A Project Report On

TRANSFORMATION OF ENCRYPTION WITH IDENTITY BASED APPROACH FOR VERSATILE ENCRYPTED DATA SHARING IN PUBLIC CLOUD

SUBMITTED TO OSMANIA UNIVERSITY FOR THE PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

MASTER OF COMPUTER APPLICATIONS

BY

DURGAM VINOD

(1305-22-862-064)

UNDER THE GUIDANCE OF

MS.ANNAPURNA

Assistant Professor

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

AVANTHI PG COLLEGE

(AFFILIATED TO OSMANIA UNIVERSITY & RECOGNIZED BY AICTE)

MOOSARAMBAGH, DILSUKHNAGAR, HYDERABAD-500036

2023-2024

MASTER OF COMPUTERAPPLICATIONS

AVANTHI PG COLLEGE

(AFFILIATED TO OSMANIA UNIVERSITY &RECOGNIZED BY AICTE)

MOOSARAMBAGH, DILSHUKNAGAR, HYDERABAD-500036.

CERTIFICATE

THIS IS TO CERTIFY THAT THE PROJECT **TRANSFORMATION OF ENCRYPTION WITH IDENTITY BASED APPROACH FOR VERSATILE ENCRYPTED DATA SHARING IN PUBLIC CLOUD** IS A BONAFIDE WORK CARRIED OUT BY **DURGAM VINOD (130522862064)** IN PARTIAL FULFILLMENT OF THEREQUIREMENTS FOR THE AWARD OF DEGREE OF **MASTER OF COMPUTER APPLICATIONS** FROM AVANTHI PG COLLEGE, AFFILIATED TO OSMANIA UNIVERSITY, HYDERABAD, UNDER GUIDANCE AND SUPERVISION.

INTERNAL GUIDE

MS.ANNAPURNA

ASST. PROFESSOR,
AVANTHI PG COLLEGE.

HEAD OF THE DEPARTMENT

MR. P. NAVEEN CHANDRA

ASST. PROFESSOR,
AVANTHI PG COLLEGE.

EXTERNAL EXAMINER

Ph: 040-6641 4450
2404 4725
Fax: 040-2405 2117

ESTD 1992

AVANTHI P.G. COLLEGE

(Approved by AICTE, Affiliated to Osmania University & Recog. by Govt. of T.S.)

16-11-741/B/1/A, Moosarambagh, Dilsukhnagar, Hyderabad - 500 036.

Date:_____

CERTIFICATE

This is to certify that the project work **TRANSFORMATION OF ENCRYPTION WITH IDENTITY BASED APPROACH FOR VERSATILE ENCRYPTED DATA SHARING IN PUBLIC CLOUD**, a student of MCA II year bearing Hall Ticket No. **1305-22-862-064** in partial fulfillment to award then degree of MASTER OF COMPUTER APPLICATIONS in Osmania University for the academic year 2022 - 2024.

Project Guide | Head of Dept | External examiner

MANAC Infotech (P) Limited

Ref: MANAC/PROJ/23-24/D2009

JULY 03, 2024

PROJECT COMPLETION CERTIFICATE

This is to certify that **Mr.DURGAM VINOD (1305-22-862-064)** has successfully completed the Project on **TRANSFORMATION OF ENCRYPTION WITH IDENTITY BASED APPROACH FOR VERSATILE ENCRYPTED DATA SHARING IN PUBLIC CLOUD** in Partial fulfillment of the **MCA (Master of Computer Applications)** course during the period **APRIL 01, 2024 TO JULY 31, 2024.**

SOFTWARE: JAVA

We wish him/her good luck in all future endeavors.

Project Manager

ACKNOWLEDGEMENT

I would like to express my sincere gratitude and indebtedness to my project guide **MS.ANNAPURNA** for his valuable suggestions and interest throughout the course of this project.

I am also thankful to our principal **Dr.S.Apparao** and **Mr.P.Naveen Chandra**, Head of Department, Master of Computer Applications, Avanthi P.G College, Hyderabad for providing excellent infrastructure and a nice atmosphere for completing this project successfully as a part of our course.

My sincere thanks to my project coordinator **MS.ANNAPURNA** for given valuable suggestions to fulfilment of this project and I convey my thanks to the lab staff for allowing me to use the required equipment whenever needed.

I sincerely acknowledge and thank all those who have directly / in-directly given their support in completion of this Project work.

DURGAM VINOD

1305-22-862-064

DECLARATION

This is certify that the project report on **TRANSFORMATION OF ENCRYPTION WITH IDENTITY BASED APPROACH FOR VERSATILE ENCRYPTED DATA SHARING IN PUBLIC CLOUD** is a record of bonafide work done by me in the department of Master of Computer Applications , Avanthi P.G College, Osmania University. The reports are based on the project work done entirely by me and not copied from any other source.

The results embodied in this project report have not been submitted to any other institute or institute For the award of any degree or diploma to the best of my knowledge and belief.

DURGAM VINOD

1305-22-862-064

INDEX

LIST OF FIGURES

TRANSFORMATION OF ENCRYPTION WITH IDENTITY BASED APPROACH FOR VERSATILE ENCRYPTED DATA SHARING IN PUBLIC CLOUD

ABSTRACT

With the rapid development of cloud computing, an increasing number of individuals and organizations are sharing data in the public cloud. To protect the privacy of data stored in the cloud, a data owner usually encrypts his data in such a way that certain designated data users can decrypt the data. This raises a serious problem when the encrypted data needs to be shared to more people beyond those initially designated by the data owner. To address this problem, we introduce and formalize an identity-based encryption transformation (IBET) model by seamlessly integrating two well-established encryption mechanisms, namely identity-based encryption (IBE) and identity-based broadcast encryption (IBBE). In IBET, data users are identified and authorized for data access based on their recognizable identities, which avoids complicated certificate management in usual secure distributed systems. More importantly, IBET provides a transformation mechanism that converts an IBE ciphertext into an IBBE ciphertext so that a new group of users not specified during the IBE encryption can access the underlying data. We design a concrete IBET scheme based on bilinear groups and prove its security against powerful attacks. Thorough theoretical and experimental analyses demonstrate the high efficiency and practicability of the proposed scheme.

CHAPTER-1
INTRODUCTION

Cloud computing provides powerful and flexible storage services for individuals and organizations [1]. It brings about lots of benefits of sharing data with geographically dispersed data users, and significantly reduces local burden of storage management and maintenance. However, the concerns on data security and privacy are becoming one of the major obstacles impeding more widespread usage of cloud storage [2], since data owners lose physical control on their data after data are outsourced to cloud servers maintained by a cloud services provider (CSP). Data owners may worry about whether their sensitive data have been accessed by unauthorized users or malicious CSP. Cryptographic encryptions are widely suggested as standard approaches to protect the security and privacy of data outsourced to clouds [3]. With encryption mechanisms, data owners first encrypt their data and then outsource to cloud servers. Then the data in clouds are stored in ciphertext format and can only be accessed by the users having matching decryption keys. In a public cloud storage system, where different data owners may employ different encryption mechanisms according to their own data sharing requirements, it is often that a data owner wants to share his data with only one user and thus encrypts the data to generate a particular ciphertext that can only be decrypted by the specific user. However, as data sharing requirement changes, the same data owner would like to share his data with more users, which, therefore, requires to transform the ciphertext format so that multiple users can decrypt. There are many scenarios in which the ciphertext transformation mentioned above is highly desirable. Consider a group of medical insurance agents draft a health insurance plan for a client. To do so, each agent needs to collect the client's personal information (e.g., electronic health records, occupations data, financial reports) from various data sources such as hospitals, employers, tax departments. The required data may be stored in remote cloud servers and especially, may be encrypted under different encryption mechanisms. To allow the agents to read and make use of the required data, a naive way is to let each agent acquire the corresponding decryption keys from the authorities who manage respective data. However, this would pose great concerns on data privacy. The authorities would ask a natural question: "If I give my decryption key to the agents, how to assure that all the agents would not leak the decryption key or use the decryption key to access other clients' stored data?" This paper attempts to solve such problem technically so that the authorities can transform the ciphertexts from one encryption system to another, without handing over their decryption keys. In particular, we consider an encryption

transformation mechanism that connects two types of well established encryption systems, i.e., identity-based encryption (IBE) and identity-based broadcast encryption (IBBE). We take electronic health records sharing as a motivation of our work. Suppose a patient is equipped with implantable or wearable medical sensors to collect personal physiological records. These records are aggregated at a mobile device and then uploaded to a remote server. To protect personal privacy, the patient may encrypt his health records by some encryption mechanism, e.g., IBE, so that only his doctor can read the health records and then make proper diagnosis. At some point, the doctor finds a complicated situation about the patient's health and consequently, decides to consult a group of experts from different hospitals. For full understanding of the patient's health condition, the experts first need to read the health records (see Fig. 1). Since the records are encrypted previously, the experts are impossible to directly read the data. Meanwhile, the encryption method taken by the patient and the corresponding decryption key are unknown to the experts. This results in a dilemma for the experts: "How could we read the patient's health records in order to provide our treatment advices?" A trivial solution would be that the doctor first decrypts all the encrypted records and then sends out the data in plaintext (not encrypted) format to each expert. This, however, may be impractical for the doctor since a considerable computation and communication costs may be caused due to the massive health data uploaded every day. More importantly, there is a risk of privacy disclosure by sending data in plaintext format. There exists a cryptographic tool called proxy re-encryption (PRE) that would be of help here. PRE can transform the doctor's ciphertext into a ciphertext that can be decrypted by one expert. Then, for n experts, PRE needs to run n times repeatedly for transferring the patient's health data to all experts, which is inefficient. We observe that IBBE achieves a useful encryption mechanism that allows multiple users to simultaneously decrypt a ciphertext. Thus, we ask: "Can we find an efficient way to transform the encrypted data in IBE ciphertext format into an IBBE ciphertext so that multiple users can decrypt at the same time?" 1.1 Our contributions In this paper, we try to answer the above question by studying encryption transformation between two different encryption systems. For the first time, we propose a novel notion called identitybased encryption transformation (IBET). We also define the notion (including algorithm definition and security model) of IBET. Then we design a concrete IBET scheme in bilinear groups, which provides the following attractive features. • Identity-based data storage. Data owner can securely outsource their data to a remote cloud server which is not fully trusted. The data are encrypted and stored in the server in IBE/IBBE ciphertext format so that only the users

authorized by the data owners can access them. All users, including data owners and data consumers, are recognized with their unique identities, which avoids the usage of complicated public-key certificates.

- Cross-domain encryption transformation. Our IBET scheme achieves a cross-domain encryption transformation which can be viewed as a bridge connecting IBE and IBBE. In particular, a data owner (or an authorized data consumer) can transform the data stored in IBE ciphertext format into the data in IBBE ciphertext format, so that a set of users specified by the data owner (or the authorized data consumer) can simultaneously access the data.

- Strong security guarantee. Our IBET scheme achieves a strong security in the sense that: 1) it can deter any unauthorized access to the data stored in the cloud server; 2) it can prevent leakage of some private information (e.g., private key) about the one who authorizes to transform encrypted data; 3) the transformation would not reveal any useful information about the sensitive data. We also conduct a series of experiments on our IBET scheme and make comparisons with some related schemes. The results show that the IBET scheme achieves a high performance in transforming the encrypted data, without incurring any significant computation costs to cloud clients or cloud servers.

Applications. Our IBET scheme can be applied to many real-world data sharing applications. First of all, the example of health records sharing described previously is an appropriate area where our IBET can be applied. Cloud-based encrypted email forwarding is another possible application. Imagine that several companies deploy their email systems on cloud servers. IBET can be used as a gateway to transform an encrypted email destined to an employee in one company into an encrypted email what can be received and decrypted by multiple employees in different companies. Vehicular ad-hoc network is also a potential application for IBET. When a car receives an encrypted report about front car condition or accident ahead and would like further to broadcast the situation to rear vehicles, IBET can be used to directly transform the encrypted report into a broadcast ciphertext that allows multiple receivers to decrypt. Last but not least, in a mobile office environment, IBET may be utilized as a mobile application to securely share business data with a company director via a public cloud, and then transform the encrypted business data (if requested) so that the whole management team can access.

