ROBE- ENCRYPTION PROTOCOL FOR SECURE MOBILE CLOUD COMPUTING

Project Report

Submitted by

Mr. ALBIN JOHN
Mr. SHAJAL N C
Ms. DRISHYA BABU V
Ms. MEGHA T P
Ms. NEETHU PREM

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

COLLEGE OF ENGINEERING VATAKARA
(Engineering College Under CAPE, Estd by Govt. of Kerala)

MARCH 2018

COLLEGE OF ENGINEERING VATAKARA
(Engineering College Under CAPE, Estd by Govt. of Kerala)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

*This is to certify that the project entitled "**ROBE- ENCRYPTION PROTOCOL FOR SECURE MOBILE CLOUD COMPUTING**" is the bonafide work carried out by. **MR. ALBIN JOHN (Reg.no: 12152404), MR. SHAJAL N C(Reg.no: 12152418), MS. DRISHYA BABU V (Reg.no: 12152432), MS. MEGHA T P (Reg.no: 12152443), MS. NEETHU PREM (Reg.no:12152445)** of B.Tech (Computer Science & Engineering, 2014 Admission), College of Engineering Vatakara, Kerala during the year 2018, in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology of CUSAT and that the project has not formed the basis for the award previously of any degree, diploma, associateship, fellowship or any other similar title.*

PROJECT COORDINATOR

AKHILA M
Assistant Professor
Department of Computer Science &Engg
College of Engineering Vatakara

HEAD OF THE DEPARTMENT

ARUNA A S
Assistant Professor
Department of Computer Science& Engg.
College of Engineering Vatakara

COLLEGE OF ENGINEERING VATAKARA
(Engineering College Under CAPE, Estd by Govt. of Kerala)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

*Certified that this project report "**ROBE- ENCRYPTION PROTOCOL FOR SECURE MOBILE CLOUD COMPUTING**" is the bonafide work of **MR. ALBIN JOHN (Reg.no: 12152404), MR. SHAJAL N C (Reg.no: 12152418), MS. DRISHYA BABU V (Reg.no: 12152432), MS. MEGHA T P (Reg.no: 12152443), MS. NEETHU PREM (Reg.no:12152445)** who carried out the project work under my supervision.*

PROJECT GUIDE

SHIBILI T
Assistant Professor
Department of Computer Science & Engg
College of Engineering Vatakara

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
COLLEGE OF ENGINEERING VADAKARA

INSTITUTION

VISION :-

To be a world class technical education institute with the highest quality and standards of excellence to meet the demands of business, industry and the community and thereby to contribute to India's socioeconomic progress.

MISSION :-

To develop high quality technical personnel with a sound footing on basic engineering principles, innovative research capabilities and exemplary professional conduct to lead and use technology for the progress of mankind ,adapting themselves to changing technological environment with the highest ethical values as the inner strength.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
COLLEGE OF ENGINEERING VADAKARA

DEPARTMENT

VISION:-

To produce competent Software Professionals that are capable of analysing organizational problems, design, formulate and evaluate, develop and maintain technological solutions for the betterment of the global economy.

MISSION:-

- Upgrading the existing software and hardware facilities in the computer science centers and laboratories to match the outcomes of the programs offered.
- Organize need based faculty and staff development programs and encourage faculty to participate in national and international conferences.
- Encourage faculty to organize frequent departmental seminars and use class room seminar methods to improve abilities of the students.
- Sponsor faculty and staff on study leave to upgrade their qualification.
- Explore opportunities for development assessment of Graduate Attributes in the co-curricular and extracurricular activities.
- To mould the students by inculcating the professional ethics.
- To prepare graduates for leadership in profession and in higher education.
- To develop entrepreneurship abilities by intense interaction with professional entrepreneurs.

DECLARATION

We hereby declare that the project entitled ***"ROBE- ENCRYPTION PROTOCOL FOR SECURE MOBILE CLOUD COMPUTING"*** *submitted for the B.Tech(Computer Science & Engineering) Degree is our original work and theproject has not formed the basis for the award of any degree, associate ship, fellowship or any other similar titles.*

ALBIN JOHN
SHAJAL N C
DRISHYA BABU V
MEGHA T P
NEETHU PREM

Place : Vatakara
Date :

ACKNOWLEDGEMENT

Any mission never concludes without cordial co-operation from surroundings. we take this opportunity to acknowledge all the people who have helped us kind heartedly in every stages of this project.

It is a matter of great pleasure for us to express our sincere gratitude and appreciation to our beloved principal **Dr. N.K. NARAYANAN.**

We are really thankful to our Head of Department **Mrs. ARUNA A.S.** and rest of the teachers in department CS.

We are also thankful to our Project co-ordinator **Mrs. AKHILA M.** for her kind support and valuable guidance.

We are also thankful to our Project guide **Mrs. SHIBILI T.** for giving instruction to build up this project and valuable guidance.

We are also thankful to our fellow classmates and well-wishers for sharing their knowledge and suggestions.

We express our sincere gratitude to the GOD for giving us sound, health and all the support in completing this project.

ABSTRACT

The emergence of Mobile Cloud Computing (MCC) has created a situation where both the data storage and the data processing transpire outside of the mobile device. For storage and sharing of data we use a variety of mobile applications. Unfortunately, the way we store and share such personal data are often unencrypted and insecure. So in this paper, we introduce a secure way to store and share such data using a lightweight, fast, computationally efficient, easy to use protocol suitable for resource constrained devices. ROBE protocol is based on stream cipher and it uses an External Cloud Server for management of cryptographic tokens. We use a strong algorithm with less complexity to secure data. The security of cryptographic tokens is ensured at various levels using Advanced Encryption Standard (AES), Secure Hash Algorithm (SHA) and deceit methods. In ROBE, all data are shared securely between mobile and the server with mutual identity verification. Also various attacks are computationally in-feasible for our protocol.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATION

AES	:	ADVANCED ENCRYPTION STANDARD
MD	:	MOBILE DEVICES
MCC	:	MOBILE CLOUD COMPUTING
CSPRN	:	CRYPTOGRAPHICALLY SECURED RANDOM NUMBER
NHEP	:	NEW HYBRID ENCRYPTION PROTOCOL
ES	:	EXTERNAL CLOUD SERVER
DFD	:	DATA FLOW DIAGRAM
VM	:	VIRTUAL MACHINE
APK	:	ANDROID PACKAGES

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

Mobile device and its applications have revolutionized the way we store and share data. It is becoming a warehouse of user's personal information. Unluckily, most of these data are stored in an unencrypted format, prone to security threats. In our project, we implement a lightweight, computationally efficient protocol, called *ROBE*, for the mobile device.

The word "ROBE" means a cloth, which is used to get protection from the harmful climate. As well as our proposed protocol will provide protection from all types of attacks like man in the middle attack, boot force attack ect. ROBE is based on stream cipher and takes the help of an external server for the generation and distribution of cryptographically secure pseudo-random number (Cryptographic Token). In order to enhance the security of our protocol, we use the concept of symmetric key cryptography. In ROBE, the core encryption/decryption operation is performed within the mobile devices to secure data at its origin. The security of Cryptographic Token is ensured using deception method. We implement AES-128 bits for generating of first 128 bits of Cryptographic Token. In ROBE, all messages are exchanged securely between mobile and the server with mutual identity verification. We evaluate ROBE on Android smart phones and use cloud server for generating Cryptographic Token. Additionally, we perform attack analysis and show that various attacks are computationally infeasible for the proposed protocol.

Advancements in mobile technology, innovative applications and decreasing prices of smartphones, wearable computers and other Mobile devices (MD) have contributed significantly in increasing popularity of mobile devices in our modern lifestyle. Since, mobile devices are meant for personal usage, it is often used as a repository for storing user's personal information, such as user profile, passwords, bank account information and medical records. More significantly, the data is stored in a clear text format in a mobile device, which can be easily retrieved and lead to serious security complications. Security threats on mobile device can be

from various sources including malwares, third party applications, eavesdropping over wireless network, theft and lost devices. As a result, many companies do not allow employees to store corporate data in smartphones or use the corporate network through personal devices.

Mobile cloud computing (MCC) is an emerging research area focusing on supplementing the storage and computational requirement of mobile device by utilizing the cloud infrastructure. By interacting with cloud, mobile device can deliver various services to the user, such as healthcare mobile commerce and online education. Users can upload and store data (photos, medical records) from their mobile device to the cloud and can share them with others. In addition, mobile device can offload computation intensive tasks to the cloud to overcome its resources limitation and for saving battery. However, security is a major concern in mobile cloud computing, particularly for mobile applications sending unencrypted personal information over insecure wireless medium to the cloud. Data encryption is also required for protecting user's data against external and internal attacks within the cloud environment.

Encryption/decryption algorithms are commonly used for providing security to user's personal information. Encryption is a process of converting plaintext (PT) data into an incomprehensible code called ciphertext (CT) and a decryption algorithm is used for inverting the ciphertext to original plaintext. In this project, we focus on the encryption and decryption of files for the mobile device.

A stream cipher is a symmetric key cipher where plaintext digits are combined with a pseudorandom cipher digit stream (keystream). In a stream cipher, each plaintext digit is encrypted one at a time with the corresponding digit of the keystream, to give a digit of the ciphertext stream. Since encryption of each digit is dependent on the current state of the cipher, it is also known as state cipher. In practice, a digit is typically a bit and the combining operation an exclusive-or (XOR).The pseudorandom keystream is typically generated serially from a random seed value using digital shift registers. The seed value serves as the cryptographic key for decrypting the ciphertext stream and also same starting state (seed) must never be used twice. Stream ciphers represent a different approach to symmetric encryption from block ciphers.

