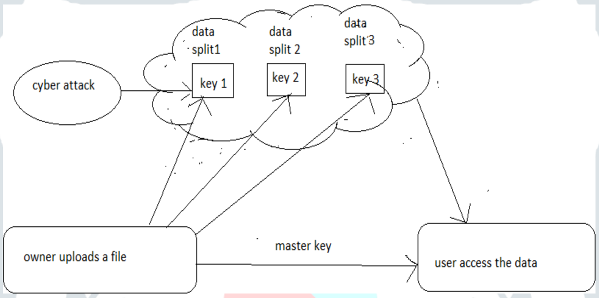
- Operating system : - Windows Family.
- Coding Language : JAVA/J2EE
- Data Base : MYSQL
  Apache Tomcat server

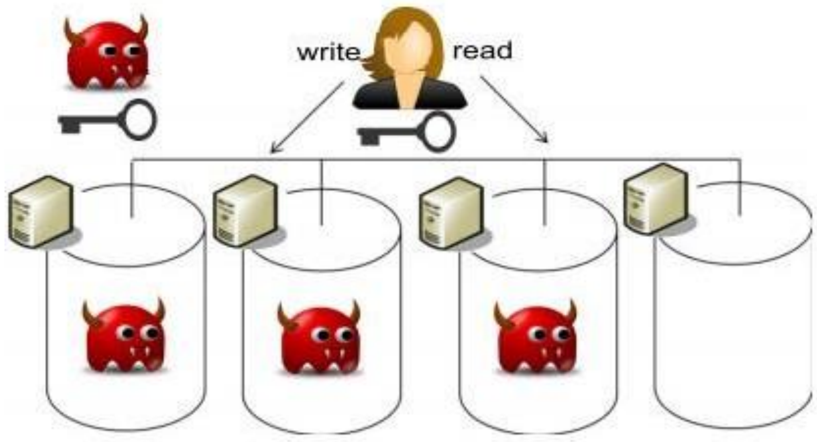
4.SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

To secure a cloud data, the data owner must encrypt the file or the document before he uploads it to accessing policies etc. Cryptography is using in this scheme to have access control over the cloud data. In this, the data is encrypted by using a special technique. The data is encrypted over the attributes with an access structure and a secret pass code is stamped on owner attributes. The user can only decrypt the file if the secret pass code linked with the attributes matches the pass code entered by the user.

In this work, the generation of keys is done by the Central authority for verified users. In this each authority is managed by the attribute individually. To improve the security of the process we also propose a mechanism which can detect the incorrectly verification process. Key assignment scheme aim to minimize the expense in storing and managing secret keys for general cryptographic use. Key assignment schemes most likely non-constant decryption key size, symmetric or public key for a predefined hierarchy is used. Only hash functions are used for a node to derive a descendant's key from its own key. The space complexity of the public information is the same as that of storing hierarchy and is asymptotically optimal; the private information at a node consists of a single key associated with that node and updates are handled locally in the hierarchy . Presented anencryption scheme which is originally proposed for concisely transmitting large numberof keys in broadcast scenario. And uses Symmetric-key encryption with Compact Key. In this paper build an efficient system that allows patients both to share partial access rights with others, and to perform searches over their records. They formalize the requirements of a Patient Controlled Encryption scheme, and give several instances, based on existing cryptographic primitive sand protocols, each achieving a different set of properties.