CHAPTER-2

LITERATURE SURVEY

TITLE: " Enabling cloud storage auditing with verifiable outsourcing of key updates "

AUTHORS: J. Yu, K. Ren, and C. Wang

ABSTRACT: Key-exposure resistance has always been an important issue for in-depth cyber defence in many security applications. Recently, how to deal with the key exposure problem in the settings of cloud storage auditing has been proposed and studied. To address the challenge, existing solutions all require the client to update his secret keys in every time period, which may inevitably bring in new local burdens to the client, especially those with limited computation resources, such as mobile phones. In this paper, we focus on how to make the key updates as transparent as possible for the client and propose a new paradigm called cloud storage auditing with verifiable outsourcing of key updates. In this paradigm, key updates can be safely outsourced to some authorized party, and thus the key-update burden on the client will be kept minimal. In particular, we leverage the third-party auditor (TPA) in many existing public auditing designs, let it play the role of authorized party in our case, and make it in charge of both the storage auditing and the secure key updates for key-exposure resistance. In our design, TPA only needs to hold an encrypted version of the client's secret key while doing all these burdensome tasks on behalf of the client. The client only needs to download the encrypted secret key from the TPA when uploading new files to cloud. Besides, our design also equips the client with capability to further verify the validity of the encrypted secret keys provided by the TPA. All these salient features are carefully designed to make the whole auditing procedure with key exposure resistance as transparent as possible for the client. We formalize the definition and the security model of this paradigm. The security proof and the performance simulation show that our detailed design instantiations are secure and efficient.

TITLE: "Achieving secure, universal, and fine-grained query results verification for secure search scheme over encrypted cloud data"

AUTHORS: H. Yin, Z. Qin, J. Zhang, L. Ou, and K. Li,

ABSTRACT: Secure search techniques over encrypted cloud data allow an authorized user to query data files of interest by submitting encrypted query keywords to the cloud server in a privacy-preserving manner. However, in practice, the returned query results may be incorrect or incomplete in the dishonest cloud environment. For example, the cloud server may intentionally omit some qualified results to save computational resources and communication overhead. Thus, a well-functioning secure query system should provide a query results verification mechanism that allows the data user to verify results. In this paper, we design a secure, easily integrated, and fine-grained query results verification mechanism, by which, given an encrypted query results set, the query user not only can verify the correctness of each data file in the set but also can further check how many or which qualified data files are not returned if the set is incomplete before decryption. The verification scheme is loose-coupling to concrete secure search techniques and can be very easily integrated into any secure query scheme. We achieve the goal by constructing secure verification object for encrypted cloud data. Furthermore, a short signature technique with extremely small storage cost is proposed to guarantee the authenticity of verification object and a verification object request technique is presented to allow the query user to securely obtain the desired verification object. Performance evaluation shows that the proposed schemes are practical and efficient

TITLE: "Security analysis on one-tomany order preserving encryption-based cloud data search"

AUTHORS: K. Li, W. Zhang, C. Yang, and N. Yu

ABSTRACT: For ranked search in encrypted cloud data, order preserving encryption (OPE) is an efficient tool to encrypt relevance scores of the inverted index. When using deterministic OPE, the ciphertexts will reveal the distribution of relevance scores. Therefore, Wang et al. proposed a probabilistic OPE, called one-to-many OPE, for applications of searchable encryption, which can flatten the distribution of the plaintexts. In this paper, we proposed a differential attack on one-to-many OPE by exploiting the

differences of the ordered ciphertexts. The experimental results show that the cloud server can get a good estimate of the distribution of relevance scores by a differential attack. Furthermore, when having some background information on the outsourced documents, the cloud server can accurately infer the encrypted keywords using the estimated distributions

TITLE: Identity-based broadcast encryption with constant size ciphertexts and private keys

AUTHORS: C. Delerable´e

ABSTRACT: This paper describes the first identity-based broadcast en- cryption scheme (IBBE) with constant size ciphertexts and private keys. In our scheme, the public key is of size linear in the maximal size m of the set of receivers, which is smaller than the number of possible users (identities) in the system. Compared with a recent broadcast encryption system introduced by Boneh, Gentry and Waters (BGW), our system has comparable properties, but with a better eciency: the public key is shorter than in BGW. Moreover, the total number of possible users in the system does not have to be fixed in the setup.

TITLE: "Secure data sharing in cloud computing using revocable-storage identity-based encryption"

AUTHORS: J. Wei, W. Liu, and X. Hu

ABSTRACT: Cloud computing provides a flexible and convenient way for data sharing, which brings various benefits for both the society and individuals. But there exists a natural resistance for users to directly outsource the shared data to the cloud server since the data often contain valuable information. Thus, it is necessary to place cryptographically enhanced access control on the shared data. Identity-based encryption is a promising cryptographical primitive to build a practical data sharing system. However, access control is not static. That is, when some user's authorization is expired, there should be a mechanism that can remove him/her from the system. Consequently, the revoked user cannot access both the previously and subsequently shared data. To this end, we propose a notion called revocable-storage identitybased encryption (RS-IBE), which can provide the forward/backward security of ciphertext by introducing the functionalities of user revocation and ciphertext update simultaneously. Furthermore, we present a concrete construction of

RS-IBE, and prove its security in the defined security model. The performance comparisons indicate that the proposed RS-IBE scheme has advantages in terms of functionality and efficiency, and thus is feasible for a practical and cost-effective data sharing system. Finally, we provide implementation results of the proposed scheme to demonstrate its practicability

CHAPTER-3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

There are many scenarios in which the ciphertext transformation mentioned above is highly desirable. Consider a group of medical insurance agents draft a health insurance plan for a client. To do so, each agent needs to collect the client's personal information (e.g., electronic health records, occupations data, financial reports) from various data sources such as hospitals, employers, tax departments. The required data may be stored in remote cloud servers and especially, may be encrypted under different encryption mechanisms. To allow the agents to read and make use of the required data, a naive way is to let each agent acquire the corresponding decryption keys from the authorities who manage respective data. However, this would pose great concerns on data privacy.

DISADVANTAGES

- The data owner has to rebuild the search index tree, which is time-consuming.
- Traditional solutions have to suffer high computational costs.

3.2 PROPOSED SYSTEM

we try to answer the above question by studying encryption transformation between two different encryption systems. For the first time, we propose a novel notion called identity-based encryption transformation (IBET). We also define the notion (including algorithm definition and security model) of IBET. Then we design a concrete IBET scheme in bilinear groups, which provides the following attractive features.

- Identity-based data storage. Data owner can securely outsource their data to a remote cloud server which is not fully trusted. The data are encrypted and stored in the server in IBE/IBBE ciphertext format so that only the users authorized by the data owners can access them. All users, including data owners and data consumers, are recognized with their unique identities, which avoids the usage of complicated public-key certificates.

- Cross-domain encryption transformation. Our IBET scheme achieves a cross-domain encryption transformation which can be viewed as a bridge connecting IBE and IBBE. In particular, a data owner (or an authorized data consumer) can transform the data stored in IBE ciphertext format into the data in IBBE ciphertext format, so that a set of users specified by the data owner (or the authorized data consumer) can simultaneously access the data.

• Strong security guarantee. Our IBET scheme achieves a strong security in the sense that: 1) it can deter any unauthorized access to the data stored in the cloud server; 2) it can prevent leakage of some private information (e.g., private key) about the one who authorizes to transform encrypted data; 3) the transformation would not reveal any useful information about the sensitive data.

We also conduct a series of experiments on our IBET scheme and make comparisons with some related schemes. The results show that the IBET scheme achieves a high performance in transforming the encrypted data, without incurring any significant computation costs to cloud clients or cloud servers.

ADVANTAGES

- Data security protection: If data have been encrypted before outsourced, then only the clients holding correct decryption keys can access (these clients are also called authorized clients). The encrypted data are unreadable to CSP or unauthorized clients (those having no correct decryption keys).
- Controllable transformation: Only the files specified by the data owner in the authorization token can be transformed by CSP. CSP and other clients cannot cooperatively deduce a valid authorization token in order to transform unspecified files, nor detect sensitive information about the data encrypted in unspecified files.

CHAPTER-4
SYSTEM REQUIREMENTS

4.1 FUNCTIONAL REQUIREMENTS

- Data Owner
- Cloud Service provider
- Registry Authority

4.2 NON FUNCTIONAL REQUIREMENTS

4.2.1 HARD REQUIREMENTS

- Processor : 1GHz or faster CPU or System on a Chip with two or more cores.
- RAM : 4GB.
- Hard drive : 64GB or larger.
- System firmware : UEFI, Secure Boot capable.
- TPM : Trusted Platform Module (TPM) version 2.0.

4.2.2 SOFTWARE REQUIREMENTS

- Operating system : Windows 10.
- Coding Language : JAVA, HTML, CSS
- Data Base : MY SQL

CHAPTER-5

SYSTEM STUDY

5.1 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

5.2 FEASIBILITY ANALYSIS

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER-6

SYSTEM DESIGN

6.1 SYSTEM ARCHITECTURE

6.2 UML DIAGRAMS

UML stands for Unified Modeling Language. 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

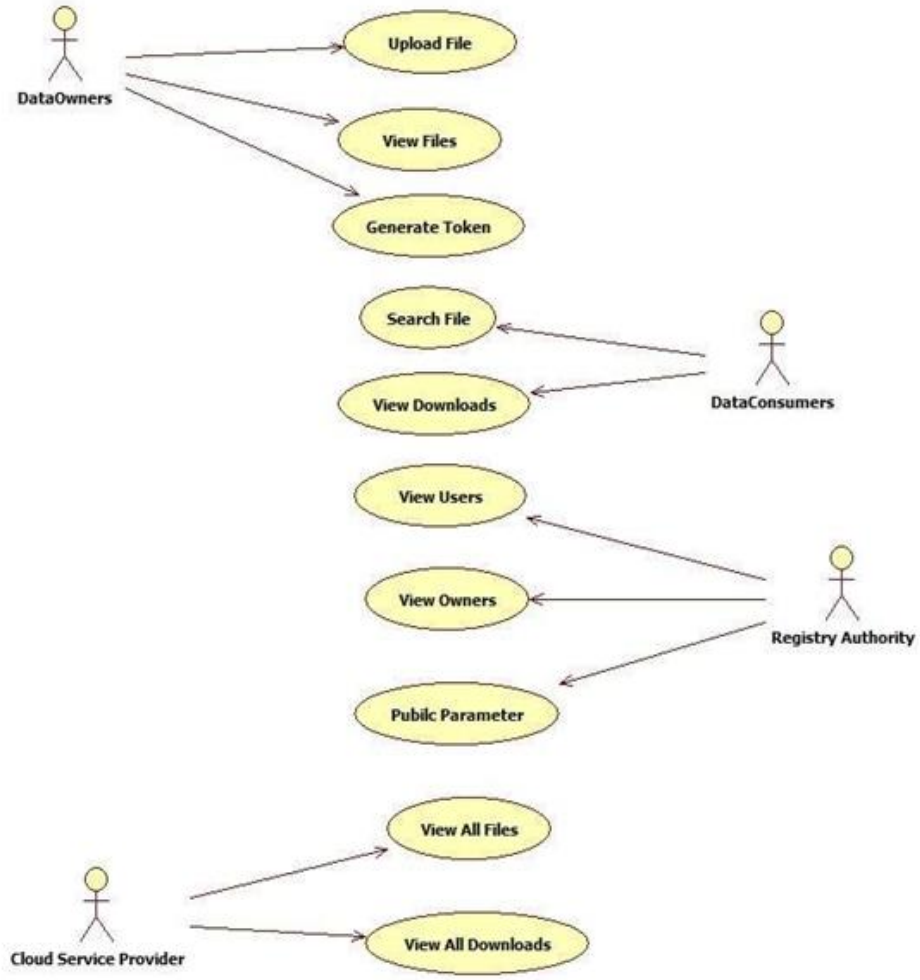
The Primary goals in the design of the UML are as follows:

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.

