CRYPTOGRAPHY BASED DATA SECURITY TECHNIQUES FOR THE CLOUD COMPUTING

A THESIS

Submitted by

THARUNGANDHI A S (RCAS2021MCS201)

in partial fulfillment for the award of the degree of

MASTER OF SCIENCE SPECIALIZATION IN INFORMATION SECURITY AND CYBER FORENSICS

DEPARTMENT OF COMPUTER SCIENCE

RATHINAM COLLEGE OF ARTS AND SCIENCE

(AUTONOMOUS)

COIMBATORE - 641021 (INDIA)

MAY-2023

RATHINAM COLLEGE OF ARTS AND SCIENCE

(AUTONOMOUS)

COIMBATORE - 641021

BONAFIDE CERTIFICATE

This is to certify that the thesis entitled **CRYPTOGRAPHY BASED DATA SECURITY TECHNIQUES FOR THE CLOUD COMPUTING** submitted by **A.S.THARUNGANDHI**, for the award of the Degree of Master in Computer Science specialization in **"INFORMATION SECURITY AND CYBER FORENSICS"** is a bonafide record of the work carried out by him under my guidance and supervision at Rathinam College of Arts and Science, Coimbatore

Ms. Sarmila M.E., **Dr.P.Sivaprakash M.Tech.,Ph.d**

Supervisor Mentor

Submitted for the University Examination held on 09.05.2023

INTERNAL EXAMINER **EXTERNAL EXAMINER**

RATHINAM COLLEGE OF ARTS AND SCIENCE

(AUTONOMOUS)

COIMBATORE - 641021

DECLARATION

I **A.S.THARUNGANDHI** hereby declare that this THESIS entitled **"CRYPTOGRAPHY BASED DATA SECURITY TECHNIQUES FOR THE CLOUD COMPUTING",** is the record of the original work done by me under the guidance of Ms. Sarmila M.E., Faculty Rathinam college of arts and science, Coimbatore. To the best of my knowledge this work has not formed the basis for the award of any degree a similar award to any candidate in any University.

Place: Coimbatore **THARUNGANDHI A S**

Date: 09.05.2023

Signature of the Student

COUNTERSIGNED

Ms. Sarmila M.E.,

Supervisor

Contents

Acknowledgement

On successful completion for project look back to thank who made in possible. First and foremost, thank **"THE ALMIGHTY"** for this blessing on us without which we could have not successfully our project.I am extremely grateful to **Dr.Madan.A. Sendhil, M.S., Ph.D.,** Chairman, Rathinam Group of Institutions, Coimbatore and **Dr. R.Manickam MCA., M.Phil., Ph.D.,** Secretary, Rathinam Group of Institutions, Coimbatore for giving me opportunity to study in this college. I am extremely grateful to **Dr.S.Balasubramanian,M.Sc.,Ph.D(Swiss).,PDF(SwissUSA).,** Principal, Rathinam College of Arts and Science(Autonomous), Coimbatore.Extend deep sense of valuation to **Mr.A.Uthiramoorthy, M.C.A., M.Phil., (Ph.D),** Rathinam College of Arts and Science (Autonomous) who has permitted to undergo the project. Unequally I thank **Dr.P.Sivaprakash, M.Tech., Ph.D.,** Mentor and **Dr.Mohamed Mallick, M.E.,Ph.D.,** Project Coordinator, and all the Faculty members of the Department - iNurture Education Solution pvt ltd for their constructive suggestions, advice during the course of study.I convey special thanks, to the supervisor **Ms. Sarmila M.E.,** who offered their inestimable support, guidance, valuable suggestion, motivations, helps given for the completion of the project.

I dedicated sincere respect to my parents for their moral motivation in completing the project.

Abstract

over the last years, cloud computing has become an important theme in the computer field. Essentially, it takes the information processing as a service, such as storage, computing. It relieves of the burden for storage management, universal data access with independent geographical locations. In PKI (Public Key Infrastructure), provable data possession protocol needs public key certificate distribution and management. It will incur considerable overheads since the verifier will check the certificate when it checks the remote data integrity. In addition to the heavy certificate verification, the system also suffers from the other complicated certificates management such as certificates generation, delivery, revocation, renewals, etc. In cloud computing, most verifiers only have low computation capacity. Identity-based public key cryptography can eliminate the complicated certificate management. In order to increase the efficiency, identity-based provable data possession is more attractive. Thus, it will be very meaningful to study the ID-DPDP.