Here we have an additional feature named Vault. It is a hidden folder which hide photos, videos and other files that we want to hide. They are stored as encrypted files. This can be decrypted only by the owned user. It has no storage limitations to hide the files. Vaulted files are

stored in the user device itself. No one can see this vaulted file other than the owned user. Users vaulted files will be completely secure even if the app is deleted from the device.

The design considerations of our ROBE protocol are as follows:

To design a lightweight encryption protocol for mobile device. We assume that a single file in any size for images and documents in txt, pdf, doc formats. The protocol must be able to handle such files on most mobile devices that are currently available in the market. The encryption/decryption operation must be performed in an acceptable time frame. The users must be able to control the encryption/decryption operation. This is important for establishing user's confidence on the system. All operations on the plaintext must be performed locally on the mobile device and the computations on external cloud server should not affect the security of the protocol. Finally, and most importantly, the protocol must be cryptographically secure.

1.2 OBJECTIVE

The aim of the project is to develop an android application providing lightweight, computationally efficient, highly secure protocol for quick file encryption in mobile devices so that users can safely stop or share them on the cloud or any other services. Based on stream cipher and it takes the help of a cloud server for the generation and distribution of cryptographic tokens and for the authorization of recipients.

1.3 SUMMERY

We present a stream cipher based encryption/decryption protocol for mobile devices. We use a MCC environment for implementing the protocol. This project addresses the challenges of insecure wireless media using Cryptographic Token and securing the message communication. For the evaluation of protocol we developed an android application and generated Cryptographic Token on ES.

CHAPTER 2

SYSTEM ANALYSIS

System analysis is a general term that refers to an orderly, structured process for identify and solving a problem. The system analysis process is calling the life cycle methodology, since it relates to four significant phases in the life cycle of all business information system: study, design, development and operation. The definition of system analysis includes not only the process but also the process of putting together to form a new system. A system analyst is an individual who performs system analysis during any, or all, of the life cycle phases of a business information system. The system analyst not analyses business information system problems, but also synthesizes new to solve those problem or to meet other information needs.

2.1 EXISTING SYSTEM

Mobile cloud computing (MCC) is an emerging research area focusing on supplementing the storage and computational requirement of mobile devices (MD) by utilizing the cloud infrastructure. By interacting with cloud, mobile device can deliver various services to the user, such as healthcare, mobile commerce and online education. Users can upload and store data (photos, medical records) from their mobile device to the cloud and can share them with others.

Based on recent smartphone trends, SP (service providers) such as Facebook and Twitter, are offering their applications to more mobile Internet application users. This has caused traffic overload problems for some network companies that require excessive network maintenance, creating an imbalance between profit and investment. Excessive traffic requires network companies to secure MO (mobile operators) availability, mobile network bandwidth, and minimize information processing by users.

Cloud is extremely powerful to perform computations while computing ability of mobile devices has a limit so many issues occur to show how to balance the differences between these two. So there are some issues in implementing cloud computing for mobile. These issues can be related to limited resources, related to network, related to security of mobile users and clouds.

Most of mobile devices have almost same functionalities like a desktop computer. So mobile devices also have to face a number of problems related to security and privacy. To overcome this problem threat detection services are now performed at clouds but this also has to face a lot of challenges. Some security issues are like device security, privacy of mobile user and securing data on cloud etc. There are so many security threats like viruses, hacking, Trojans in mobile devices etc.

2.1.1 LITERATURE SURVEY

Nowadays new challenges and opportunities are introduced in the field of security and privacy. That is related to proliferation of mobile device and maturing of cloud computing technologies. Many works continue to boost the security and privacy of the mobile device storage and processing resources offered by cloud computing. Security architecture and encryption schemes for mobile cloud computing have been proposed by others working to find an effective way to secure data stored on an external server while at the same time limiting the amount of resources such as memory foot print and CPU storage.

A. Symmetric and Asymmetric Encryption

There exist much conflict over whether using symmetric or asymmetric encryption schemes are best for data encryption. The resource cost of symmetric encryption allows fast encryption of data. The asymmetric encryption needs more resources to attain same level of security as symmetric encryption. The protocol introduced by Subasree and Sakthivel in [7] use ECC to encrypt data and dual RSA to encrypt the message hash used to ensure integrity. But both these algorithms are asymmetric algorithm, therefore it requires more resources and also it is significantly slow. Kader et al proposed a New Hybrid Encryption Protocol (NHEP). In hybrid protocols includes both symmetric and asymmetric encryption algorithms. In NEPH protocol, block of data are divided in to half. The first half is encrypted using AES (symmetric) with ECC (asymmetric), and the second half is encrypted using Blowfish (symmetric), RSA(asymmetric). For both halves, the respective asymmetric encryption are used to encrypt secret key, which is used by symmetric algorithm to encrypt the data. NHEP has a significant edge over other hybrid encryption algorithm.

B. FACOR

Flexible Access control with Outsourceable Revocation in Mobile Clouds (FACOR) is a key mechanisms to secure out sourced data in mobile clouds. Outsourcing computation is useful tool

to allow computational weak clients to deliver time consuming operations to the third parties. Revocation is cryptographic approach to deprive users to access right to out sourced data in mobile cloud computing. Here they consider limited computing capability of mobile devices and manage to outsource the time consuming operations of both encryption and decryption and complicated revocation operation on mobile devices. In this system there are six polynomial time algorithms. The security of the system is based on q-Decisional Multi- Exponent Bilinear Difie – Hellman Assumption [8]. Therefore the shared key is an asymmetric key, so it is inherently slow and impractical for bulk encryption.

C. Stream Cipher based Encryption Protocol

Our proposed protocol is based on stream cipher based symmetric algorithm. We use AES based algorithm for generating Cryptographic Token (Cryptographically Secured Random Number) in the cloud. The formation of cipher text is done by simple XORing operation between Plain Text and modified Cryptographic Token. It is quick and efficient process. The cloud sends secured Cryptographic Token to users. The user will encrypt or decrypt the Plain text by using this Cryptographic Token. It consumes very less power and resources for transfer, encryption and decryption of data.

2.2 PROPOSED SYSTEM

A protocol for encrypting and decrypting files within the mobile devices in a mobile cloud environment, referred as ROBE. Our goal is to secure personal information stored in mobile device (images, pdf, doc),with no size limit. The ROBE protocol is based on stream cipher and takes the help of an external cloud server (ES) for generating the key-stream or a cryptographically secure pseudo-random number (Cryptographic Token). The advantage of using stream cipher as the basis of our protocol, is that it is less computation intensive compared to block cipher and can easily be handled by existing mobile devices. Stream cipher is a cryptographically secure encryption algorithm, used in various protocols (WPA, TLS), applications (Internet Explorer, Google Chrome, Firefox) and in communication standards (GSM, 3GPP, LTE).

Vault is another feature that we proposed here. It is a hidden folder which hide all the needed file as encrypted, without any size limit. This encrypted file can be only decrypted by the owned user.

Advantages:

- We proposes a light weight encryption protocol which is able to handle large sized files found on Mobile Devices.
- The encryption or decryption algorithm must be performed within an acceptable time frame.
- The control of encryption or decryption operation must rely with the users. This is necessary to establish user's confidence on the system.
- All PT operations must be performed locally on the Mobile Devices itself and the computations on External Server should be in such a way that doesn't affect protocol security.
- Most importantly the protocol must be cryptographically secure.

2.3 FEASIBILITY STUDY

Feasibility is defined as the practical extent to which a project can be performed successfully. The objective of feasibility study is to establish the reasons for developing the software that is acceptable to the users, adaptable to changes and conformable to the established standards. Various types of feasibility that are commonly considered include:

- Economic feasibility
- Operational feasibility
- Technical feasibility

2.3.1 ECONOMIC FEASIBILITY

A cost evaluation is weighed against the ultimate income or benefit derived from the developed system or product. When compared to the advantages obtained from implementing the system its cost is affordable. Also the system is designed to meet the modification required in the future. Most of the required modification can be done without network. Since cost input for the software is almost nil the output of the software is always a profit. Hence software is economic feasible.

The proposed system aims at processing the data efficiently, thus saving time and money. Though the system requires a cloud for storing encrypted data since it provides very fast and accurate results, the proposed system is economically feasible.

2.3.2 OPERATIONAL FEASIBILITY

The proposed system offers:

- User friendliness
- Higher Speed
- Light weight computation
- Less chance of errors
- Easier access of data

The operations on the application are very simple. The new proposed system is very much useful to the users and there for it will accept broad audience from around the world. User can manage this system easily. So the proposed system is operationally feasible.

2.3.3 TECHNICAL FEASIBILITY

A study of function, performance and constraints may improve the ability to create an acceptable system. Technical feasibility is frequently the most difficult area to achieve at the stage of product engineering process.

This deals with the hardware as well as software requirements. The scope was whether the work for the project is done with the current equipment's and software technologies has to examine in the feasibility study. The outcome was found to be positive.

In the proposed system data can be easily stored and managed through mobile cloud computing using ROBE protocol. The reports and results for various queries can be generated easily. Therefore the system is technically feasible.