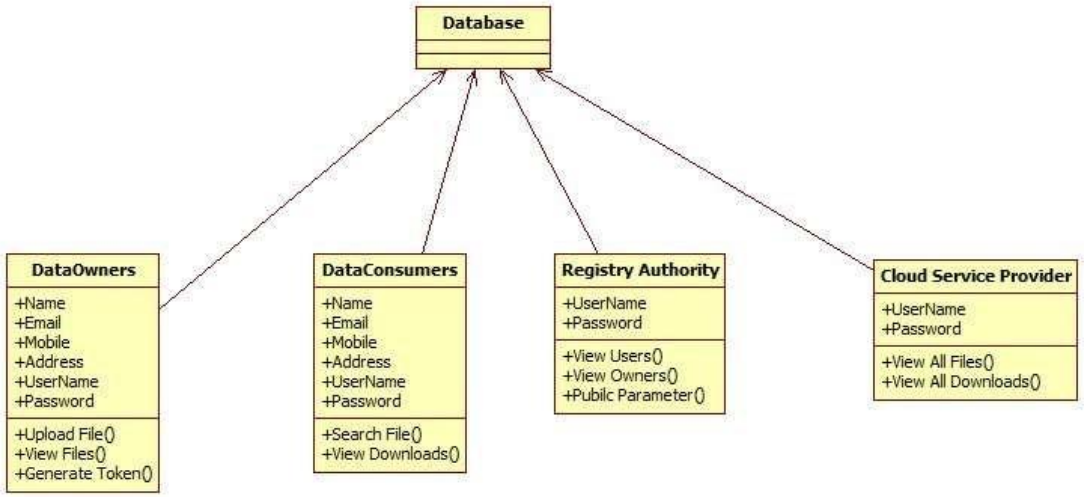
6.2.1 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



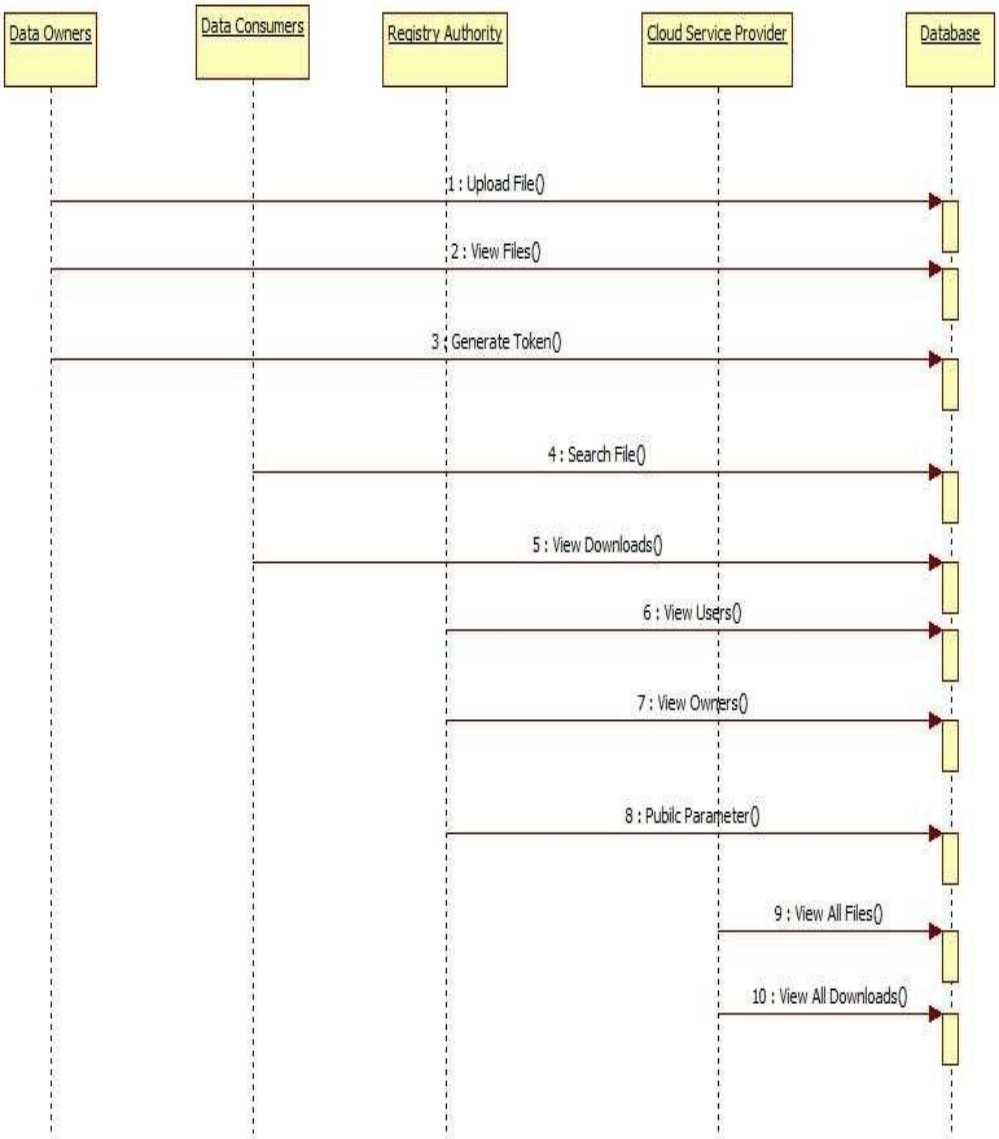
6.2.2 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



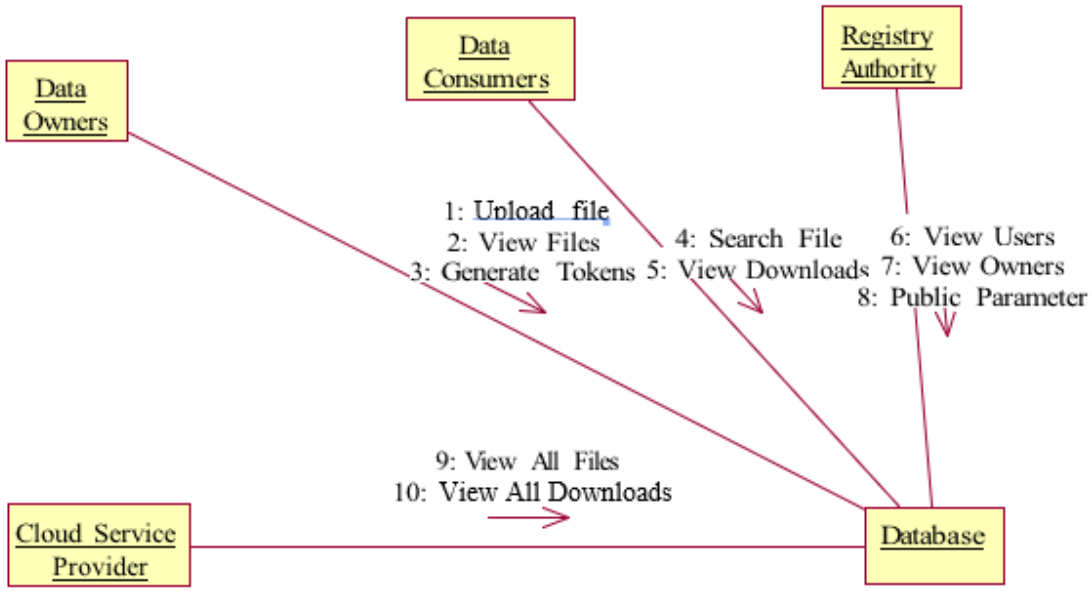
6.2.3 SEQUENCE DIAGRAM

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.

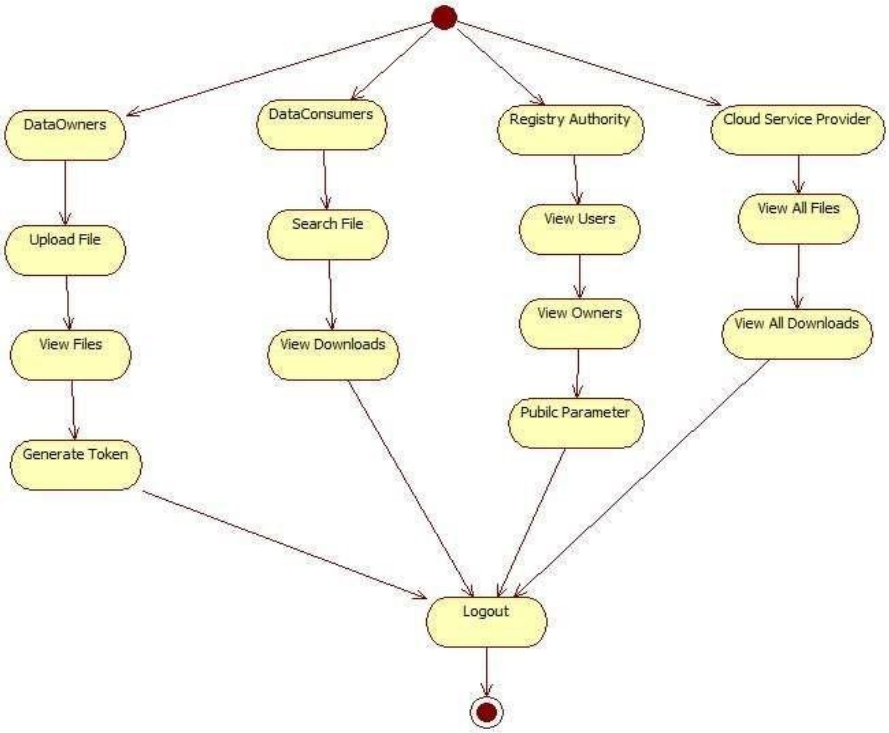

Data Owners
Data Consumers
Registry Authority
Cloud Service Provider
Database
1 : Upload File()
2 : View Files()
3 : Generate Token()
4 : Search File()
5 : View Downloads()
6 : View Users()
7 : View Owners()
8 : Pubilc Parameter()
9 : View All Files()
10 : View All Downloads()

6.2.4 COLLABRATION DIAGRAM

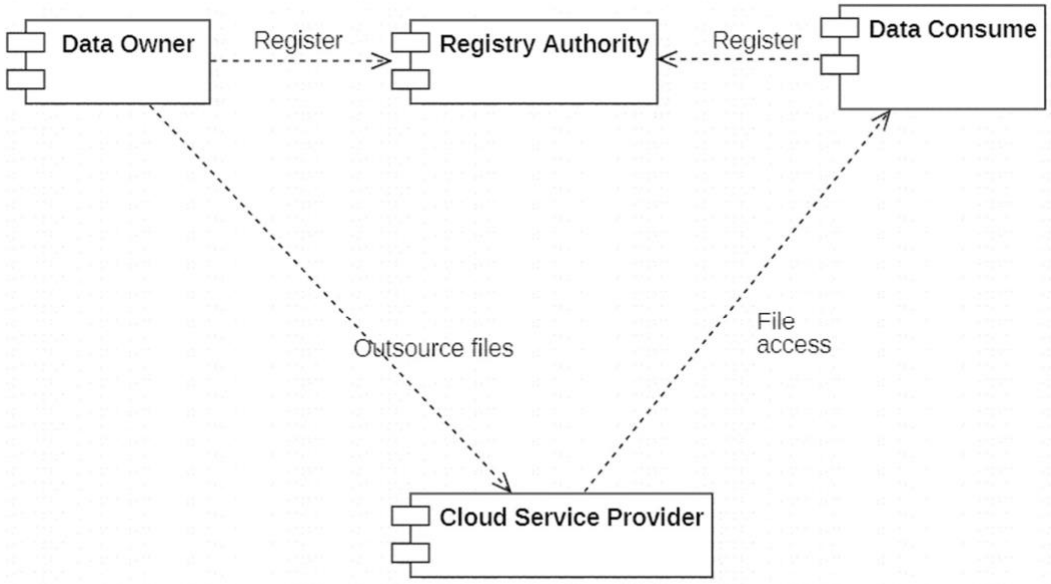
Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