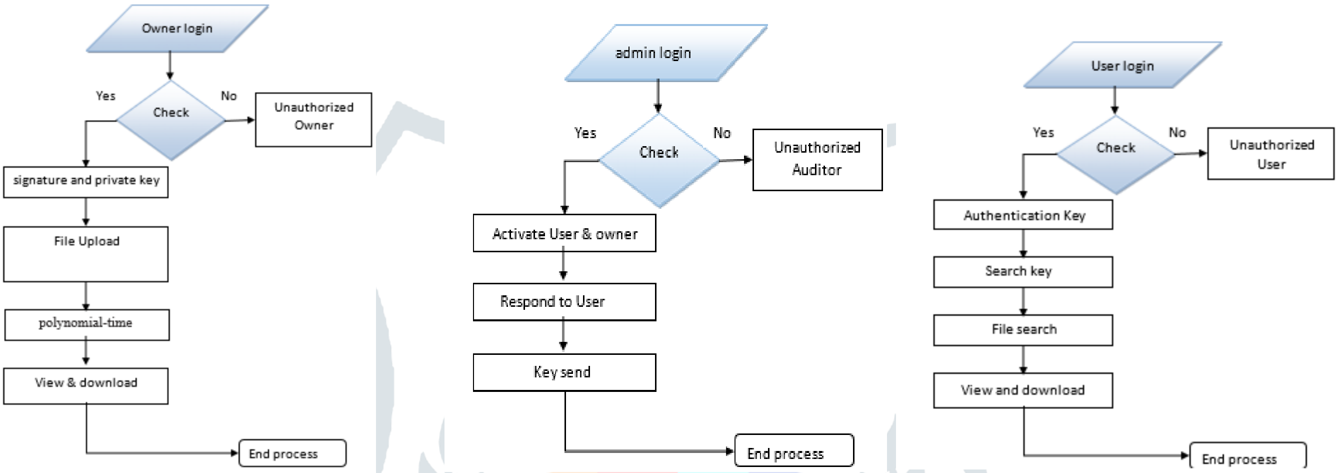




However, it is designed for the symmetric-key setting instead. The encryptor needs to get the corresponding secret keys to encrypt data, which is not suitable for many applications. Since their method is used to generate a secret value rather than a pair of public/secret keys, it is unclear how to apply this idea for public-key encryption scheme. Identity-based encryption is a type of public-key encryption in which the public-key of a user can be set as an identity-string of the user (e.g., an email address). There is a trusted party called private key generator in IBE which holds a master-secret key and issues a secret key to each user with respect to the user identity. The encryptor can take the public parameter and a user identity to encrypt a message. The recipient can decrypt this cipher text by his secret key. In this scheme, key aggregation is constrained in the sense that all keys to be aggregated must come from different "identity divisions". While there are an exponential number of identities and thus secret keys, only a polynomial number of them can be aggregated. This greatly increases the costs of storing and transmitting cipher texts, which is impractical in many situations such as shared cloud storage. Attribute-based encryption allows each cipher text to be associated with an attribute, and the master-secret key holder can extract a secret key for a policy of these attributes so that a cipher text can be decrypted by this key if its associated attribute conforms to the policy. In a multi-authority ABE scheme, multiple attribute-authorities monitor different sets of attributes and issue corresponding decryption keys to users and encryptors can require that a user obtain keys for appropriate attributes from each authority before decrypting a message.

4.2 DATAFLOW DIAGRAMS

1)The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2)The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3)DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4)DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD my be partitioned into levels that represent increasing information flow and functional detail.
5)The data flow diagrams for secure cloud computing project have data owner login dataflow diagram, data user login data flow diagram and Admin login data flow diagram.



4.3 UML DIAGRAMS

UML stands for Unified Modeling Language (UML). UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software.

In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML. The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

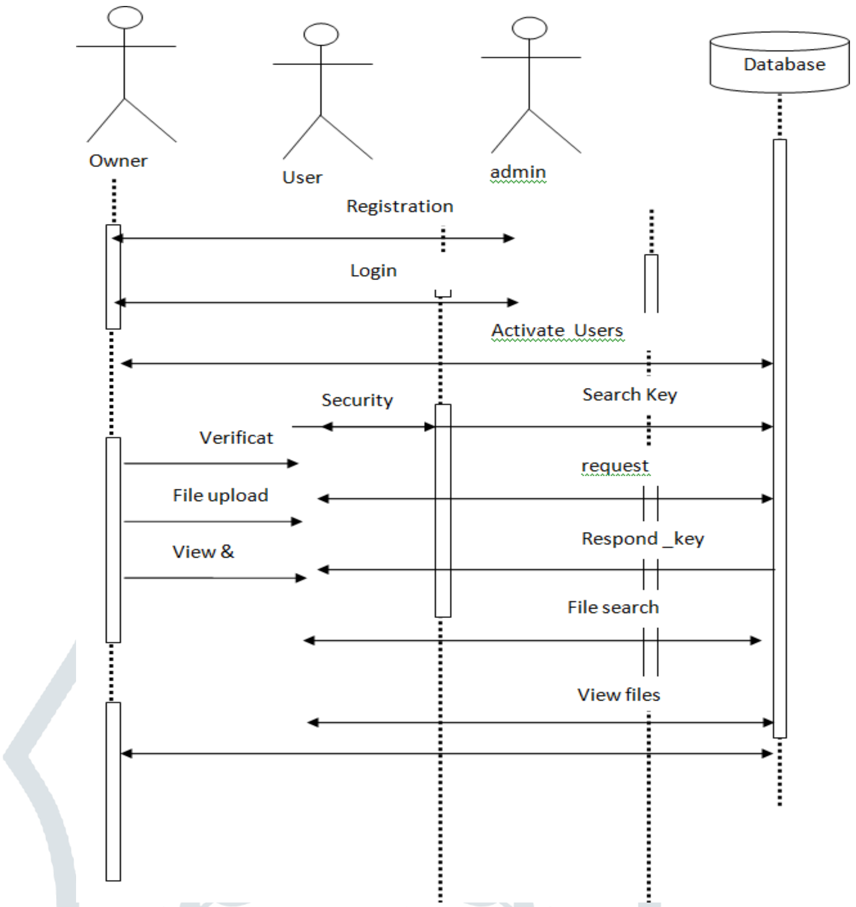
GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.