CHAPTER 3

SYSTEM SPECIFICATION

Hardware and software requirements for the installation and smooth functioning of this product could be configured based on the requirements needed by the component of the operating environment that works as front end and back end. Here we suggest minimum configuration for the both hardware and software components.

Working off with this software is requirements concrete on system environments. It includes these phase.

- Hardware Requirements
- Software Requirements

3.1 HARWARE REQUIREMENTS

The selection of hardware is very important in the existence and proper working of any software. When selecting hardware, the size and requirements are also important.

- Processor : Intel Core i3 processor
- RAM : 4GB
- Hard Disk Drive : 50GB

3.2 SOFTWARE REQUIREMENTS

- OS : Windows 10
- IDE : Android Studio 3.0
- SQL Server : XAMPP
- Front End : XML
- Back End : PHP, JAVA, MySQL, SQLite

CHAPTER 4
SOFTWARE DESCRIPTION

A native mobile app is developed for a particular mobile platform and is written in a language that is supported by the operating system of that platform. Since a native app can fully leverage a device's hardware and functionality, it offers the best user experience. Android developers utilize different tools to create their apps. An Android app is made up of two parts: the front end and the back end. The front end is the visual part of the app that the user interacts with, and the back end, which contains all the code that drives the app.

4.1 FRONT END

Android app developers use XML, a mark-up language, to set up the layout of their apps.

4.1.1 XML

The Android framework gives you the flexibility to declare your application's default layouts in XML, including the screen elements that will appear in them and their properties. You could then add code in your application that would modify the state of the screen objects, including those declared in XML, at run time.

The advantage to declaring your UI in XML is that it enables you to better separate the presentation of your application from the code that controls its behavior. Your UI descriptions are external to your application code, which means that you can modify or adapt it without having to modify your source code and recompile. For example, you can create XML layouts for different screen orientations, different device screen sizes, and different languages. Additionally, declaring the layout in XML makes it easier to visualize the structure of your UI, so it's easier to debug problems. As such, this document focuses on teaching you how to declare your layout in XML. If you're interested in instantiating View objects at runtime, refer to the view group and view class references.

In general, the XML vocabulary for declaring UI elements closely follows the structure and naming of the classes and methods, where element names correspond to class names and attribute names correspond to methods. In fact, the correspondence is often so direct that you can guess what XML attribute corresponds to a class method, or guess what class corresponds to a

given XML element. Android uses several XML files to create the app's front end. There is at least one XML layout file for each activity (or several if you are supporting multiple device sizes), as well as layout files for custom views. Using Android's XML vocabulary, you can quickly design UI layouts and the screen elements they contain, in the same way you create web pages in HTML with a series of nested elements.

Features

- XML separates data from HTML. If you need to display dynamic data in your HTML document, it will take a lot of work to edit the HTML each time the data changes.
- With XML, data can be stored in separate XML files. This way you can focus on using HTML/CSS for display and layout, and be sure that changes in the underlying data will not require any changes to the HTML.
- With a few lines of JavaScript code, you can read an external XML file and update the data content of your web page.
- XML simplifies data sharing. In the real world, computer systems and databases contain data in incompatible formats.
- XML data is stored in plain text format. This provides a software- and hardware-independent way of storing data. This makes it much easier to create data that can be shared by different applications.
- XML simplifies data transport. One of the most time-consuming challenges for developers is to exchange data between incompatible systems over the Internet.
- Exchanging data as XML greatly reduces this complexity, since the data can be read by different incompatible applications.
- XML simplifies Platform change. Upgrading to new systems (hardware or software platforms), is always time consuming. Large amounts of data must be converted and incompatible data is often lost.
- XML data is stored in text format. This makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data.
- XML increases data availability. Different applications can access your data, not only in HTML pages, but also from XML data sources.

- With XML, your data can be available to all kinds of "reading machines" (Handheld computers, voice machines, news feeds, etc.), and make it more available for blind people, or people with other disabilities.
- XML can be used to create new internet languages.

4.2 BACK END

The backend is where you find the implementation of system logic and data handling as well as integration with third party systems. To create a server that communicates purposefully and efficiently with the front end, many technologies may be used.

4.2.1 JAVA

The platform for app development in Android is Java. There are multiple reasons for this such as; Java is a commonly used language and many programmers know it, it can run on a virtual machine (VM) so no need to recompile for different phones, better security, many development tools available for Java, and Java is a known industry language with most phones compatible with it.

Features

- “Write once, run anywhere” is Java’s motto and that compatibility remains a major selling point. While most Android applications are written in Java, there is no Java Virtual Machine in the platform and Java byte code is not executed.
- Java classes are compiled into Dalvik executables, bundled as Android Package (APK) along with other resources and run on using Android Runtime or in Dalvik in older versions, a specialized virtual machine designed specifically for Android and optimized for battery-powered mobile devices with limited memory and CPU.
- The class files are compiled into Dalvik Executable (DEX) format, and bundled as Android Package (APK) along with other resources.
- Java is one of the few languages that is close to 100% object-oriented, with all the benefits of object-oriented programming: easy development, modular software, flexibility, and extensibility.
-

4.2.2 PHP

PHP started out as a small open source project that evolved as more and more people found out how useful it was.

- PHP is a recursive acronym for "PHP: Hypertext Preprocessor"
- PHP is server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire-commerce sites.
- It is integrated with a number of popular databases, including MySQL. PostgresSQL, Oracle, Sybase, Informix and Microsoft SQL Server.
- PHP is a pleasingly zippy in its execution, especially when complied as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge results set in record-setting time.
- PHP supports a large number of major protocols such as POP3,IMAP,and LDAP.PHP4 added support for Java and distributed object architectures (COM and CORBA),making n-tier development a possibility for the first time.
- PHP is forgiving :PHP language tries to be as forgiving as possible

Features

Common uses of PHP:

- PHP performs system functions i.e. from files on system it can create, open, read, write and close them.
- PHP can handle forms i.e. gather data from files, save data to a file, through email can send data, return data to the user.
- You add, delete, modify elements within your database thru PHP
- Access cookies variables and set cookies.
- Using PHP, we can restrict users to access some pages of your website.
- It can encrypt data

Characteristics of PHP

- Simplicity
- Efficiency

- Security
- Flexibility
- Familiarity

4.2.3 MYSQL

A back-end database is a database that is accessed by users indirectly through an external application rather than by application programming stored within the database itself or by low level manipulation of the data (e.g. through SQL commands). A back-end database stores data but does not include end-user application elements such as stored queries, forms, macros or reports.

Features

MySQL Database:

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. MySQL is developed, marketed and supported by MySQL AB, which is a Swedish company, MySQL is becoming so popular because of many good reasons:

- MySQL is released under a open-source license. So you have nothing to pay to use it.
- MySQL is a very powerful program in its own right .It handles a large subset of the functionality of the most expensive and powerful database packages.
- MySQL uses the standard form of the well-known SQL data languages.
- MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
- MySQL works very quickly and works well even with data large data sets.
- MySQL is very friendly to PHP, the most appreciated language for web development.

4.2.4 XAMPP SERVER

XAMPP is a free and open source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. XAMPP server is a web development environment. It allows us to create web

applications with Apache 2, PHP and a MYSQL database. Alongside, PHPMY Admin allows us to manage easily databases.

4.2.5 SQLITE

SQLite is an open source SQL database that stores data to a text file on a device. Android comes in with built in SQLite database implementation. SQLite supports all the relational database features. In order to access this database, you don't need to establish any kind of connections for it like JDBC, ODBC etc.

Features

- Due to the server-less design, SQLite applications require less configuration than client-server databases. SQLite is called 'zero-conf' because, access control is handled by means of file system permissions given to the database file itself.
- SQLite uses an unusual type system for an SQL-compatible DBMS; instead of assigning a type to a column as in most SQL database systems, types are assigned to individual values; in language terms it is dynamically typed.
- Several computer processes or threads may access the same database concurrently. Several read accesses can be satisfied in parallel. A write access can only be satisfied if no other accesses are currently being serviced
- In 2015, with the json1 extension and new subtype interfaces, SQLite version 3.9 introduced JSON content managing.
- In android, the main package is android.database.sqlite that contains the classes to manage your own databases
- In android, in order to create a database you just need to call this method "openOrCreateDatabase" with your database name and mode as a parameter. It returns an instance of SQLite database which you have to receive in your own object.

CHAPTER 5
PROJECT DESCRIPTION

5.1 PROBLEM DEFINITION

Security is a major concern in Mobile Cloud Computing, particularly for mobile applications sending unencrypted personal information over insecure wireless medium to the cloud. In order to ensure the security of Mobile Cloud Computing there is a need for a lightweight encryption protocol for resource constrained mobile devices. The protocol must be able to handle such files on most mobile devices that are currently available in the market. The encryption/decryption operation must be performed in an acceptable time frame. The users must be able to control the encryption/ decryption operation. This is important for establishing user's confidence on the system. All operations on the plain text must be performed locally on the mobile devices and the computations on external server should not affect the security of the protocol. Finally, and most importantly, the protocol must be cryptographically secure.