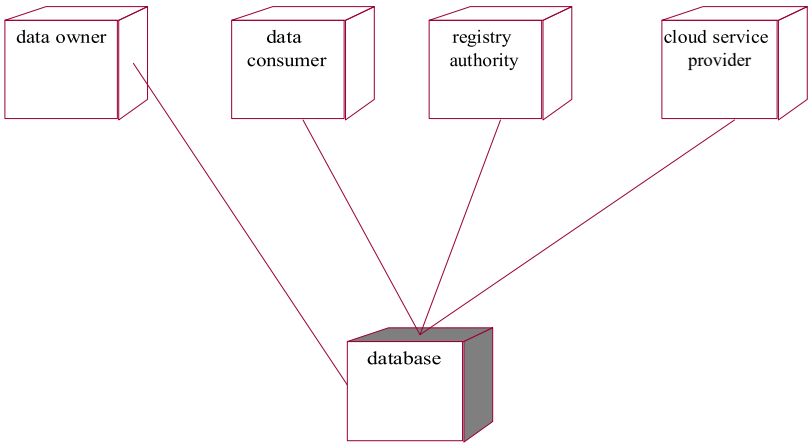
6.2.5 ACTIVITY DIAGRAM



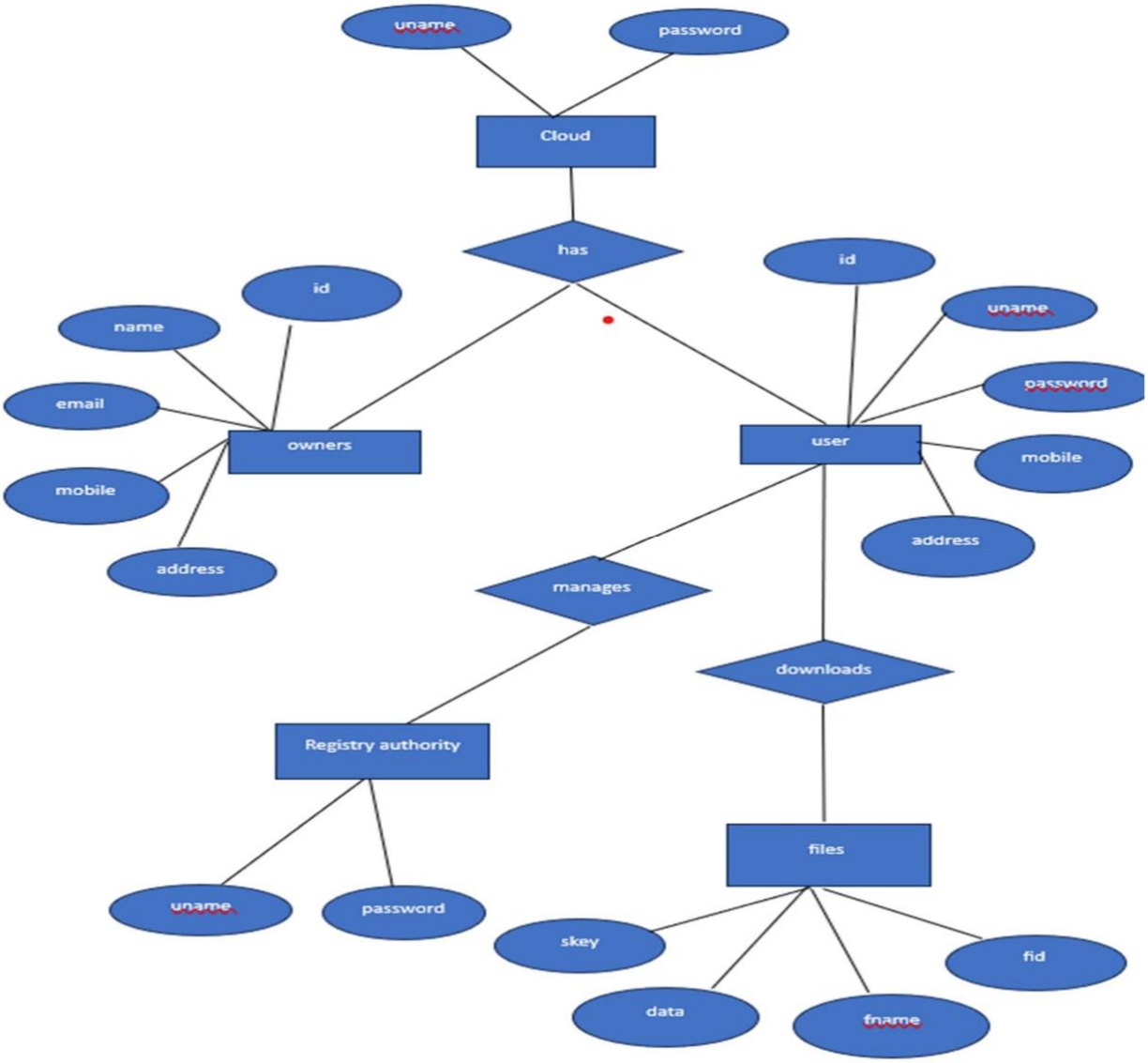
6.2.6 COMPONENT DIAGRAM



6.2.7 DEPLOYMENT DIAGRAM



6.2.8 ER DIAGRAM



6.2.9 DATA DICTIONARY

CLOUD

Column	Datatype	Constraints	Description
username	Varchar(20)	NOTNULL	To enter username
password	Varchar(20)	NOTNULL	To enter password

CONSUMER

Column	Datatype	Constraints	Description
id	Varchar(20)	PRIMARY KEY, UNIQUE	To enter id
name	Varchar(20)	NOTNULL	To enter name
email	Varchar(20)	NOTNULL	To enter email
mobile	Int(10)	NOTNULL	To enter mobile
address	varchar(100)	NOTNULL	To enter address
username	Varchar(20)	NOTNULL	To enter username
password	Varchar(20)	NOTNULL	To enter password
status	Varchar(20)	NOTNULL	To view status

DOWNLOAD

Column	Datatype	Constraints	Description
uid	Int(20)	PRIMARY KEY, UNIQUE	To enter uid
username	Varchar(20)	NOTNULL	To enter username
fname	Varchar(20)	NOTNULL	To enter fname
date	Varchar(20)	NOTNULL	To enter date

FILE

Column	Datatype	Constraints	Description
fid	Int(20)	PRIMARY KEY, UNIQUE	To enter fid
owner	Varchar(20)	NOTNULL	To enter owner
owner_name	Varchar(20)	NOTNULL	To enter owner_name
fname	Varchar(20)	NOTNULL	To enter fname
data	Varchar(20)	NOTNULL	To enter data
skey	Int(20)	NOTNULL	To enter skey

OWNER

Column	Datatype	Constraints	Description
id	Int(20)	PRIMARY KEY, UNIQUE	To enter id
name	Varchar(20)	NOTNULL	To enter name
email	Varchar(20)	NOTNULL	To enter email
mobile	Int(10)	NOTNULL	To enter mobile
address	Varchar(100)	NOTNULL	To enter address
username	Varchar(20)	NOTNULL	To enter username
password	Varchar(20)	NOTNULL	To enter password
status	Varchar(20)	NOTNULL	To view status

REGISTRY AUTHORITY

Column	Datatype	Constraints	Description
username	Varchar(20)	NOTNULL	To enter username
password	Varchar(20)	NOTNULL	To enter password

USER

Column	Datatype	Constraints	Description
id	Int(20)	PRIMARY KEY, UNIQUE	To enter id
name	Varchar(20)	NOTNULL	To enter name
email	Varchar(20)	NOTNULL	To enter email
mobile	Int(10)	NOTNULL	To enter mobile
address	Varchar(20)	NOTNULL(100)	To enter address
username	Varchar(20)	NOTNULL	To enter username
password	Varchar(20)	NOTNULL	To enter password
status	Varchar(20)	NOTNULL	To view status

CHAPTER-7

INPUT AND OUTPUT DESIGN

7.1 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.

7.1.1 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.

7.2 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.

CHAPTER-8
IMPLEMENTATION

8.1 MODULES

- Data Owner
- Data Owner
- Cloud Service provider
- Registry Authority

8.1.1 MODULE DESCRIPTION

1. **DATA OWNER**

Is an entity who encrypts its documents under an arbitrary access control policy and outsources them to the cloud. He/She considers the time of encrypting in generating the cipher texts. We should highlight that the data owner also encrypts his/her documents under his/her arbitrary access control policy. However, in this paper we concentrate on the encryption of the extracted keywords from documents.

2. **DATA USER**

Is an entity who is looking for documents which contains an intended keyword, and are encrypted in a determined time interval. The time interval is arbitrarily selected by the data user.

3. **CLOUD SERVICE PROVIDER**

Is an entity with powerful computation and storage resources. CS stores a massive amount of encrypted data, and receives the search tokens to look for the required documents on behalf of the data user. The cloud finds the relevant documents, and sends them back to the data user.

4. **REGISTRY AUTHORITY**

Is a fully trusted entity who receives each user's access tree, and generates their secret keys corresponding to his/her attributes set presented in his/her access tree. Then, the TTP sends back the users' credentials through a secure and authenticated channel.

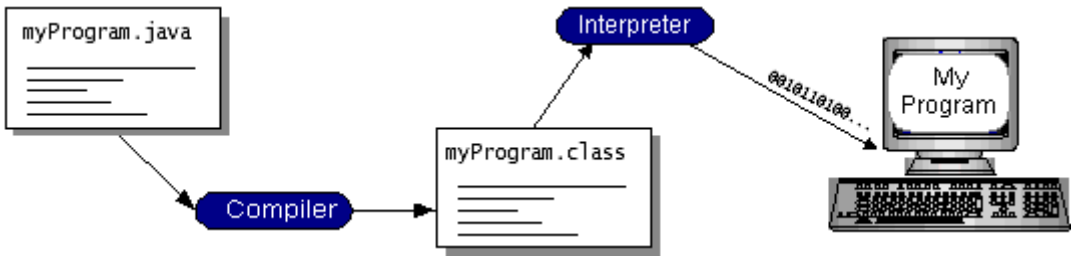
CHAPTER-9
SOFTWARE ENVIRONMENT

9.1 JAVA TECHNOLOGY

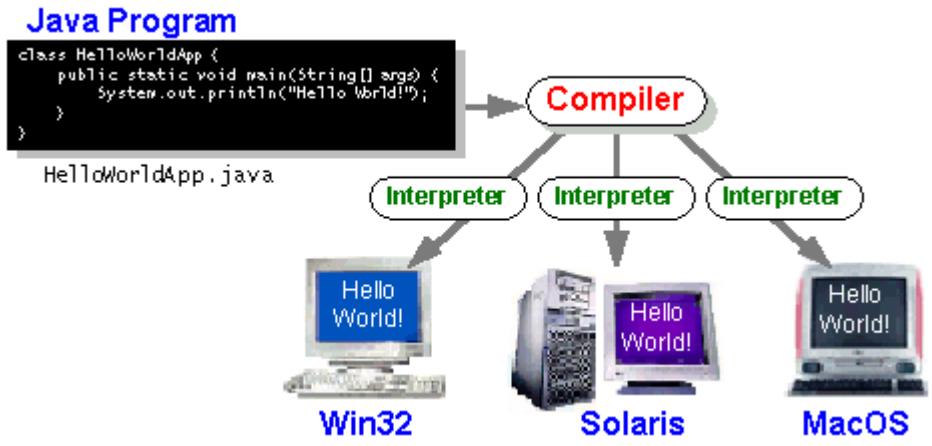
Java technology is both a programming language and a platform. The Java Programming Language. **The Java programming language is a high-level language that can be characterized by all of the following buzzwords:**

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

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. The following figure illustrates how this works.



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.



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 MacOS. 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:

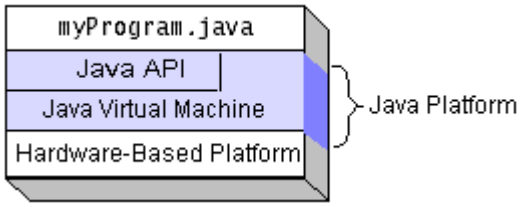
- The *Java Virtual Machine* (Java VM)
- The *Java Application Programming Interface* (Java API)

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.

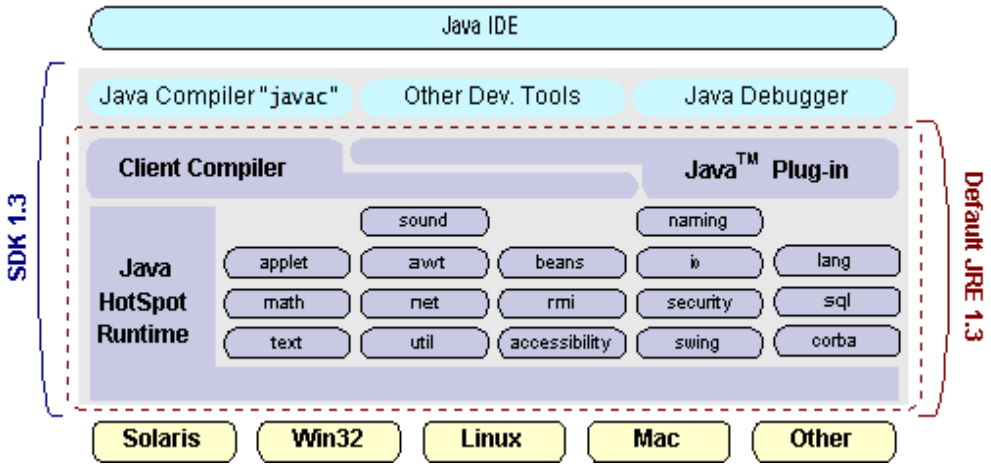
An application is a standalone program that runs directly on the Java platform. A special kind of application known as a *server* serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a *servlet*. 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. Every full implementation of the Java platform gives you the following features:

- **The essentials**: Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
- **Applets**: The set of conventions used by applets.
- **Networking**: URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.
- **Internationalization**: Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
- **Security**: Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
- **Software components**: Known as JavaBeans™, can plug into existing component architectures.
- **Object serialization**: Allows lightweight persistence and communication via Remote Method Invocation (RMI).
- **Java Database Connectivity (JDBC™)**: Provides uniform access to a wide range of relational databases.

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.



How Will Java Technology Change My Life?

We can't promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages. We believe that Java technology will help you do the following:

- **Get started quickly**: Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.
- **Write less code**: Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
- **Write better code**: The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people's tested code and introduce fewer bugs.
- **Develop programs more quickly**: Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.
- **Avoid platform dependencies with 100% Pure Java**: You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure Java™ Product Certification Program has a repository of

historical process manuals, white papers, brochures, and similar materials online.

- **Write once, run anywhere**: Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
- **Distribute software more easily**: You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded "on the fly," without recompiling the entire program.

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:

1. ***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.

2. ***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.

3. ***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.

4. ***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.

5. ***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.

6. ***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.

7. ***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.

Finally, we decided to proceed the implementation using Java Networking. And for dynamically updating the cache table we go for MS Access database.