4.4 INPUT AND OUTPUT DESIGN

INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

□ What data should be given as input?

□ How the data should be arranged or coded?

□ The dialog to guide the operating personnel in providing input.

□ Methods for preparing input validations and steps to follow when error occur.

- OBJECTIVES

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will

not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow.

OUTPUT DESIGN:

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
2. Select methods for presenting information.
3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

□ Convey information about past activities, current status or projections of the

□ Future.

□ Signal important events, opportunities, problems, or warnings.

□ Trigger an action.

Confirm an action.

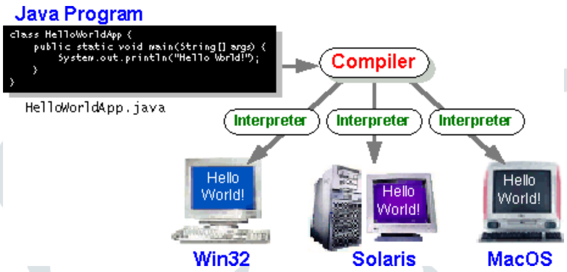
4.5 SOFTWARE ENVIRONMENT

In this project we use jdbc , java , netbeans , mysql like software environment uses.

JAVA

Java technology is both a programming language and a platform. The Java programming language is a high-level language that can be characterized by simple, architecture neutral, portable, object oriented, distributed.