5.2 MODULE DESCRIPTION

There are five modules in the system design for the project. These five modules are:

- Login and Registration Module
- Encryption Module
- Decryption Module
- Cryptographic Token Generation Module
- Vault module

5.2.1 LOGIN AND REGISTRATION MODULE

The registration can be done in smartphones or tablets interested in sending a file or message to another device. The device should be registered with the external cloud server. During registration the user has to provide name, email id and password. User can login application using email id and password. The communications between sender and external cloud server can take place via any wireless communication media such as Wi-Fi, 3G, 4G, UMTS, LTE. For secure communication, we use a pre-shared key between the External Cloud

Server and Mobile Device, defined during the registration of the Mobile Device and also a common one way hashed function.

5.2.2 ENCRYPTION MODULE

This module helps us to move to the cloud and achieve secure Multi-Tenancy in the cloud. When we own the key we can easily decommission/depravation. Separating data from key services can prevent service providers from accessing or accidently exposing the data. Encryption helps us to meet regulations and gives safe Harbour from Breach notification. Also this module gives service providers a competitive edge.

In encryption module the user sends a request message which consist of user id, email and file name to External Server (ES). The cloud server return a modified Cryptographic Token with a shared key. Using this shared key the modified Cryptographic Token will be XORed to get the original Cryptographic Token. The encryption can be done by XORing the corresponding file with Cryptographic Token. The encrypted file will be stored in the mobile device itself. Then the user can send or decrypt the encrypted file according to his need. The encrypted file will be appended with a user id and filename of the sender in the last line of encrypted file.

5.2.3 DECRYPTION MODULE

In this module issued user can decrypt the encrypted file send by another user. This decryption can be done by sending a request along with uid of sender and filename to the external server. As a result corresponding Cryptographic Token of the file will be return to the user. Using this Cryptographic Token the encrypted file can be decrypted by performing an XORing operation.

Decryption is also performed in vault file too. In this folder the file will be stored as encrypted. To decrypt this file user send a request along with file's path name.

5.2.4 CRYPTOGRAPHICAL TOKEN GENERATION MODULE

External cloud server module is used for the generation and distribution of unpredictable Cryptographic Token. The external cloud server is specifically configured according to the requirement of the application and the workload. The external cloud server module receives the

request for Cryptographic Token from the sender specifying unique user-id, file-name and Cryptographic Token –size. On receiving this request, the external cloud server module produces Cryptographic Token using the algorithm and returns the Cryptographic Token back to the sender. The cloud server stores the user-id, file-name, sequence number, seed and key in it's database. For more security, the external cloud server uses encryption before storing the data in the database. In the decryption phase, the receiver module sends a request to the external cloud server specifying the user-id and file-name for the Cryptographic Token.

The external cloud server module consults its database with these parameters, retrieves the corresponding sequence number and key from its database, generates the same Cryptographic Token and forwards it to the receiver. In external cloud server module, we implement Advanced Encryption Standard (AES) for generating of 128 bits of a Cryptographic Token. The parameters passed to the AES encryption algorithm, are seed/key and sequence number. The sequence number and the key are integers that can be user defined or can be selected randomly. To secure the transmission, the external cloud server module modifies the generated Cryptographic Token by XORing it with a pre-shared secret key, which is defined at the time of registration of devices with external cloud server.

5.2.5 VAULT MODULE

Vault is a hidden folder which hide photos, videos and other files that we want to hide. They are stored as encrypted files. This can be decrypted only by the owned user. User can view the vault file only by logging into that. The encrypted file will be removed from the user device until it gets decrypted. The encrypted file will be appended with the path of that file. While decrypting it the appended path will be removed from the file and after decryption the file will be removed from vault and stored in the device as in the path.

5.3 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a diagram that describes the flow of data and the processes that change data throughout a system. It's a structured analysis and design tool that can be used for flowcharting in place of or in association with information. Oriented and process oriented system flowcharts. When analysts prepare the Data Flow Diagram, they specify the user

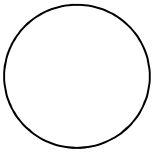
needs at a level of detail that virtually determines the information flow into and out of the system and the required data resources. This network is constructed by using a set of symbols that do not imply physical implementations. The Data Flow Diagram reviews the current physical system, prepares input and output specification, specifies the implementation plan etc.

Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

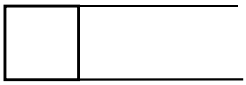
5.3.1 DATA FLOW DIAGRAM SYMBOLS:

- Source or Destination of data

- Data Flow



- Process



- Storage

Level 0 DFD

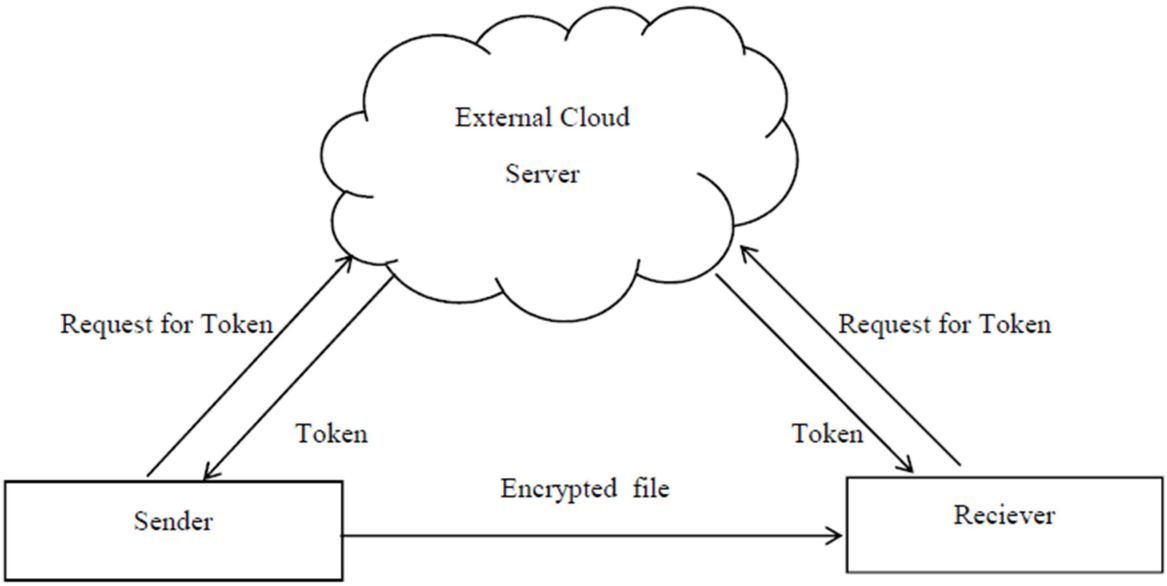


Figure 5. 1 Level 0 DFD

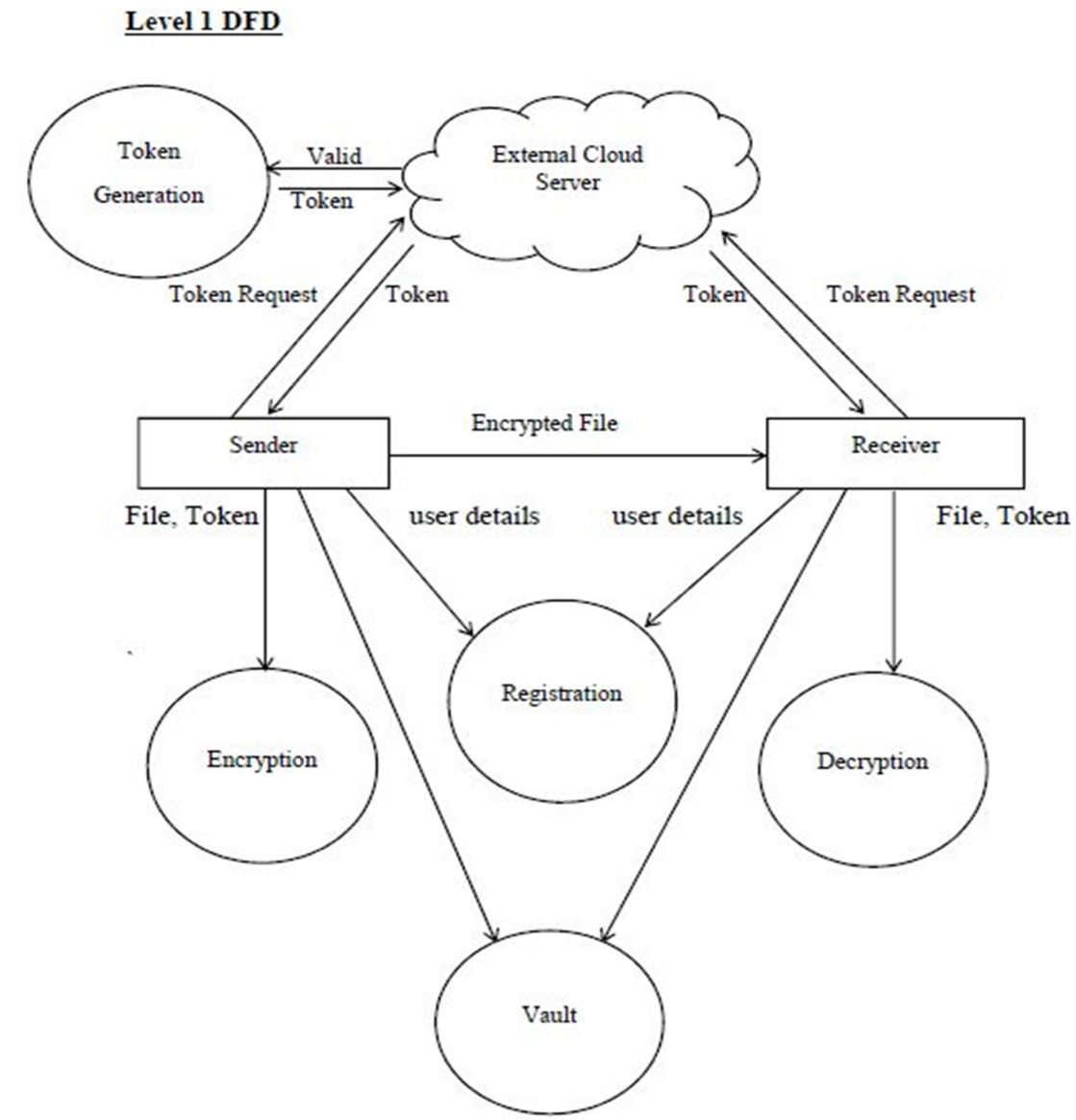


Figure 5.2 Level 1 DFD

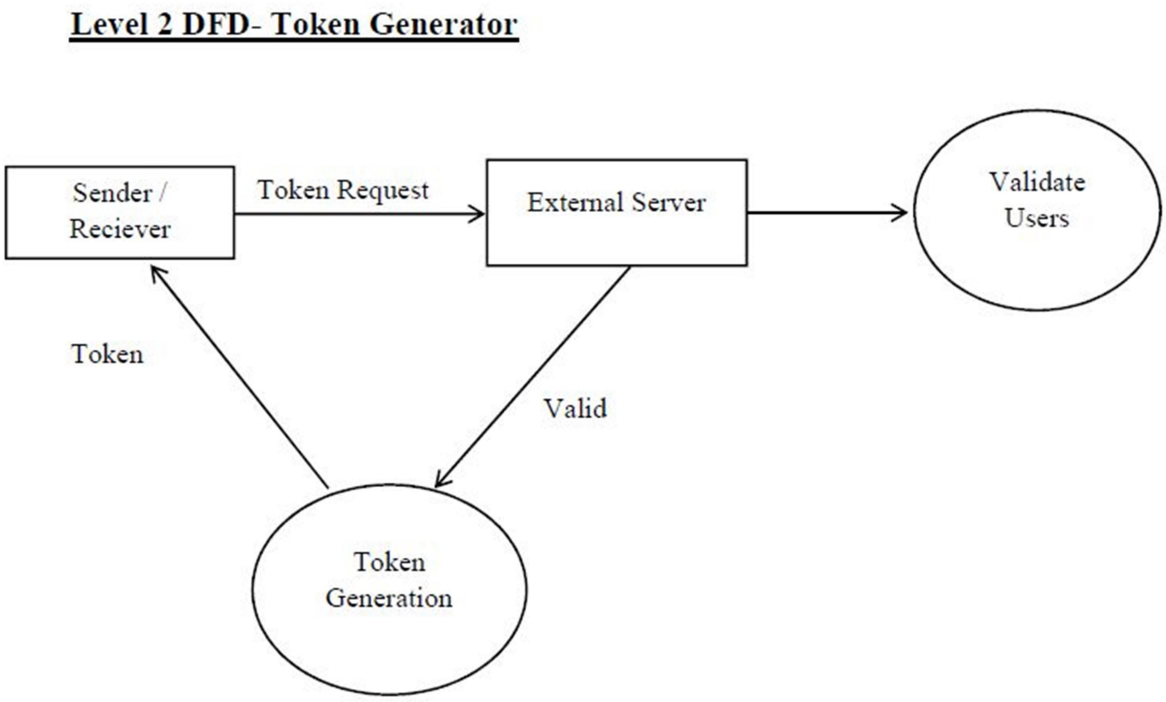


Figure5.3 Level 2 DFD-Token generator

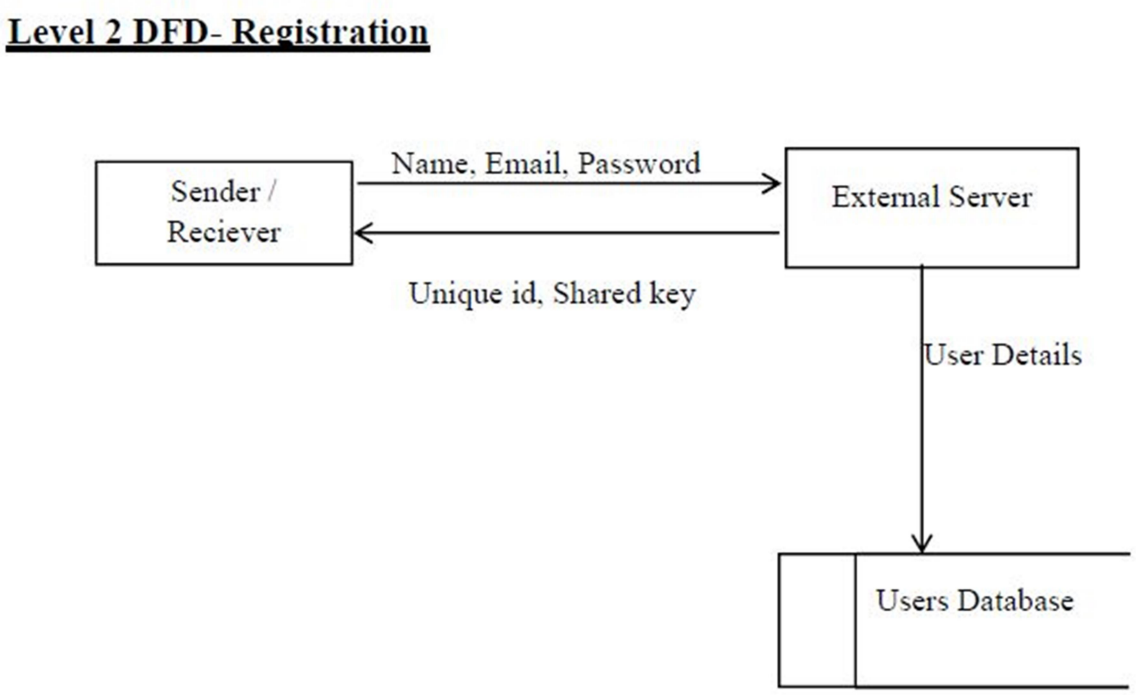


Figure 5.4 Level 2 DFD-Registration

Level 2 DFD- Vault

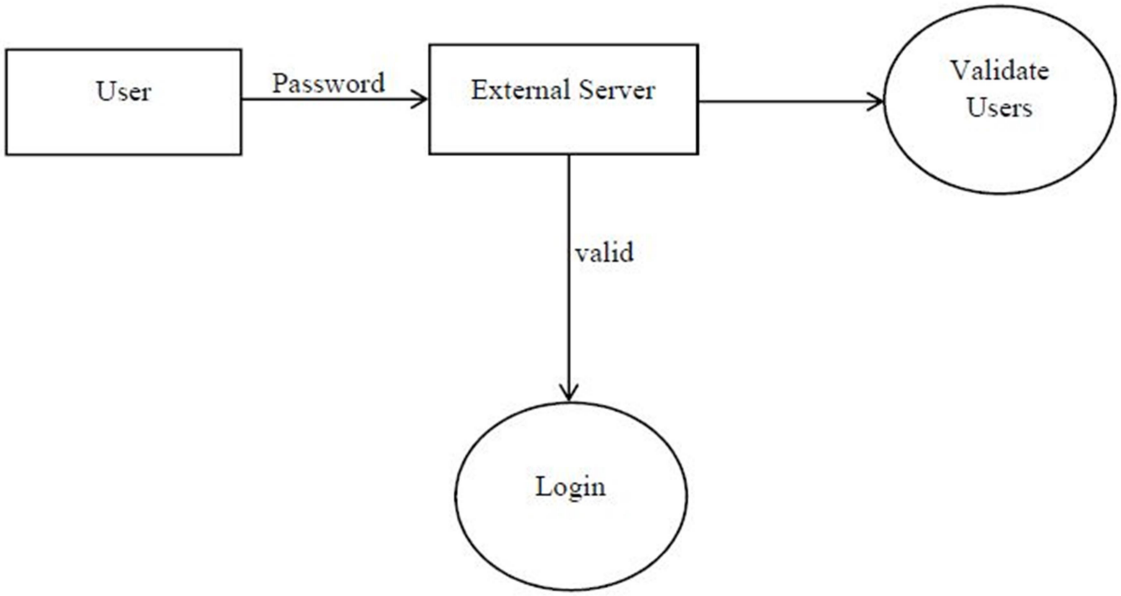


Figure 5.5Level 2 DFD Vault

5.4 ER DIAGRAM

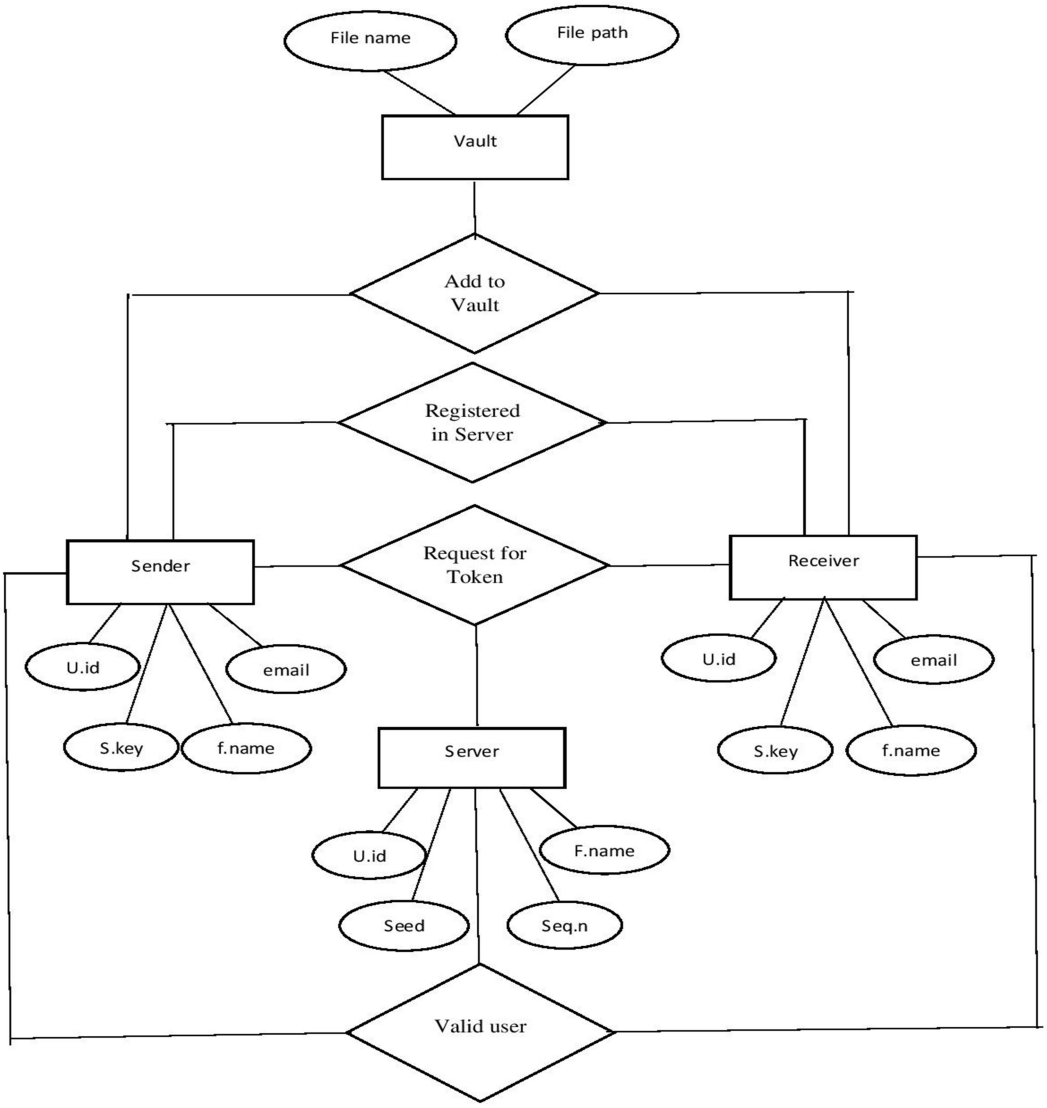


Figure 5.6 ER Diagram

5.5 DATABASE DESIGN

Database design is the process of producing a detailed data model of database. The objective of database design is to provide auxiliary storage and to contribute to the overall efficiency of the computer program component. One auxiliary storage medium must provide efficient access to the data. The general theme of database to handle information as an integrated whole.

This data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity. However, the term database design could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application within the database management system (DBMS).

The process of doing database design generally consists of a number of steps which will be carried out by the database designer. Usually, the designer must:

- Determine the data to be stored in the database.
- Determine the relationships between the different data elements.
- Superimpose a logical structure upon the data on the basis of these relationships.

5.5.1 TABLES

The overall objective in the development of the database technology has been to treat data as an organization resource and as integrated whole. DBMS allow data to be protected and organized separately from other resource. Database is an integrated collection of data. The most significant data has been by the programmers is data as stored on the direct access storage device. This is the difference between logical and physical design.

The database contains tables where each table corresponds to one particular type of information. Each piece of information in the table is called a field or column. A table also

contains records which is a set of fields. All record in a table have the same set of fields with different information. There are primary key fields that uniquely identify a record in the table.

1. Table Users

Column Name	Data Type	Allow null
id	int(11)	No
unique_id	varchar(23)	No
name	varchar(50)	No
email	varchar(100)	No
encrypted_password	varchar(80)	No
salt	varchar(10)	No
shared_key	varchar(50)	No
created_at	datetime	No
updated_at	datetime	No

2. Table Tokens

Column Name	Data Type	Allow null
id	int(11)	No
unique_id	varchar(23)	No
email	varchar(100)	No
Fn	varchar(100)	No
Seed	varchar(80)	No
Sequence	varchar(80)	No
created_at	datetime	No

3.Table Logged Users

Column Name	Data Type	Allow null
key_id	integer primary key	No
key_name	text	No
key_email	text unique	No
key_uid	text	No
key_shared_key	text	No
key_created_at	text	No

4.Table Recipient List

Column Name	Data Type	Allow null
id	int(10)	No
unique_id	varchar(23)	No
fn	varchar(100)	No
recipient	varchar(100)	No
created_at	datetime	No

CHAPTER 6

SYSTEM TESTING

6.1 TEST PLAN

The Software Test Plan describes plans for qualification testing of Computer Software configuration Items and Software systems. It describes the Software test environment to be used for the testing, identifies the test to performed, and provide schedules for test activities.

6.2 TESTING STRATEGY

The overall strategy for Software testing is described in the following section. We will use four different methods to test our Software.

6.2.1 UNIT TESTING