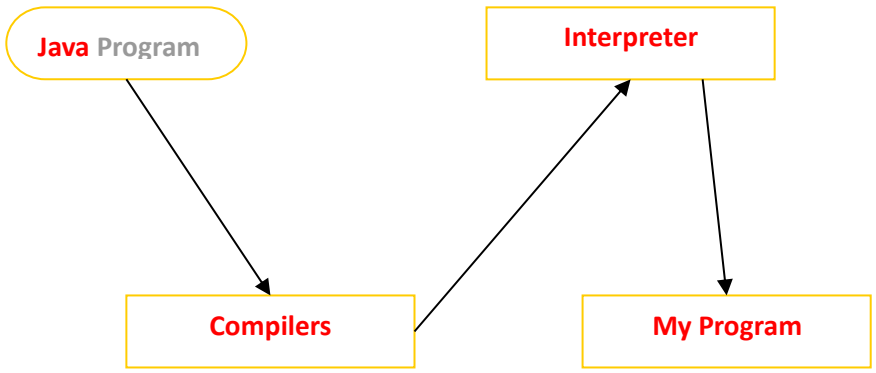
Java ha two things: a programming language and a platform.

Java is a high-level programming language that is all of the following

Simple	Architecture-neutral
Object-oriented	Portable
Distributed	High-performance
Interpreted	multithreaded
Robust	Dynamic
Secure	

Java is also unusual in that each Java program is both compiled and interpreted. With a compile you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer.

Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.



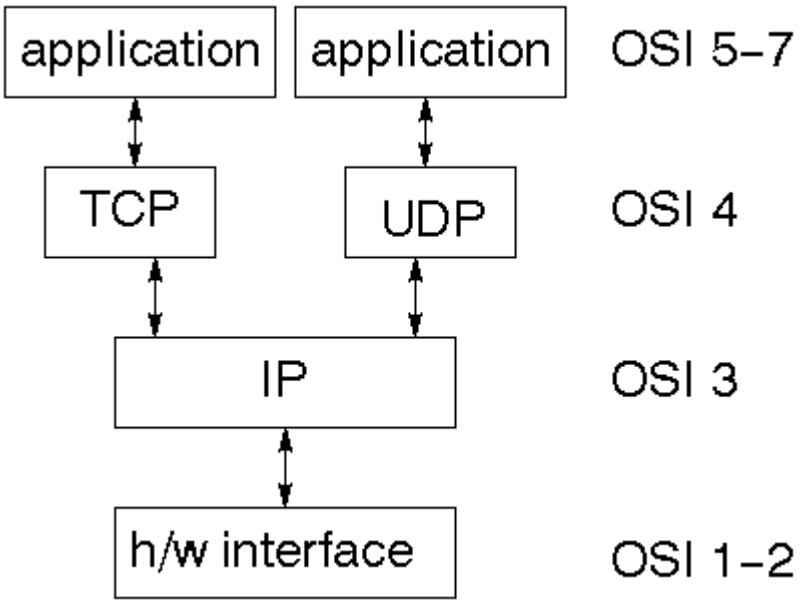
You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

Java byte codes help make "write once, run anywhere" possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

Networking

TCP/IP stack

The TCP/IP stack is shorter than the OSI one:



TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

IP datagram's

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

UDP

UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model - see later.

TCP

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

Internet addresses

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32 bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

Network address

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16 bit network addressing. Class C uses 24 bit network addressing and class D uses all 32.

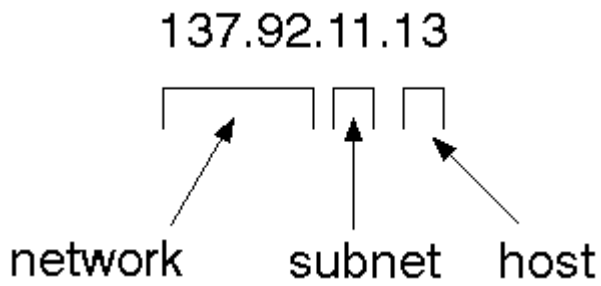
Subnet address

Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

Host address

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

Total address



The 32 bit address is usually written as 4 integers separated by dots.

Port addresses

A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

Sockets

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call socket. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with Read File and Write File functions.

```
#include <sys/types.h>
#include <sys/socket.h>
int socket(int family, int type, int protocol);
```

Here "family" will be AF_INET for IP communications, protocol will be zero, and type will depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

JFree Chart

JFreeChart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart's extensive feature set includes:

A consistent and well-documented API, supporting a wide range of chart types;

A flexible design that is easy to extend, and targets both server-side and client-side applications;

Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

JFreeChart is "open source" or, more specifically, free software. It is distributed under the terms of the GNU Lesser General Public Licence (LGPL), which permits use in proprietary applications.

1. Map Visualizations

Charts showing values that relate to geographical areas. Some examples include: (a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world. The tasks in this project include:

Sourcing freely redistributable vector outlines for the countries of the world, states/provinces in particular countries (USA in particular, but also other areas);

Creating an appropriate dataset interface (plus default implementation), a rendered, and integrating this with the existing XYPlot class in JFreeChart;

Testing, documenting, testing some more, documenting some more.

2. Time Series Chart Interactivity

Implement a new (to JFreeChart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data, with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

3. Dashboards

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of JFreeChart chart types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

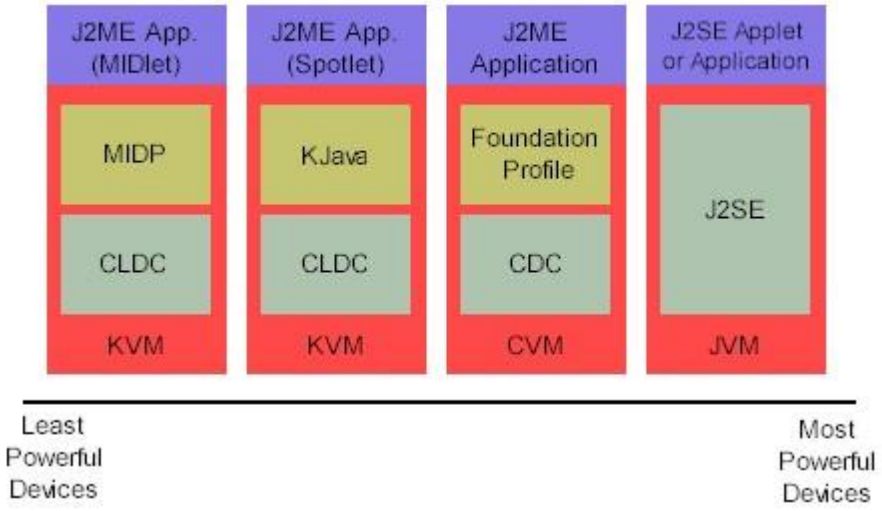
4. Property Editors

The property editor mechanism in JFreeChart only handles a small subset of the properties that can be set for charts. Extend (or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

J2ME (Java 2 Micro edition):-

Sun Microsystems defines J2ME as "a highly optimized Java run-time environment targeting a wide range of consumer products, including pagers, cellular phones, screen-phones, digital set-top boxes and car navigation systems." Announced in June 1999 at the JavaOne Developer Conference, J2ME brings the cross-platform functionality of the Java language to smaller devices, allowing mobile wireless devices to share applications. With J2ME, Sun has adapted the Java platform for consumer products that incorporate or are based on small computing devices.

1. General J2ME architecture



J2ME uses configurations and profiles to customize the Java Runtime Environment (JRE). As a complete JRE, J2ME is comprised of a configuration, which determines the JVM used, and a profile, which defines the application by adding domain-specific classes. The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. We'll discuss configurations in detail in the The profile defines the application; specifically, it adds domain-specific classes to the J2ME configuration to define certain uses for devices. We'll cover profiles in depth in the The following graphic depicts the relationship between the different virtual machines, configurations, and profiles. It also draws a parallel with the J2SE API and its Java virtual machine. While the J2SE virtual machine is generally referred to as a JVM, the J2ME virtual machines, KVM and CVM, are subsets of JVM. Both KVM and CVM can be thought of as a kind of Java virtual machine -- it's just that they are shrunken versions of the J2SE JVM and are specific to J2ME.

2.Developing J2ME applications

Introduction In this section, we will go over some considerations you need to keep in mind when developing applications for smaller devices. We'll take a look at the way the compiler

is invoked when using J2SE to compile J2ME applications. Finally, we'll explore packaging and deployment and the role preverification plays in this process.

3.Design considerations for small devices

Developing applications for small devices requires you to keep certain strategies in mind during the design phase. It is best to strategically design an application for a small device before you begin coding. Correcting the code because you failed to consider all of the "gotchas" before developing the application can be a painful process. Here are some design strategies to consider:

* Keep it simple. Remove unnecessary features, possibly making those features a separate, secondary application.

* Smaller is better. This consideration should be a "no brainer" for all developers. Smaller applications use less memory on the device and require shorter installation times. Consider packaging your Java applications as compressed Java Archive (jar) files.

* Minimize run-time memory use. To minimize the amount of memory used at run time, use scalar types in place of object types. Also, do not depend on the garbage collector. You should manage the memory efficiently yourself by setting object references to null when you are finished with them. Another way to reduce run-time memory is to use lazy instantiation, only allocating objects on an as-needed basis. Other ways of reducing overall and peak memory use on small devices are to release resources quickly, reuse objects, and avoid exceptions.

4.Configurations overview

The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. Currently, two configurations exist for J2ME, though others may be defined in the future:

* **Connected Limited Device Configuration (CLDC)** is used specifically with the KVM for 16-bit or 32-bit devices with limited amounts of memory. This is the configuration (and the virtual machine) used for developing small J2ME applications. Its size limitations make CLDC more interesting and challenging (from a development point of view) than CDC.

CLDC is also the configuration that we will use for developing our drawing tool application. An example of a small wireless device running small applications is a Palm hand-held computer.

* **Connected Device Configuration (CDC)** is used with the C virtual machine (CVM) and is used for 32-bit architectures requiring more than 2 MB of memory. An example of such a device is a Net TV box.

5.J2ME profiles

What is a J2ME profile?

As we mentioned earlier in this tutorial, a profile defines the type of device supported. The Mobile Information Device Profile (MIDP), for example, defines classes for cellular phones. It adds domain-specific classes to the J2ME configuration to define uses for similar devices. Two profiles have been defined for J2ME and are built upon CLDC: KJava and MIDP. Both KJava and MIDP are associated with CLDC and smaller devices. Profiles are built on top of configurations. Because profiles are specific to the size of the device (amount of memory) on which an application runs, certain profiles are associated with certain configurations.

A skeleton profile upon which you can create your own profile, the Foundation Profile, is available for CDC.

Profile 1: KJava

KJava is Sun's proprietary profile and contains the KJava API. The KJava profile is built on top of the CLDC configuration. The KJava virtual machine, KVM, accepts the same byte codes and class file format as the classic J2SE virtual machine. KJava contains a Sun-specific API that runs on the Palm OS. The KJava API has a great deal in common with the J2SE Abstract Windowing Toolkit (AWT). However, because it is not a standard J2ME package, its main package is com.sun.kjava. We'll learn more about the KJava API later in this tutorial when we develop some sample applications.

Profile 2: MIDP

MIDP is geared toward mobile devices such as cellular phones and pagers. The MIDP, like KJava, is built upon CLDC and provides a standard run-time environment that allows new

applications and services to be deployed dynamically on end user devices. MIDP is a common, industry-standard profile for mobile devices that is not dependent on a specific vendor. It is a complete and supported foundation for mobile application

development. MIDP contains the following packages, the first three of which are core CLDC packages, plus three MIDP-specific packages.