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Java byte codes the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make "write once, run anywhere" possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.



```
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

□ The Java Platform

A platform is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and Mac OS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

- The Java Virtual Machine (Java VM)
- The Java Application Programming Interface (Java API)

You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as packages. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.

Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just- in-time byte code compilers can bring performance close to that of native code without threatening portability.

□ What Can Java Technology Do?

The most common types of programs written in the Java programming language are applets and applications. If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a server serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a servlet. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.

 ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a de facto standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer

to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE. There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources. From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn't change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low- level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many

of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn't as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

 JDBC

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database

access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of "plug-in" database connectivity modules, or drivers. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC's framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

 JDBC Goals

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

- SQL Level API

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher- level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to "generate" JDBC code and to hide many of JDBC's complexities from the end user.

- SQL Conformance

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed

through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

- JDBC must be implemental on top of common database interfaces

The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.

- Provide a Java interface that is consistent with the rest of the Java system

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

- Keep it simple

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

- Use strong, static typing wherever possible

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

- Keep the common cases simple

Because more often than not, the usual SQL calls used by the programmer are simple SELECT’s, INSERT’s, DELETE’s and UPDATE’s, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible.

□ MYSQL

MySQL is the most popular Open Source Relational SQL Database Management System. MySQL is one of the best RDBMS being used for developing various web- based software applications. MySQL is developed, marketed and supported by MySQL AB, which is a Swedish company. This tutorial will give you a quick start to MySQL and make you comfortable with MySQL programming. MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. MySQL is developed, marketed

and supported by MySQL AB, which is a Swedish company. MySQL is becoming so popular because of many good reasons,

- MySQL is released under an open-source license. So you have nothing to pay to use it.
- MySQL uses a standard form of the well-known SQL data language.
- MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.

□ JDBC connectivity

The JDBC provides database-independent connectivity between the J2EE platform and a wide range of tabular data sources. JDBC technology allows an Application Component Provider to:

- Perform connection and authentication to a database server
- Manager transactions
- Move SQL statements to a database engine for preprocessing and execution
- Execute stored procedures
- Inspect and modify the results from Select statements.

□ JAVA SERVER PAGE(JSP)