In unit testing, the analyst tests the programs making up a system. This is also called program testing. The software units that make up the system are modules and routines, which are assembled and integrated to perform a specific function in a large system. Many modules at different levels are needed. Unit testing, independent of one another, focuses on modules to locate error. This enables the test to detect error in coding and logic that are contained within that module alone. Those resulting from the interaction between modules are initially avoided. Unit Test comprises the test of performed prior to integration of the unit into the entire project. Four categories of test are performed on each unit.

- **Functional Test**

  The code is exercised with normal input values for which the expected results are shown, as well as boundary values (Minimum values, Maximum values) and values on and just outside the functional boundaries and special values such as logically related inputs.

- **Performance Test**

  Performance test is done to determine the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization by the program unit. Some time is taken initially to link to the SQL.

- **Stress Test**

  Stress Test has been done to intentionally break the unit. This helps in the learning about the strength and limitations of the program by examining the manner in which a program units breaks.

- **Structure Test**

  Structure test are done to internal logic of a program and traversing particular exercise, deriving test data to exercise those paths, determining the test coverage criteria to be used.

6.2.2 ACCEPTANCE TESTING

After the developers complete the system testing successfully acceptance testing is done at the customer end. It is the customer or the end user who knows designs the test cases. In this type of testing emphasis is on the usability of the product. Alpha testing is done when the software is made operational for the first time to be tested by the users at developer's site. Hence it is possible that it will involve making lot of changes to program code. Beta testing follows alpha testing but now the testing is done at the customer's Site that Validate the product after using it for few days. At this stage few changes as compared to alpha testing would make to the product.

6.2.3 TEST CASES

The evaluation of test cases is done through test case review. And for any review, a formal document or work product is needed. This is the primary reason for having the test case specification in the form of document. The test case specification document is reviewed, using a formal review process, to make sure that the test cases are consistent with the policy specified in the plan, satisfies the chosen criteria, and in general cover the various aspect of the unit to be tested.

For this process, the reason for selecting the test case and the expected output are also given in the test case specification document. By looking at the conditions being tested by the test cases, the reviewer can check if all the important conditions are being tested. As conditions can also be based on the output.

CHAPTER 7
CONCLUSION

In this project, we implemented a highly secure, light-weight, stream cipher based encryption/decryption protocol for the mobile devices. We developed application for android smartphones and used the cloud server for placing the Cryptographic Token generator as external server. The protocol is designed for the Mobile Cloud Computing environment. We handle the challenges of insecure wireless media by modifying the Cryptographic Token and securing the message communication. We found that ROBE protocol can resist various security challenges like brute force attack, man in the middle attack and Impersonation attacks. In addition, we secured the messages exchanged between mobile devices and the external cloud server. We also proved that the performance of the protocol on different mobile devices are highly efficient.