* java.lang

* java.io

* java.util

* javax.microedition.io

* javax.microedition.lcdui

* javax.microedition.midlet

* javax.microedition.rms

9.2 SOURCE CODE

```
<!DOCTYPE html>
<html>
<head>
<title>Identity_Based_Encryption</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" href="layout/styles/layout.css" type="text/css" />
<script type="text/javascript" src="layout/scripts/jquery.min.js"></script>
<script type="text/javascript" src="layout/scripts/jquery.ui.min.js"></script>
<script type="text/javascript" src="layout/scripts/jquery.defaultvalue.js"></script>
<script type="text/javascript" src="layout/scripts/jquery.scrollTo-min.js"></script>
<script type="text/javascript">
$(document).ready(function () {
$("#fullname, #validemail, #message").defaultvalue("Full Name", "Email Address",
"Message");
$('#shout a').click(function () {
var to = $(this).attr('href');
$.scrollTo(to, 1200);
return false;
});
$('a.topOfPage').click(function () {
$.scrollTo(0, 1200);
return false;
});
$("#tabcontainer").tabs({
```

```
event: "click"
});
});
</script>
<!-- Homepage Only Scripts -->
<script type="text/javascript" src="layout/scripts/jquery.cycle.min.js"></script>
<script type="text/javascript">
$(function() {
$('#hpage_slider').after('<div id="fsn"><ul id="fs_pagination">').cycle({
timeout: 5000,
fx: 'fade',
pager: '#fs_pagination',
pause: 1,
pauseOnPagerHover: 0
});
});
</script>
<script type="text/javascript" src="layout/scripts/piecemaker/swfobject/swfobject.js"></script>
<script type="text/javascript">
var flashvars = {};
flashvars.cssSource = "layout/scripts/piecemaker/piecemaker.css";
flashvars.xmlSource = "layout/scripts/piecemaker/piecemaker.xml";
var params = {};
params.play = "false";
```

```
params.menu = "false";

params.scale = "showall";

params.wmode = "transparent";

params.allowfullscreen = "true";

params.allowscriptaccess = "sameDomain";

params.allownetworking = "all";

swfobject.embedSWF('layout/scripts/piecemaker/piecemaker.swf', 'piecemaker', '960',
'430', '10', null, flashvars, params, null);

</script>

<!-- End Homepage Only Scripts -->

</head>

<body id="top">

<div class="wrapper col1">

<div id="topbar" class="clear">

<h3 style="color:red;">Identity-Based Encryption Transformation for Flexible Sharing
of

Encrypted Data in Public Cloud</h3>

</div>

</div>

<!--

################################################################################
#####

############################## -->

<div class="wrapper col2">

<div id="header" class="clear">
```

```
<div class="fl_left">
<h1><a href="#">IBE</a></h1>
<p>Identity-Based Encryption</p>
</div>
<div id="topnav">
<ul>
<li class="last"><a href="cloudserviceprovider.jsp">Cloud Service Provider</a></li>
<li><a href="RegisterAuthority.jsp">Registry Authority</a></li>

<li><a href="DataConsumer.jsp">Data Consumers</a></li>
<li><a href="DataOwner.jsp">Data Owners</a></li>
<li class="active"><a href="index.html">Home Page</a></li>
</ul>
</div>
</div>
</div>
<!--
################################################################################
#####
############################## -->
<div class="wrapper col3">
<div id="featured_slide">
<!--
```

```
################################################################################
#####

############################### -->

<div id="piecemaker"><img src="images/Capture.PNG" width="960" height="360"
alt=""

/></div>

<!--

################################################################################
#####

############################### -->

</div>

</div>

<!--

################################################################################
#####

############################### -->

<div class="wrapper col4">

<div id="container" class="clear">

<!--

################################################################################
#####

############################### -->

<div id="shout" class="clear">

<div class="fl_left">
```

<h2>Identity-Based Encryption Transformation for Flexible Sharing of Encrypted Data in Public Cloud</h2>

<!--

##
#####

############################## -->

<image src="images/imag.jpg"/>

<h2>Abstract</h2>

<p align="justify">With the rapid development of cloud computing,an increasing number of individuals and organizations are sharing data in the public cloud.

To protect the privacy of data stored in the cloud,a data owner usually encrypts his data in such a way that certain

designated data users can decrypt the data.

This raises a serious problem when the encrypted data needs to be shared to more people beyond those initially

designated by the data owner.To address this problem,

we introduce and formalize an identity-based encryption transformation(IBET)model by seamlessly integrating two

well-established encryption mechanisms,namely identity-based encryption(IBE)and identitybased broadcast encryption

(IBBE).In IBET,data users are identified and authorized for data access based on their recognizable identities,

which avoids complicated certificate management in usual secure distributed systems.More importantly,

IBET provides a transformation mechanism that converts an IBE cipher text into an IBBE ciphertext

so that a new group of users not specified during the IBE encryption can access the underlying data.

We design a concrete IBET scheme based on bilinear groups and prove its security against powerful attacks.

Thorough theoretical and experimental analyses demonstrate the high efficiency and practicability of the proposed scheme.</p>

</body>

</html>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<title>Identity_Based_Encryption</title>

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />

```
<link rel="stylesheet" href="layout/styles/layout.css" type="text/css" />
<script type="text/javascript" src="layout/scripts/jquery.min.js"></script>
<script type="text/javascript" src="layout/scripts/jquery.ui.min.js"></script>
<script type="text/javascript" src="layout/scripts/jquery.defaultvalue.js"></script>
<script type="text/javascript" src="layout/scripts/jquery.scrollTo-min.js"></script>
<script type="text/javascript">
$(document).ready(function () {
$("#fullname, #validemail, #message").defaultvalue("Full Name", "Email Address",
"Message");
$('#shout a').click(function () {
var to = $(this).attr('href');
$.scrollTo(to, 1200);
return false;
});
$('a.topOfPage').click(function () {
$.scrollTo(0, 1200);
return false;
});
$("#tabcontainer").tabs({
event: "click"
});
});
</script>
<!-- Homepage Only Scripts -->
<script type="text/javascript" src="layout/scripts/jquery.cycle.min.js"></script>
```

```
<script type="text/javascript">
$(function() {
$('#hpage_slider').after('<div id="fsn"><ul id="fs_pagination">').cycle({
timeout: 5000,
fx: 'fade',
pager: '#fs_pagination',
pause: 1,
pauseOnPagerHover: 0
});
});
</script>
<script type="text/javascript" src="layout/scripts/piecemaker/swfobject/swfobject.js"></script>
<script type="text/javascript">
var flashvars = {};
flashvars.cssSource = "layout/scripts/piecemaker/piecemaker.css";
flashvars.xmlSource = "layout/scripts/piecemaker/piecemaker.xml";
var params = {};
params.play = "false";
params.menu = "false";
params.scale = "showall";
params.wmode = "transparent";
params.allowfullscreen = "true";
params.allowscriptaccess = "sameDomain";
params.allownetworking = "all";
```

```
swfobject.embedSWF('layout/scripts/piecemaker/piecemaker.swf', 'piecemaker', '960',
'430', '10', null, flashvars, params, null);
</script>
<!-- End Homepage Only Scripts -->
</head>
<body id="top">
<div class="wrapper col1">
<div id="topbar" class="clear">
<h3 style="color:red;">Identity-Based Encryption Transformation for Flexible Sharing
of
Encrypted Data in Public Cloud</h3>
</div>
</div>
<!--
################################################################################
#####
############################## -->
<div class="wrapper col2">
<div id="header" class="clear">
<div class="fl_left">
<h1><a href="#">IBE</a></h1>
<p>Identity-Based Encryption</p>
</div>
<div id="topnav">
<ul>
```

```
<li class="last"><a href="cloudserviceprovider.jsp">Cloud Service Provider</a></li>
<li><a href="RegisterAuthority.jsp">Registry Authority</a></li>
<li><a href="DataConsumer.jsp">Data Consumers</a></li>
<li class="active"><a href="DataOwner.jsp">Data Owners</a></li>
<li><a href="index.html">Home Page</a></li>
</ul>
</div>
</div>
</div>
<!--
##########################################################################
#####
############################## -->
<%--
Document   : ActivateOwner
Created on : 30 Sep, 2020, 12:36:23 PM
Author     : KishanVenky
--%>
<%@page import="com.database.Queries"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<%
String id=request.getParameter("id");
try{
String query="update consumer set status='Authorized' where id='"+id+"'";
```

```
int i=Queries.getExecuteUpdate(query);
if(i>0){
%>
<script type="text/javascript">
window.alert("Successfully Authorized");
window.location="RAViewUsers.jsp";
</script>
<%
}else{
%>
<script type="text/javascript">
window.alert("Failed Authorization");
window.location="RAViewUsers.jsp";
</script>
<%
}
}catch(Exception e){
out.println(e);
}
%>
<!--
############################## -->
<div class="wrapper col4">
<div id="container" class="clear">
```

<!--

##
#####

############################### -->

<h2>Identity-Based Encryption Transformation for Flexible Sharing of Encrypted Data in Public Cloud</h2>

CHAPTER-10
RESULTS/DISCUSSIONS

10.1 SYSTEM 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, subassemblies, 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.

TYPES OF TESTS

UNIT TESTING

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

INTEGRATION TESTING

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

FUNCTIONAL TEST

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

Functional testing is centered on the following items:

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

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

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

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

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

SYSTEM TEST

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

WHITE BOX TESTING

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

BLACK BOX TESTING

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

UNIT TESTING

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

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

Test objectives

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

Features to be tested

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

Integration Testing

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

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

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

ACCEPTANCE TESTING

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

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

10.1.1 TEST CASES

Test case1:

Test case for Login form

FUNCTION:	LOGIN
EXPECTED RESULTS:	Should Validate the user and check his existence in database
ACTUAL RESULTS:	Validate the user and checking the user against the database
LOW PRIORITY	No
HIGH PRIORITY	Yes

Test case2 :

Test case for User Registration form

FUNCTION:	USER REGISTRATION
EXPECTED RESULTS:	Should check if all the fields are filled by the user and saving the user to database.
ACTUAL RESULTS:	Checking whether all the fields are field by user or not through validations and saving user.
LOW PRIORITY	No

HIGH PRIORITY	Yes

Test case3:

Test case for Change Password

When the old password does not match with the new password ,then this results in displaying an error message as " OLD PASSWORD DOES NOT MATCH WITH THE NEW PASSWORD".

FUNCTION:	Change Password
EXPECTED RESULTS:	Should check if old password and new password fields are filled by the user and saving the user to database.
ACTUAL RESULTS:	Checking whether all the fields are field by user or not through validations and saving user.
LOW PRIORITY	No
HIGH PRIORITY	Yes

Test case 4:

Test case for Forget Password:

When a user forgets his password he is asked to enter Login name, ZIP code, Mobile number.

If these are matched with the already stored ones then user will get his Original password.

Module	Functionality	TestCase	Expected Results	Actual Results	Result	Priority
User	Login Usecase	1. Navigate To Www.Sample.Com 2. 2.Click On Submit Button Without Entering Username and Password	A Validation Should Be As Below “Please Enter Valid Username & Password”	A Validation Has Been Populated As Expected	Pass	High
		1. aNavigate To Www.Sample.Com 1. 2. Click On Submit Button With Out Filling Password And With Valid Username Test UsernameField	A Validation Should Be As Below “Please Enter Valid Password Or Password Field Can Not Be Empty “	A Validation Is Shown As Expected	Pass	High

		1. NNavigate To Www.Sample.Com 2. Enter Both	A Validation Shown As Below “The Username Entered Is Wrong”	A Validation Is Shown As Expected	Pass	High
		Username And Password Wrong And Hit Enter				
		1. Navigate To Www.Sample.Com 2. Enter Validate Username And Password And Click On Submit	Validate Username And Password In DataBase And Once If They Correct Then Show The Main Page	Main Page/ Home Page Has Been Displayed	Pass	High

10.2 SCREEN SHOTS

FIG 1 : HOME PAGE

A home page is a webpage that serves as the starting point of website. It is the default webpage that loads when you visit a web address that only contains a domain name.

FIG 2: DATA OWNER:LOGIN PAGE

A data owner login page is a web interface designed to allow authorized users, typically individuals or organizations, to access and manage their data stored on a particular platform or system.

FIG 3 : DATA OWNER: HOME PAGE

Data owner home page is a dashboard or landing page within a system or application specifically designed for individuals or organizations who own and manage data. This page provides a comprehensive overview and access to various features, tools, and information related to the data owned by the user.

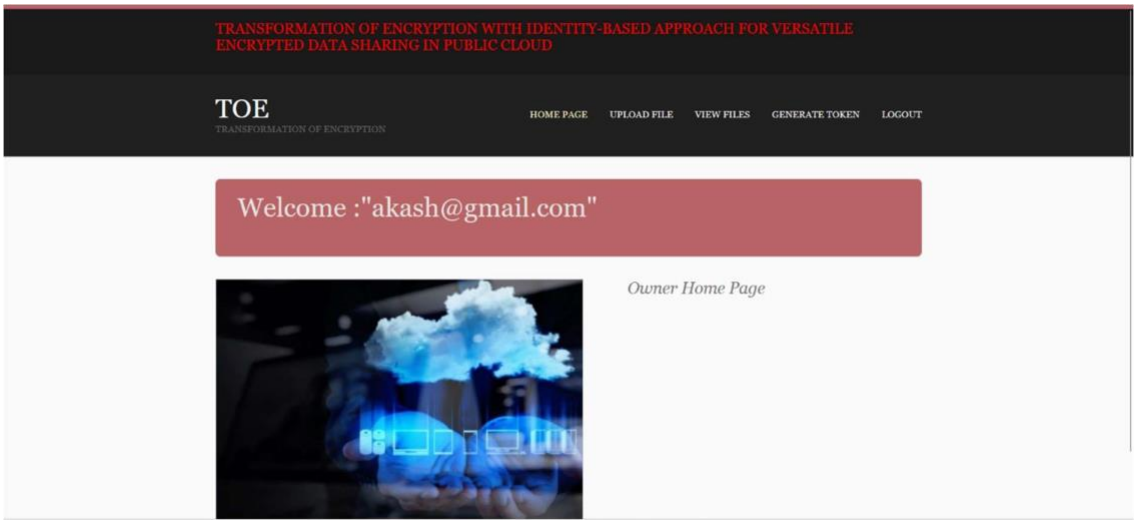


FIG 4 : FILE UPLOADING PAGE

A file uploading page is a web interface or screen within an application that allows users to upload files from their local devices or cloud storage to a server or a storage system.