Java server page is a simple, yet powerful technology for creating and maintaining dynamic-content web pages. Based on the java programming language, Java Server Page offers proven portability, open standards, and a mature re-usable component model. The Java Server Page architecture enables the separation of content generation from content presentation. This separation not eases maintenance headaches, it allows web team members to focus on their areas of expertise. Now, web page designer can concentrate on layout , and web application designers on programming, with minimal concern about impacting each other’s work.

□ Steps in the execution of a JSP Application:

1. The client sends a request to the web server for a JSP file by giving the name of the JSP file within the form tag of a HTML page.

2. This request is transferred to the JavaWebServer. At the server side JavaWebServer receives the request and if it is a request for a jsp file server gives this request to the JSP engine.
3. JSP engine is program which can understands the tags of the jsp and then it converts those tags into a Servlet program and it is stored at the server side. This Servlet is loaded in the memory and then it is executed and the result is given back to the JavaWebServer and then it is transferred back to the result is given back to the JavaWebServer and then it is transferred back to the client.

□ Tomcat web server

Tomcat is an open source web server developed by Apache Group. Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed by Sun under the Java Community Process. Web Servers like Apache Tomcat support only web components while an application server supports web components as well as business components (BEAs Weblogic, is one of the popular application server).To develop a web application with jsp/servlet install any web server like JRun, Tomcat etc to run your application.

5.IMPLEMENTATION

Data Owner
Data User
Admin

MODULES DESCRIPTION:

Data Owner:
In Data Owner module, Initially Data Owner must have to register their detail and admin will approve the registration by sending signature key and private key through email. After successful login he/she have to verify their login by entering signature and private key. Then data Owner can upload files into cloud server with Polynomial key generation. He/she can view the files that are uploaded in cloud by entering the secret file key.

Data User:
In Data User module, Initially Data Users must have to register their detail and admin will approve the registration by sending signature key and private key through email. After successful login he/she have to verify their login by entering signature and private key. Data Users can search all the files upload by data owners. He/she can send search request to admin then admin will send the search key. After entering the search key he/she can view the file

Admin:
In Admin module, Admin can view all the Data owners and data user's details. Admin will approve the users and send the signature key and private key to the data owners and data users. Also admin will send the search request key to the users. Admin can able see the files in cloud uploaded by the data owners.

6. SYSTEM TESTING

6.1 TYPES OF TESTING

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.

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.

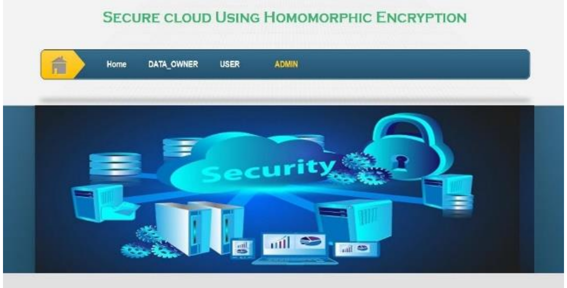
Black Box Testing

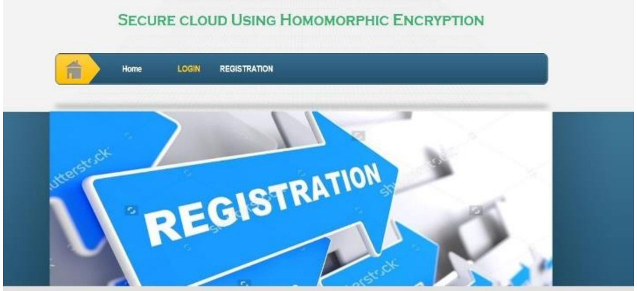
Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification 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.

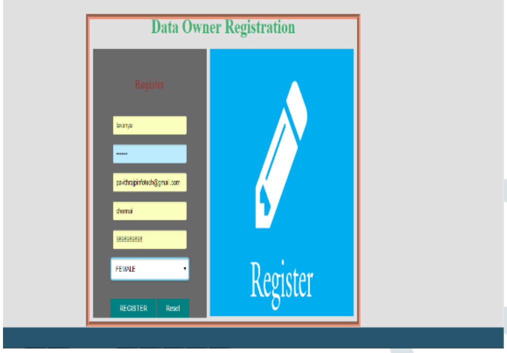
7.EXPERIMENTAL RESULTS

7.1EXECUTON SCREEN SHOTS

HOME PAGEDATA OWNER REGISTRATION PAGE






Data Owner Registration

Register

lavanya

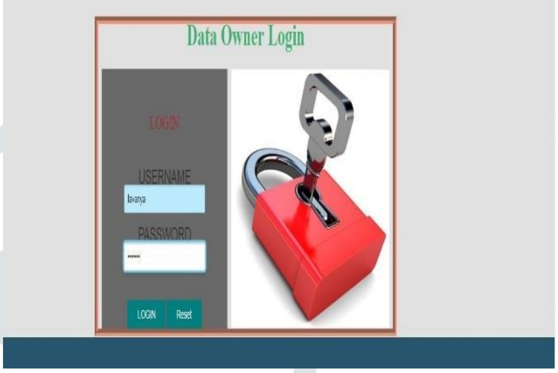
******

pavithrajpinfotech@gmail.com

chennai

9898989898

FEMALE

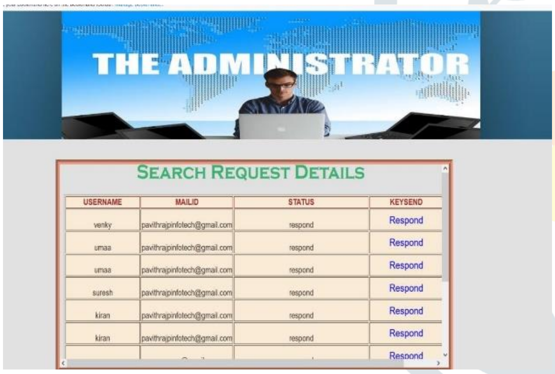

Data Owner Login

LOGIN

USERNAME lavanya

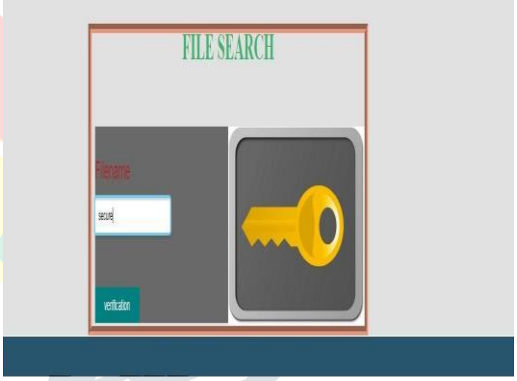
PASSWORD ******

SEARCH REQUEST DETAILS FILE SEARCH



SEARCH REQUEST DETAILS

USERNAME	MAILID	STATUS	KEYSEND
venky	pavithrajpinfotech@gmail.com	respond	Respond
umaa	pavithrajpinfotech@gmail.com	respond	Respond
umaa	pavithrajpinfotech@gmail.com	respond	Respond
suresh	pavithrajpinfotech@gmail.com	respond	Respond
kiran	pavithrajpinfotech@gmail.com	respond	Respond
kiran	pavithrajpinfotech@gmail.com	respond	Respond
			Respond


FILE SEARCH

Filename secure

8.CONCLUSION