APPENDIX 1
SCREEN SHOTS

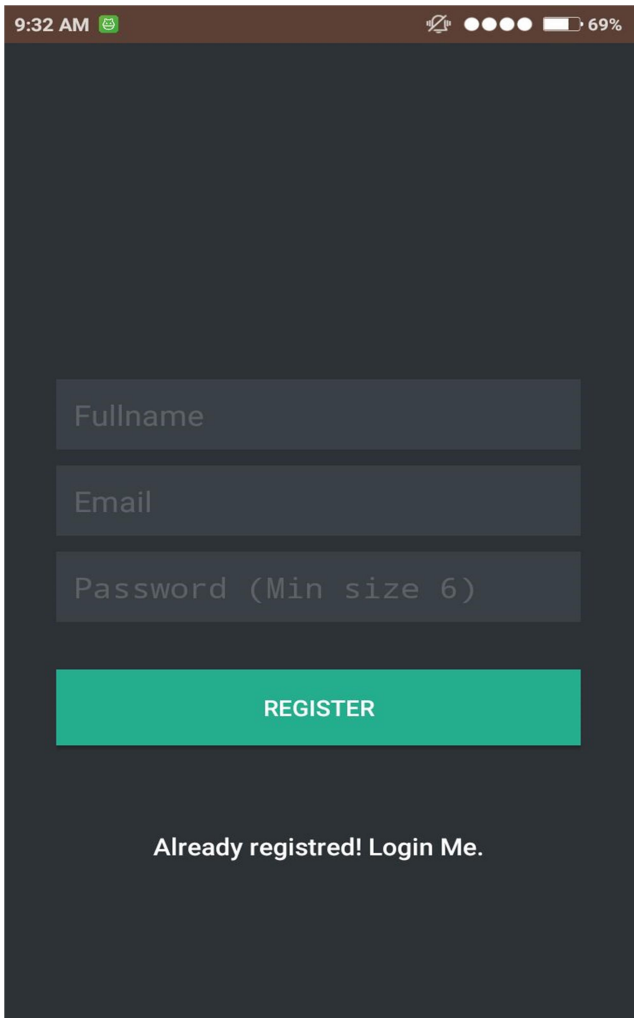


Figure 5.6.1 Registration screen

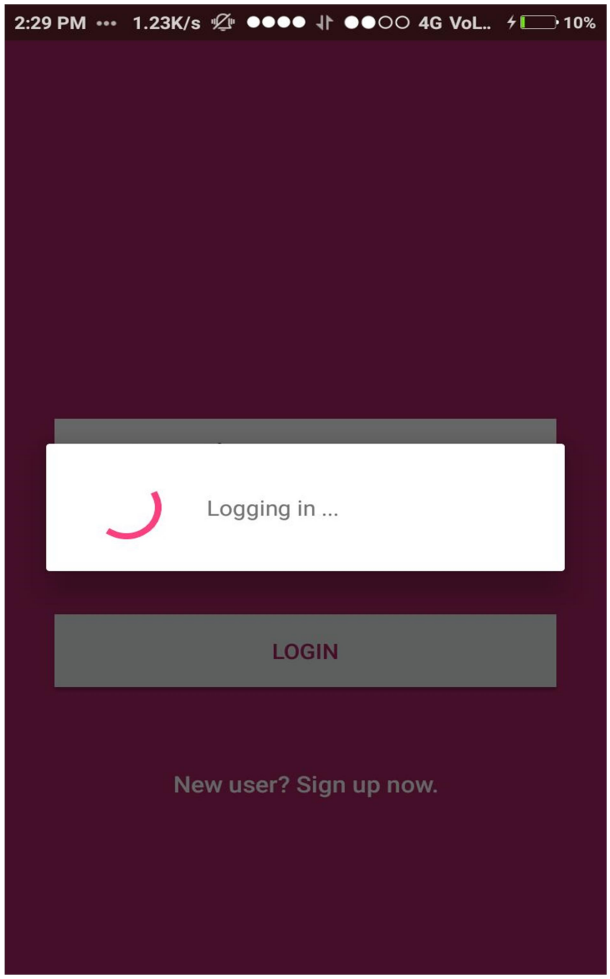


Figure 5.6.2 Login Screen

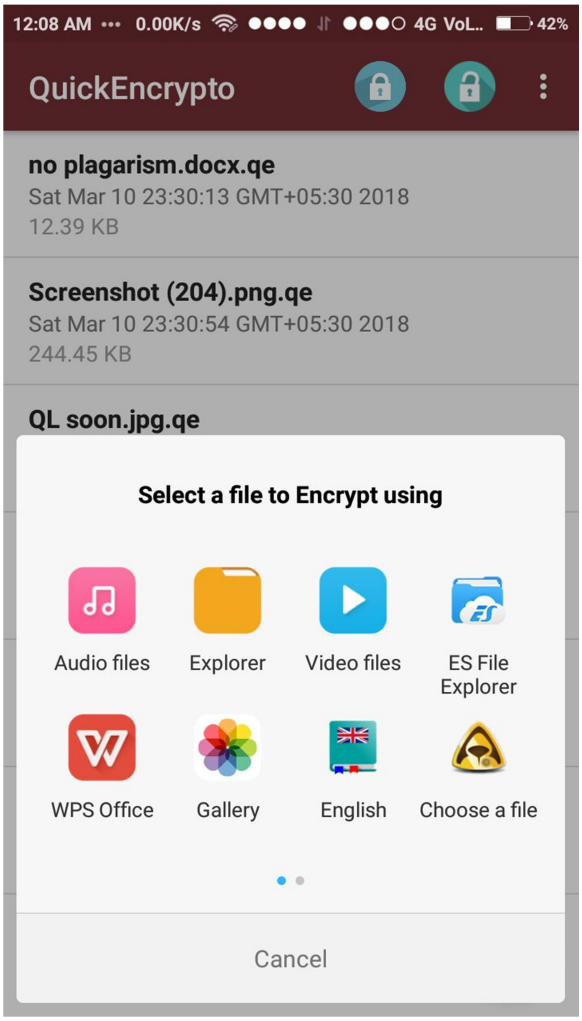


Figure 5.6.3 Choosing file for encryption

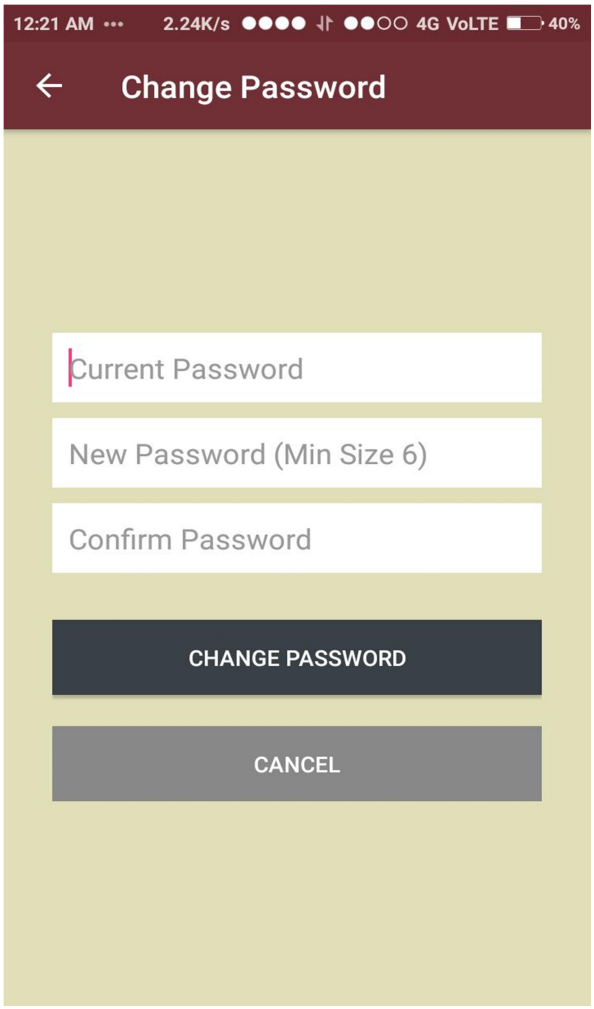


Figure 5.6.4 Changing Password

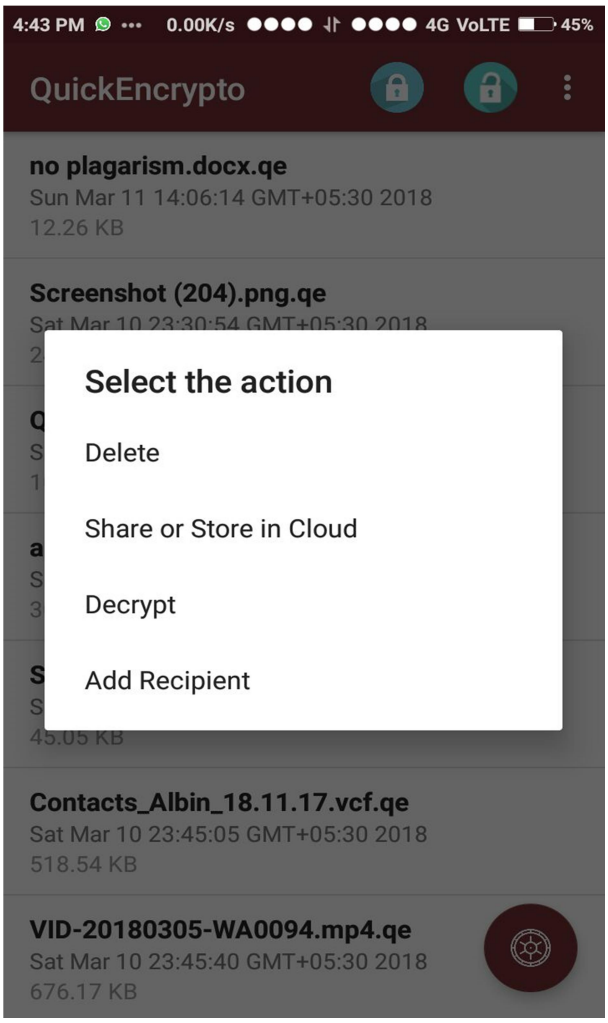


Figure 5.6.5 Options for encrypted file

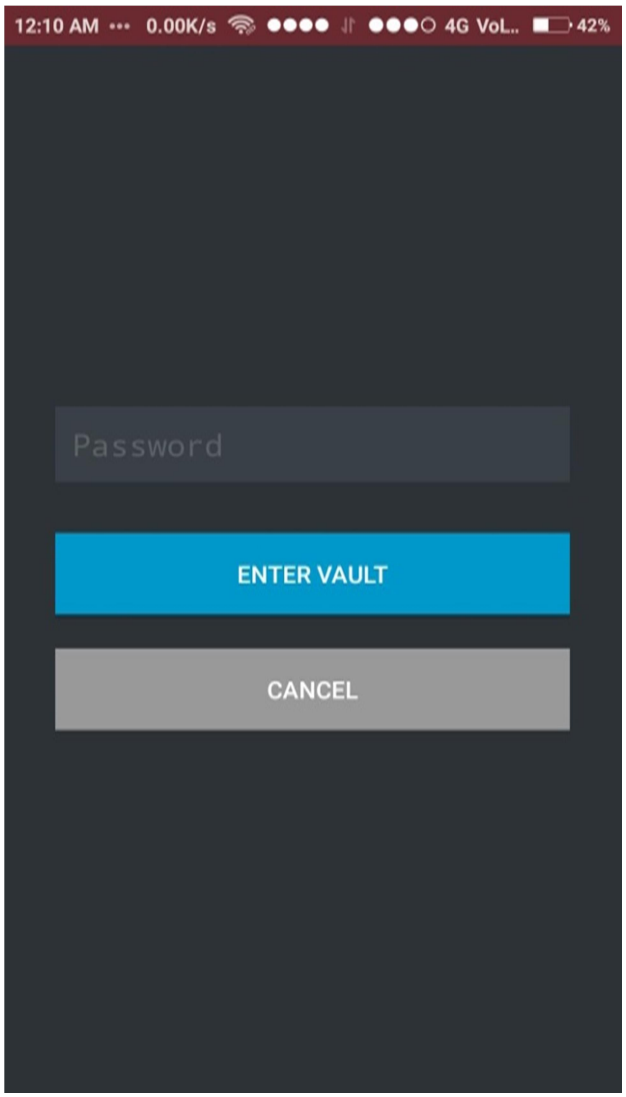


Figure5.6.6 Enter in to Vault

APPENDIX II

SOURCE CODE

```
import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Bundle;
import android.support.v7.app.AlertDialog;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.android.volley.Request.Method;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.cev.albin.quickencrypto.R;
import com.cev.albin.quickencrypto.app.AppConfig;
import com.cev.albin.quickencrypto.app.AppController;
import com.cev.albin.quickencrypto.helper.SQLiteHandler;
import com.cev.albin.quickencrypto.helper.SessionManager;

import org.json.JSONException;
import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;

public class RegisterActivity extends Activity {
    private static final String TAG = RegisterActivity.class.getSimpleName();
```

```
    private Button btnRegister;
    private Button btnLinkToLogin;
    private EditText inputFullName;
    private EditText inputEmail;
    private EditText inputPassword;
    private ProgressDialog pDialog;
    private SessionManager session;
    private SQLiteHandler db;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);

        inputFullName = (EditText) findViewById(R.id.name);
        inputEmail = (EditText) findViewById(R.id.email);
        inputPassword = (EditText) findViewById(R.id.password);
        btnRegister = (Button) findViewById(R.id.btnRegister);
        btnLinkToLogin = (Button) findViewById(R.id.btnLinkToLoginScreen);

        // Progress dialog
        pDialog = new ProgressDialog(this);
        pDialog.setCancelable(false);

        // Session manager
        session = new SessionManager(getApplicationContext());

        // SQLite database handler
        db = new SQLiteHandler(getApplicationContext());

        // Check if user is already logged in or not
        if (session.isLoggedIn()) {
            // User is already logged in. Take him to main activity
            Intent intent = new Intent(RegisterActivity.this,
                    MainActivity.class);
            startActivity(intent);
            finish();
        }
```

```
        // Register Button Click event
        btnRegister.setOnClickListener(new View.OnClickListener() {
            public void onClick(View view) {
                boolean isConnected = isNetworkConnected();
                if(isConnected) {

                    String name = inputFullName.getText().toString().trim();
                    String email = inputEmail.getText().toString().trim();
                    String password =
inputPassword.getText().toString().trim();

                    if (!name.isEmpty() && !email.isEmpty() &&
!password.isEmpty()) {
                        if (!(password.length() <6)) {
                            registerUser(name, email, password);
                        } else {
                            Toast.makeText(getApplicationContext(),
                                    "Please enter a password with min 6
characters!", Toast.LENGTH_SHORT)
                                    .show();
                        }
                    } else {
                        Toast.makeText(getApplicationContext(),
                                "Please enter your details!",
Toast.LENGTH_LONG)
                                .show();
                    }
                } else {
                    showAlertDialog();
                }

            }
        });

        // Link to Login Screen
        btnLinkToLogin.setOnClickListener(new View.OnClickListener() {
```

```
            public void onClick(View view) {
                Intent i = new Intent(getApplicationContext(),
                        LoginActivity.class);
                startActivity(i);
                finish();
            }
        });

    }

    private boolean isNetworkConnected() {
        ConnectivityManager conManager =
(ConnectivityManager)getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo netInfo = conManager.getActiveNetworkInfo();
        if (netInfo == null) {
            // There are no active networks.
            return false;
        } else {
            return true;
        }
    }

    private void showAlertDialog() {
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("No Network Connection");