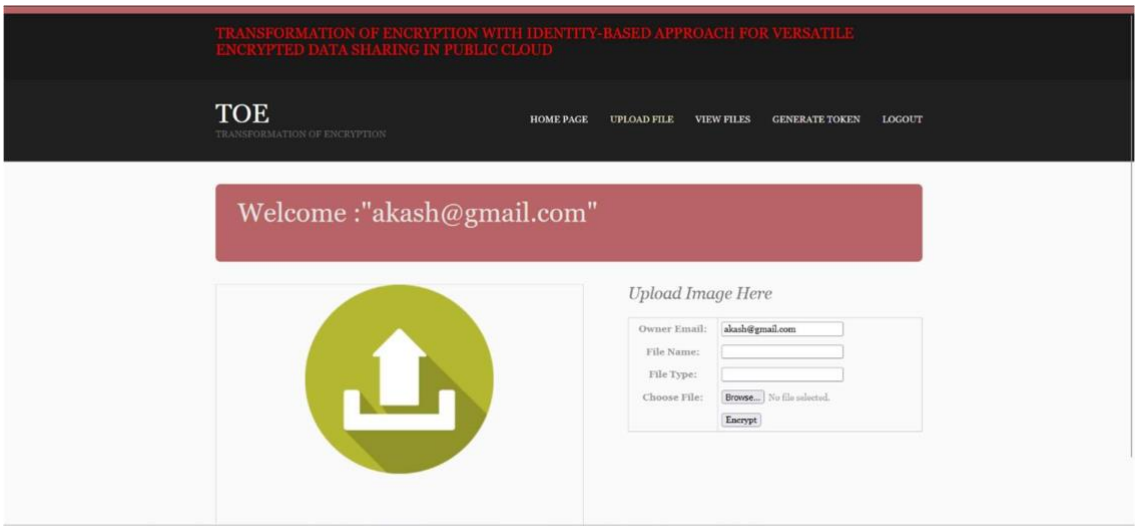


FIG 5: VIEW FILES PAGE

A View Files page is a web interface or screen within an application or platform that allows users to see and manage the files they have uploaded or stored.

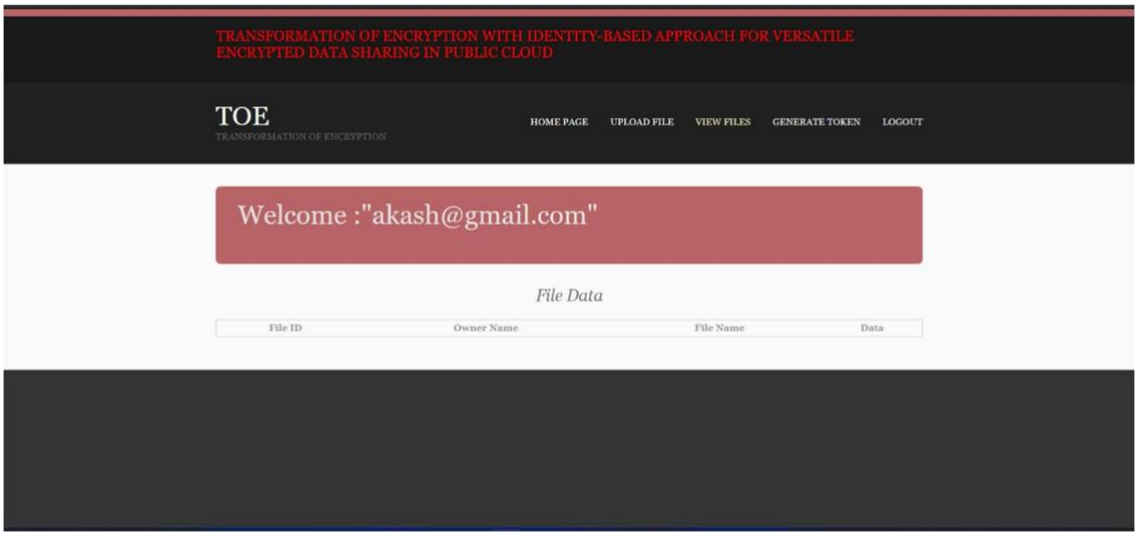


FIG 6: GENERATE TOKEN PAGE

A "Generate Token" page is a web interface within an application that allows users or developers to create authentication tokens, typically for security and access control purposes.

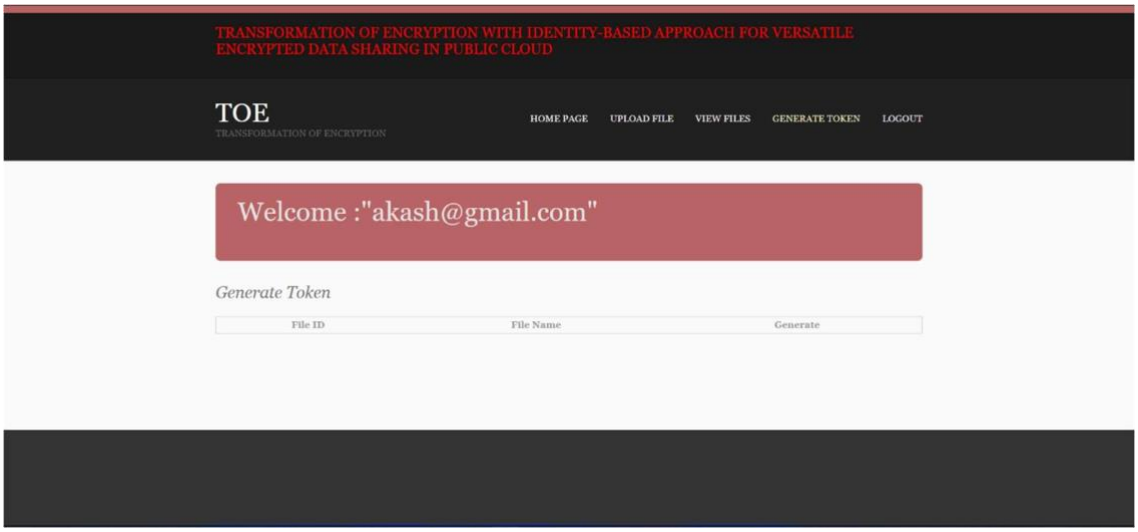


FIG 7 : DATA CONSUMER:LOGIN PAGE

A data consumer login page is a web interface designed to allow authorized users, typically individuals or organizations, to access and manage their data stored on a particular platform or system.

FIG 8: CONSUMER HOME PAGE

Data consumer home page is a dashboard or landing page within a system or application specifically designed for individuals or organizations who want to access data. This page provides a comprehensive overview and access to various features, tools, and information related to the data owned by the user.

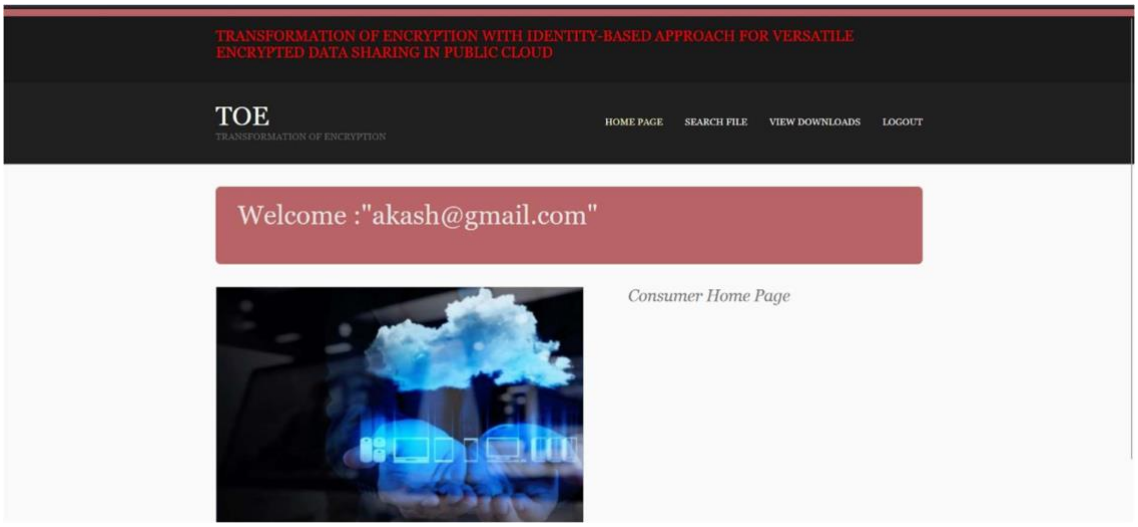


FIG 9: VIEW DOWNLOADS PAGE

A "View Downloads" page is a web interface or component within an application that allows users to see a history or list of files and data they have downloaded.

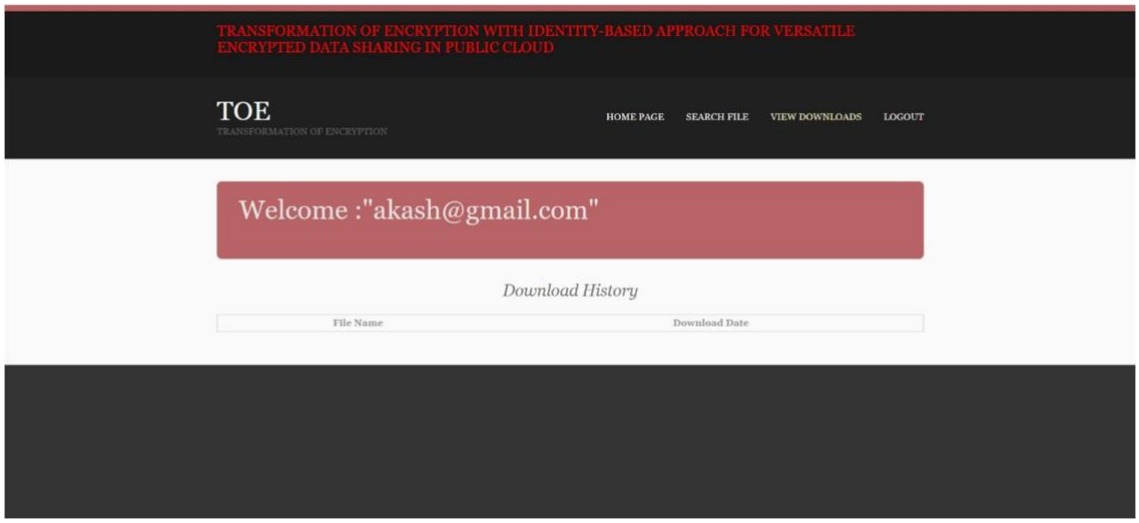


FIG 10: REGISTRY AUTHORITY:LOGIN PAGE

The "Registry Authority: Login Page" is a specific web interface or screen designed for users who are authorized to access a registry authority system. A registry authority is typically an organization or entity responsible for managing and overseeing a specific registry, which could contain information about various entities such as individuals, organizations, products, or services.

FIG 11: REGISTRY AUTHORITY HOME PAGE

The "Registry Authority Home Page" is a central web interface within a registry authority system. It serves as a dashboard and control center for authorized users, providing access to various tools, resources, and information related to managing a specific registry.

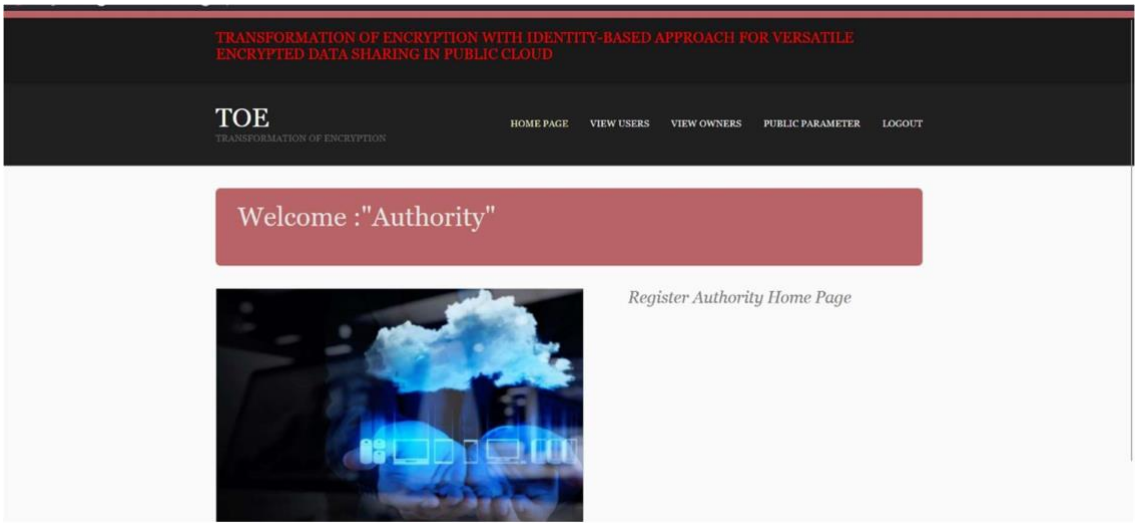


FIG 12: VIEW USERS

A "View Users" page is a web interface or screen within an application or system that allows administrators or authorized users to see and manage a list of registered users.