In this paper, we addressed the problem of securing data outsourced to the cloud againstan adversary which has access to the encryption key. For that purpose, we introduced a novel security definition that captures data confidentiality against the new adversary. We then proposed Bastion, a scheme which ensures the confidentiality of encrypted data even when the adversary has the encryption key, and all but two cipher text blocks. Bastion is most suitable for settings where the cipher text blocks are stored in multi- cloud storage systems. In these settings, the adversary would need to acquire the encryption key, and to compromise all servers, in order to recover any single block of plaintext. We analyzed the security of Bastion and evaluate edits performance in realistic settings. Bastion considerably improves (by more than 50%) the performanceof existing primitives which offer comparable security under key exposure, and only incurs a negligible overhead (less than 5%) when compared to existing semantically secure encryption modes (e.g., the CTR encryption mode). Finally, we showed how Bastion can be practically integrated within existing dispersed storage systems.

9.REFERENCES

[1]M. Abd-El-Malek, G. R. Ganger, G. R. Goodson, M. K. Reiter, and J. J. Wylie, "Fault-Scalable Byzantine Fault-Tolerant Services," in ACM Symposium on Operating Systems Principles (SOSP), 2005, pp. 59–74.

[2]M. K. Aguilera, R. Janakiraman, and L. Xu, "Using Erasure Codes Efficiently for Storage in a Distributed System," in International Conference on Dependable Systems and Networks (DSN), 2005, pp. 336–345.

[3]W. Aiello, M. Bellare, G. D. Crescenzo, and R. Venkatesan, "Security amplification by composition: The case of doublyiterated, ideal ciphers," in Advances in Cryptology (CRYPTO), 1998, pp. 390–407.

[4]C. Basescu, C. Cachin, I. Eyal, R. Haas, and M. Vukolic, "Robust Data Sharing with Key-Value Stores," in ACM SIGACTSIGOPS Symposium on Principles of Distributed Computing (PODC), 2011, pp. 221–222.

[5]A. Beimel, "Secret-sharing schemes: A survey," in International Workshop on Coding and Cryptology (IWCC), 2011, pp. 11–46.

[6]A. Bessani, M. Correia, B. Quaresma, F. André, and P. Sousa, "DepSky: Dependable and Secure Storage in a Cloud-ofclouds," in Sixth Conference on Computer Systems (EuroSys), 2011, pp. 31–46.

[7]G. R. Blakley and C. Meadows, "Security of ramp schemes," in Advances in Cryptology (CRYPTO), 1984, pp. 242–268.

[8]V. Boyko, "On the Security Properties of OAEP as an Allor-nothing Transform," in Advances in Cryptology (CRYPTO), 1999, pp. 503–518.

[9]R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky, "Deniable Encryption," in Proceedings of CRYPTO, 1997.

[10]Cavalry, "Encryption Engine Dongle," http://www. cavalrystorage.com/en2010.aspx/.

[11]C. Charnes, J. Pieprzyk, and R. Safavi-Naini, "Conditionally secure secret sharing schemes with disenrollment capability," in ACM Conference on Computer and Communications Security (CCS), 1994, pp. 89–95.

[12]A. Desai, "The security of all-or-nothing encryption: Protecting against exhaustive key search," in Advances in Cryptology (CRYPTO), 2000, pp. 359–375.

[13]C. Dubnicki, L. Gryz, L. Heldt, M. Kaczmarczyk, W. Kilian, P. Strzelczak, J. Szczepkowski, C. Ungureanu, and M. Welnicki, "HYDRAstor: a Scalable Secondary Storage," in USENIX Conference on File and Storage Technologies (FAST), 2009, pp. 197–210.

[14]M. Dürmuth and D. M. Freeman, "Deniable encryption with negligible detection probability: An interactive construction," in EUROCRYPT, 2011, pp. 610–626.

[15]EMC, "Transform to a Hybrid Cloud," http://www.emc. com/campaign/global/hybridcloud/index.htm.

[16]IBM, "IBM Hybrid Cloud Solution," http://www-01.ibm. com/software/tivoli/products/hybrid-cloud/.

[18]Penchalaiah P, Rajasekar P et al, "An Efficient Multi-User Hierarchical Authenticated Encryption Using Simultaneous Congruence for Highly Secure Data", International Journal of Future Generation Communication and Networking (WoS), ISSN: 2233-7857(Print); 2207-9645(Online), NADIA, (2020), Vol. 13, No. 2, pp. 1-10.

[19]Penchalaiah P, Ramesh Reddy K, "Random Multiple Key Streams for Encryption with Added CBC Mode of Operation", Perspectives in Science (ISSN: 2213-0209), (ELSEVIER, UGC Journal no-62532), Volume 8, pp.57-62, April 2016.

[20]Penchalaiah P, M Vijay Kumar et al, "A Research Threshold Efficient Hybrid Encryption Schema for Secure File System", International Journal of Recent Technology and Engineering (IJRTE),ISSN: 2277-3878, (SCOPUS) Volume-8, Issue- 2S3, Page 888 – 891,July 201-8, Issue- 2S3, Page 888 – 891,July 2019.