        builder.setMessage("Please turn on internet and try
again!").setCancelable(false)
                .setIcon(R.drawable.error)
                .setPositiveButton("OK", new
DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        dialog.cancel();
                    }
                });
        AlertDialog alert = builder.create();
        alert.show();
    }
```

```
/**
 * Function to store user in MySQL database will post params(tag, name,
 * email, password) to register url
 * */
private void registerUser(final String name, final String email,
                          final String password) {
    // Tag used to cancel the request
    String tag_string_req = "req_register";

    pDialog.setMessage("Registering ...");
    showDialog();

    StringRequest strReq = new StringRequest(Method.POST,
            AppConfig.URL_REGISTER, new Response.Listener<String>() {

        @Override
        public void onResponse(String response) {
            Log.d(TAG, "Register Response: " + response.toString());
            hideDialog();

            try {
                JSONObject jObj = new JSONObject(response);
                boolean error = jObj.getBoolean("error");
                if (!error) {
                    // User successfully stored in MySQL
                    // Now store the user in sqlite
                    String uid = jObj.getString("uid");

                    JSONObject user = jObj.getJSONObject("user");
                    String name = user.getString("name");
                    String email = user.getString("email");
                    String shared_key =user.getString("shared_key");
                    String created_at = user.getString("created_at");

                    // Inserting row in users table
                    db.addUser(name, email, uid, shared_key, created_at);

                    Toast.makeText(getApplicationContext(), "User
```

```
successfully registered. Try login now!", Toast.LENGTH_LONG).show();

                        // Launch login activity
                        Intent intent = new Intent(
                                RegisterActivity.this,
                                LoginActivity.class);
                        startActivity(intent);
                        finish();
                    } else {

                        // Error occurred in registration. Get the error
                        // message
                        String errorMsg = jObj.getString("error_msg");
                        Toast.makeText(getApplicationContext(),
                                errorMsg, Toast.LENGTH_LONG).show();
                    }
                } catch (JSONException e) {
                    e.printStackTrace();
                }

            }
        }, new Response.ErrorListener() {

            @Override
            public void onErrorResponse(VolleyError error) {
                Log.e(TAG, "Registration Error: " + error.getMessage());
                Toast.makeText(getApplicationContext(),
                        error.getMessage(), Toast.LENGTH_LONG).show();
                hideDialog();
            }
        }) {

            @Override
            protected Map<String, String> getParams() {
                // Posting params to register url
                Map<String, String> params = new HashMap<String, String>();
                params.put("name", name);
                params.put("email", email);
```

```
                params.put("password", password);

                return params;
            }

        };

        // Adding request to request queue
        AppController.getInstance().addToRequestQueue(strReq,
tag_string_req);
    }

    private void showDialog() {
        if (!pDialog.isShowing())
            pDialog.show();
    }

    private void hideDialog() {
        if (pDialog.isShowing())
            pDialog.dismiss();
    }
}
```

REFERENCES

[1] Banerjee, Mahamudul Hasan, MD. Auhidur Rahman, Rajesh Chapagain , "CLOAK: A Stream Cipher Based Encryption Protocol for Mobile Cloud Computing", IEEE Access August 2017.

[2] In-Shin Park , Yoon-Deock Lee, Jonpil Jeong, "Improved Identity Management Protocol for Secure Mobile Cloud Computing", IEEE Conference on System Science 2013.

[3] Sandesh Bhatewara, Aman Talreja, Avesh Sapkal, Shalaka Jadhav, "Secure Storage Outsourcing Protocol in Mobile Cloud Computing - a survey paper ", IJESC Volume 6 2016.

[4] Manmohan Chaturvedi, "Privacy & Security of Mobile Cloud Computing (MCC)", IEEE 2011.

[5] Sricharan yadavalli, Srinivas upputuri, Dileep kumar, "Mobile Cloud Computing With Safe Security ", May 2016.

[6] http://www.tutorialspoint.com/amazon_web_services.

[7] http://developer.android.com/traning/index.html.