Chapter 1

Introduction

1.1 Objective:

The objective of our project is Identity-based public key cryptography eliminate the complicated certificate management.

1.2 Scope of the project

In this project ID-DPDP protocol can realize private verification, delegated verification, and public verification

1.3 Existing system:

Essentially, it takes the information processing as a service, such as storage, computing. The integrity checking protocol must be efficient in order to make it suitable for capacity-limited end devices. It can make the clients verify whether their outsourced data is kept intact without downloading the whole data. In some application scenarios, the clients have to store their data on multi cloud servers. At the same time, the integrity checking protocol must be efficient in order to save the verifier's cost.

1.4 Drawbacks in existing system:

Data Checking is more complex using multiple servers. Needed large storage space. Insufficient data loss.

1.5 Proposed system:

Based on the client's authorization, the proposed ID-DPDP protocol can realize private verification, delegated verification, and public verification. i will study distributed remote data integrity checking model and present the corresponding concrete protocol in multi cloud storage. First ID-DPDP protocol which is provably secure under the assumption that the CDH problem is hard. Besides of the elimination of certificate management, our ID-DPDP protocol has also flexibility and high efficiency. At the same time, the proposed ID-DPDP protocol can realize private verification, delegated verification and public verification based on the client's authorization.

1.6 Advantages in proposed system:

- It has more significant storage space.
- It provides secure public data's.
- Using private key Generation.

Chapter 2

Literature survey

TITLE : Efficient Remote Data Possession Checking in Critical Information Infrastructures Ensuring Data Storage Security in Cloud Computing AUTHOR : Dr. T. Nalini, Dr.K. Manivannan, Vaishnavi Moorthy YEAR : 2013.

DESCRIPTION Cloud computing has been envisioned as the on-demand self-service, ubiquitous network access, location independent resource pooling, rapid resource elasticity, usage-based pricing and transference of risk. Today, technical research works focus on Remote data possession Checking protocols permit to check that a remote server can access an uncorrupted file with the help of third-party verifiers. In this paper, Protocol is adapted to support efficient remote data possession checking in critical information infrastructure without the help of a third-party auditor. This design allows users to audit the cloud storage with very lightweight communication and computation cost. In addition, the auditing result not only ensures strong cloud storage correctness guarantee, but also simultaneously achieves fast data error localization, i.e., the identification of misbehaving remote server. The design further supports secure and efficient dynamic operations on outsourced data, including block modification, deletion, and append.

TITLE : Efficient Remote Data Possession Checking in Critical Information Infrastructures AUTHOR : Francesca Sebe, Josep Domingo-Ferrer YEAR : 2008 DESCRIPTION Checking data possession in networked information systems such as those related to critical infrastructures (power facilities, airports, data vaults, defense systems, and so forth) is a matter of crucial importance. Remote data possession checking protocols permit checking that a remote server can access an uncorrupted file in such a way that the verifier does not need to know beforehand the entire file that is being verified. Unfortunately, current protocols only allow a limited number of successive verifications or are impractical from the computational point of view. In

this paper, I present a new remote data possession checking protocol such that 1) it allows an unlimited number of file integrity verifications and 2) its maximum running time can be chosen at set-up time and traded off against storage at the verifier.

TITLE : Identity-Based Remote Data Possession Checking in Public Clouds AUTHOR : Huaqun Wang, Qianhong Wu, Bo Qin, and Josep Domingo-Ferrer YEAR : 2010.

DESCRIPTION Checking remote data possession is of crucial importance in public cloud storage. It enables the users to check that their outsourced data have been kept intact without downloading the original data. The existing remote data possession checking (RDPC) protocols have been designed in the PKI (Public Key Infrastructure) setting. The cloud server has to validate the users' certificates before storing the data uploaded by the users in order to prevent spam. This considerable costs since numerous users may frequently upload data to the cloud server. This paper addresses this problem with a new model of identity-based RDPC (ID-RDPC) protocols. I present the first ID-RDPC protocol proven to be secure assuming the hardness of the standard computational Diffie-Hellman (CDH) problem. In addition to the structural advantage of elimination of certificate management and verification, our ID-RDPC protocol also outperforms existing RDPC protocols in the PKI setting in terms of computation and communication.