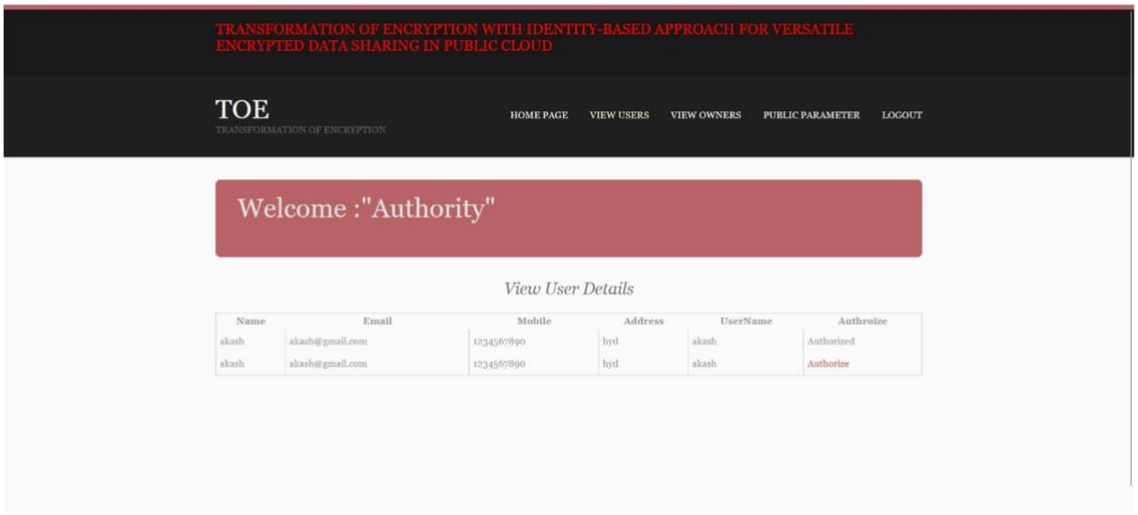


FIG 13: AUTHORIZE USERS

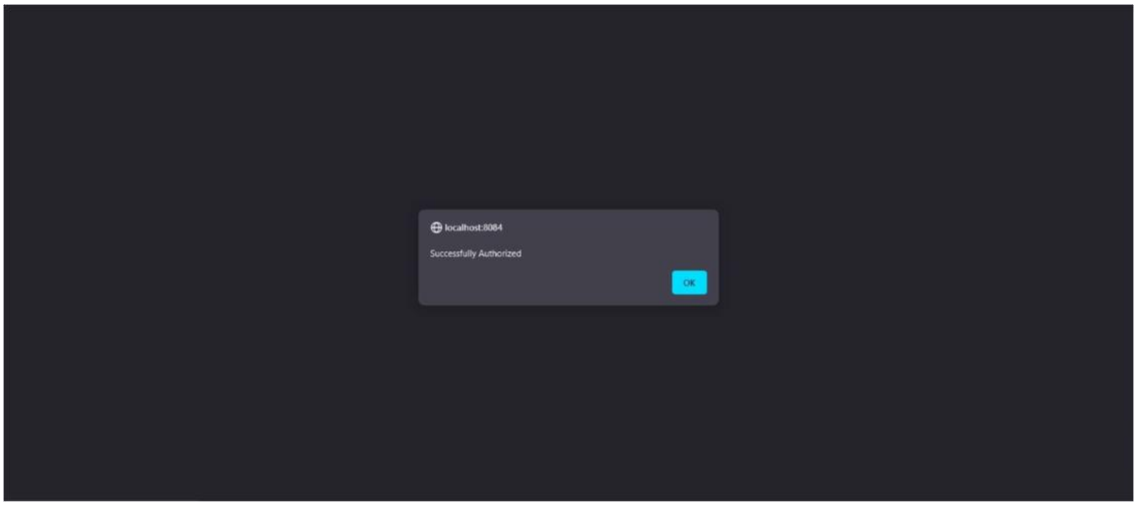


FIG 14 : VIEW OWNER DETAILS

A "View Owners Details" page is a web interface or screen within an application or system that allows authorized users, often administrators or staff members, to access and review detailed information about the owners of data.

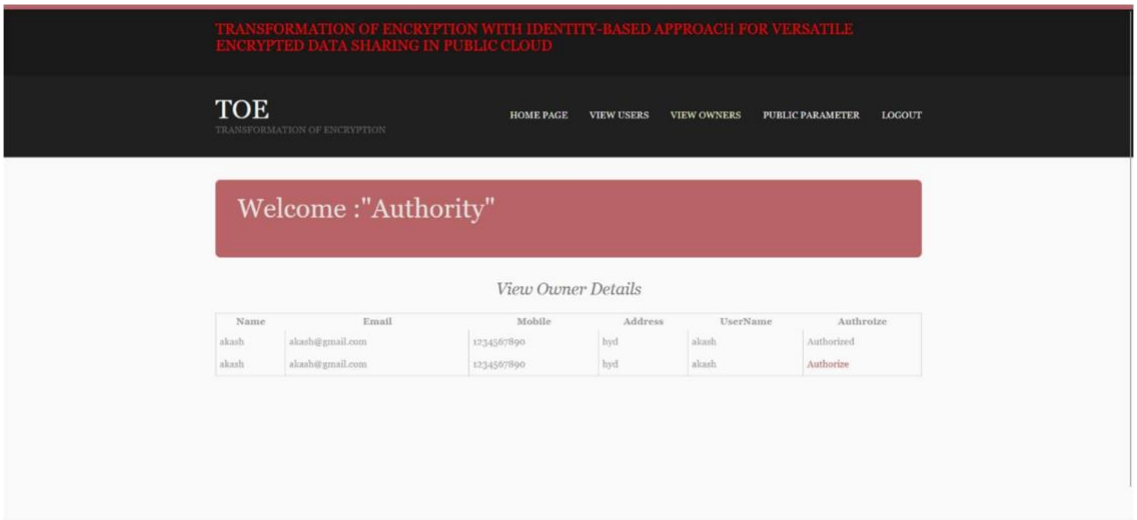


FIG 15: AUTHORIZE OWNERS

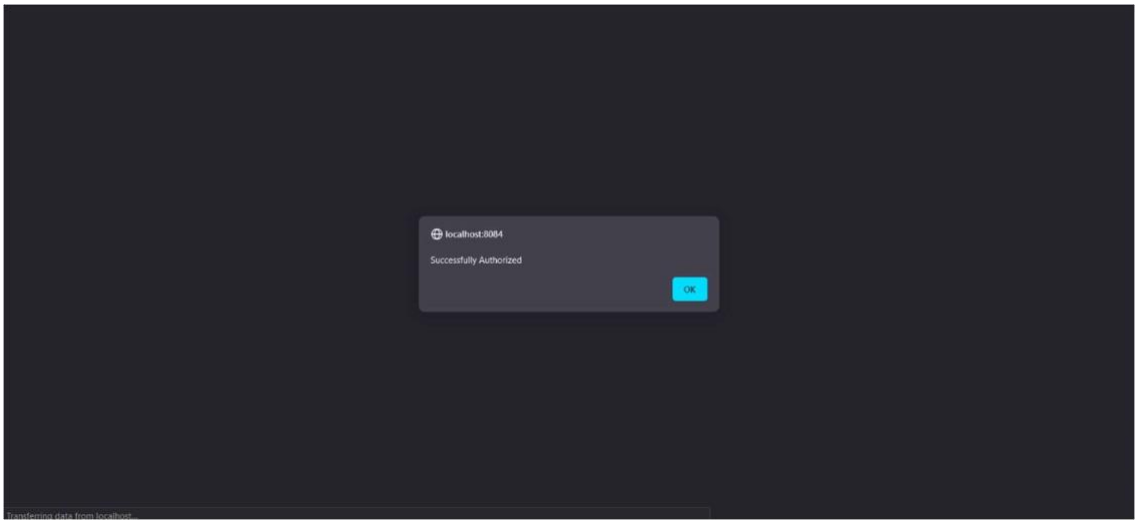


FIG 16: CLOUD: LOGIN PAGE

A cloud login page is a web interface or screen within a cloud-based service that allows users to access their accounts and resources remotely via the internet.

CHAPTER-11
CONCLUSION

11.1 CONCLUSION

In this paper we studied how to securely and efficiently transform encrypted data in clouds. To address this issue, we proposed an identity-based encryption transformation (IBET) model, which connects the well-studied IBE and IBBE systems. IBET allows data owners to secure outsourced data with identity-based access control, which eliminates complicated cryptographic certificates for all users. Moreover, IBET provides a transformation mechanism for data owners to authorize cloud service provider (CSP) to transform a file in IBE-ciphertext formant into a file in IBBE-ciphertext format, so that a set of authorized users can access the underlying data. We proposed a concrete IBET scheme that is secure against powerful attacks. Thorough experimental analyses demonstrate the efficiency and practicability of the scheme.

11.2 FUTURE SCOPE

The integration of Identity-Based Encryption (IBE) and Identity-Based Broadcast Encryption (IBBE) in the proposed Identity-Based Encryption Transformation (IBET) model signifies a significant leap in cloud data security. With the escalating use of public cloud services, IBET resolves the challenge of data sharing beyond initially designated users. By authenticating users based on recognizable identities, IBET eliminates the complexities of certificate management in secure distributed systems. This innovative approach not only ensures data privacy but also offers seamless transformation of IBE ciphertexts into IBBE ciphertexts, allowing broader access without compromising security. The use of bilinear groups and rigorous security proofs establishes IBET as a robust and efficient solution, paving the way for enhanced cloud data protection and accessibility in the future.

CHAPTER-12
REFERENCES

[1] D. Song, E. Shi, I. Fischer, and U. Shankar, “Cloud data protection for the masses,” Computer, vol. 45, no. 1, pp. 39–45, 2012.

[2] J. Yu, K. Ren, and C. Wang, “Enabling cloud storage auditing with verifiable outsourcing of key updates,” IEEE Transactions on Information Forensics and Security, vol. 11, no. 6, pp. 1362–1375, 2016.

[3] H. Yin, Z. Qin, J. Zhang, L. Ou, and K. Li, “Achieving secure, universal, and fine-grained query results verification for secure search scheme over encrypted cloud data,” IEEE Transactions on Cloud Computing, 2017.

[4] K. Li, W. Zhang, C. Yang, and N. Yu, “Security analysis on one-tomany order preserving encryption-based cloud data search,” IEEE Transactions on Information Forensics and Security, vol. 10, no. 9, pp. 1918–1926, 2015.

[5] R. Zhang, R. Xue, and L. Liu, “Searchable encryption for healthcare clouds: a survey,” IEEE Transactions on Services Computing, vol. 11, no. 6, pp. 978–996, 2018.

[6] D. Boneh and M. Franklin, “Identity-based encryption from the weil pairing,” SIAM Journal on Computing, vol. 32, no. 3, pp. 586– 615, 2003.

[7] J. Wei, W. Liu, and X. Hu, “Secure data sharing in cloud computing using revocable-storage identity-based encryption,” IEEE Transactions on Cloud Computing, 2016.

[8] D. He, N. Kumar, H. Wang, L. Wang, K.-K. R. Choo, and A. Vinel, “A provably-secure cross-domain handshake scheme with symptoms-matching for mobile healthcare social network,” IEEE Transactions on Dependable and Secure Computing, vol. 15, no. 4, pp. 633– 645, 2018.

[9] C. Delerable´e, “Identity-based broadcast encryption with constant size ciphertexts and private keys,” in International Conference on the Theory and Application of Cryptology and Information Security. Springer, 2007, pp. 200–215.

[10] H. Deng, Q. Wu, B. Qin, W. Susilo, J. Liu, and W. Shi, “Asymmetric cross-cryptosystem re-encryption applicable to efficient and secure mobile access to outsourced data,” in Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security. ACM, 2015, pp. 393–404.