TITLE : Identity-Based Distributed Provable Data Possession in Multi-Cloud Storage AUTHOR : A. Juels, B. S. Kaliski Jr. YEAR : 2012

DESCRIPTION Remote data integrity checking is of crucial importance in cloud storage. It can make the clients verify whether their outsourced data is kept intact without downloading the whole data. In some application scenarios, the clients have to store their data on multi-cloud servers. At the same time, the integrity checking protocol must be efficient in order to save the verifier's cost. From the two points, I propose a novel remote data integrity checking model: ID-DPDP (identity-based distributed provable data possession) in multi-cloud storage. The formal system model and security model are given. Based on the bilinear pairings, a concrete ID-DPDP protocol is designed. The proposed ID-DPDP protocol is provably secure

under the hardness assumption of the standard CDH (computational Diffie-Hellman) problem. In addition to the structural advantage of elimination of certificate management, our ID-DPDP protocol is also efficient and flexible. Based on the client's authorization, the proposed ID-DPDP protocol can realize private verification, delegated verification and public verification.

Chapter 3

Distributed provable data possession modules

3.1 Modules

- User interface design.
- Client
- Private key generator.
- Combiner
- Cloud server.
- User integrated output

User interface design:

To connect with server user must give their username and password then only they can able to connect the server. If the user already exits directly can login into the server else user must register their details such as username, password, Email id, City and Country into the server. Database will create the account for the entire user to maintain upload and download rate. Name will be set as user id. Logging in is usually used to enter a specific page. It will search the query and display the query.

Client module:

This module is used to help the cloud owner to view details and upload files with the security. The individual cloud owner contains the key. The Cloud owners view the user searching details and the counting of file request details. which has massive data to be stored on the cloud for maintenance and computation, can be either individual consumer or corporation.

Private key generator:

In this module is used to help the Key Generator to generate keys to the cloud owners data and check their data is in safe also provide protection to the data. Because of providing private key any unknown persons are not easily identify our data. when receiving the identity, it outputs the corresponding private key.

Combiner:

Combiner an entity, which receives the storage request and distributes the block-tag pairs to the corresponding cloud servers. When receiving the challenge, it splits the challenge and distributes them to the different cloud servers. When receiving the responses from the cloud servers, it combines them and sends the combined response to the verifier.

Cloud server:

In this module is used to help the cloud server which is managed by cloud service provider, has significant storage space and computation resource to maintain the clients' data. Cloud Servers reside in our world-class data centers, with memory, and fully redundant networking and power all the way to the Client.

User integrated output:

This module has developed an efficient method to outsource the policy updating to the cloud server, which can satisfy all the requirements. I have also proposed an expressive attribute-based access control scheme for big data in the cloud, and designed policy updating algorithms for different types of access policies.

3.2 System techniques:

Identity-Based Distributed Provable Data Possession (ID-DPDP)

I adopt the idea that overall trust degree (OTD) comprises two parts: First-hand trust (trust based on real-time and multi-source service data) and second-hand trust (feedback). This hybrid trust calculation approach is based on a combination of two kinds of known trust methodologies: feedback-based trust and experience-based trust. The Firs-Hand-Trust is collect data from cloud provider and Second-Hand-Trust is collect feedback from users. So finally i compare two trust and providing best cloud providers.

REFERENCE PAPER: "Dynamic Provable Data Possession".

3.3 Hardware requirements:

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It shows what the system does and not how it should be implemented.

PROCESSOR : PENTIUM IV 2.6 GHz, Intel Core 2 Duo.

RAM : 512 MB DD RAM

MONITOR : 15" COLOR HARD

DISK : 40 GB

3.4 Software requirements:

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the team's and tracking the team's progress throughout the development activity.

Front End : J2EE.

Back End : MY SQL 5.5

Operating System : Windows 07 IDE :Eclipse

Chapter 4

System design

4.1 File design

The design base files are the most important of the application. The performance of the application depends on how the application is design. It has been given at most attention to reduce the size of files and redundancy. At the same time all the files are design to incorporate all relevant information regarding each entity. A single database with information about all the entities will make the application more complicated. The functions and structure of each of the database files are given below. In this the file contains the details of industries with their rates for low, medium and high quality material.

4.2 Input design

The Input design is mainly concerned with an input screen in the software. In the input design, user-oriented inputs are converted into a computer based system format. User can also select desired options from the menu, the provides all possible facilities. Also the important input format is designed in such a way that accidental errors are avoided. The user has to input only just the minimum data required, the also helps in avoiding the errors that the users may make. Accurate designing of the input format is very important in developing efficient software.

4.3 Output design:

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide permanent copy of the results for later consultations. The outputs are needed to be generated as a hard copy and as well as queries to be viewed on the screen. Keeping in view these outputs, the format of the output is taken from the outputs, which are currently being obtained after a processing. The standard printer

is to be used as output media for hard copies. The main objective of output design is to interrupt and communicate the result of the computer part of the system to user in a form, which they meet their requirements and also to communication the output specification to the programmers in a way, this is unambiguous, comprehensive. This application gets an input from user for number of floors, number of rooms and square feet and when the user selects forquality of material they need, application initiates the calculation, perform, finalize and generates the report to user.

Chapter 5

Development tools

5.1 The java framework

Java is a programming language originally developed by James Gosling at Sun Microsystems and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to bytecode that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is general-purpose, concurrent, class-based, and object-oriented, and is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere". Java is considered by many as one of the most influential programming languages of the 20th century, and is widely used from application software to web applications the java framework is a new platform independent that simplifies application development internet. Java technology's versatility, efficiency, platform portability, and security make it the ideal technology for network computing. From laptops to datacenters, game consoles to scientific supercomputers, cell phones to the Internet, Java is everywhere!

5.2 Objectives of java

To see places of Java in Action in our daily life, explore java.com. Why Software Developers Choose Java has been tested, refined, extended, and proven by a dedicated community. And numbering more than 6.5 million developers, it's the largest and most active on the planet. With its versatility, efficiency, and portability, Java has become invaluable to developers by enabling them to Write software on one platform and run it on virtually any other platform

- Create programs to run within a Web browser and Web services
- Develop server-side applications for online forums, stores, polls,

- HTML forms processing, and more
- Combine applications or services using the Java language to create highly customized applications or services
- Write powerful and efficient applications for mobile phones, remote processors, low-cost consumer products, and practically any other device with a digital heartbeat Some Ways Software Developers Learn Java
- Today, many colleges and universities offer courses in programming for the Java platform. In addition, developers can also enhance their Java programming skills by reading Sun's java.sun.com Web site, subscribing to Java technology-focused newsletters, using the Java Tutorial and the New to Java Programming Center, and signing up for Web, virtual, or instructor-led courses.

5.3 Java Server Pages - An Overview

Java Server Pages or JSP for short is Sun's solution for developing dynamic web sites. JSP provide excellent server side scripting support for creating database driven web applications. JSP enable the developers to directly insert java code into jsp file, this makes the development process very simple and its maintenance also becomes very easy. JSP pages are efficient, it loads into the web servers memory on receiving the request very first time and the subsequent calls are served within a very short period of time. In today's environment most web sites servers dynamic pages based on user request. Database is very convenient way to store the data of users and other things. JDBC provide excellent database connectivity in heterogeneous database environment. Using JSP and JDBC it's very easy to develop database driven web application. Java is known for its characteristic of "write once, run anywhere." JSP pages are Java Server Pages Java Server Pages (JSP) technology is the Java platform technology for delivering dynamic content to web clients in a portable, secure and well-defined way. The Java Server Pages specification extends the Java Servlet API to provide web application developers with a robust framework for creating dynamic web content on the server using HTML, and XML templates, and Java code, which is secure, fast, and independent of server platforms. JSP has been built on top of the Servlet API and utilizes Servlet semantics. JSP has become the preferred request handler and response mechanism.

Although JSP technology is going to be a powerful successor to basic Servlets, they have an evolutionary relationship and can be used in a cooperative and complementary manner. Servlets are powerful and sometimes they are a bit cumbersome when it comes to generating complex HTML. Most servlets contain a little code that handles application logic and a lot more code that handles output formatting. This can make it difficult to separate and reuse portions of the code when a different output format is needed. For these reasons, web application developers turn towards JSP as their preferred servlet environment.

5.4 Evolution of web applications

Over the last few years, web server applications have evolved from static to dynamic applications. This evolution became necessary due to some deficiencies in earlier web site design. For example, to put more of business processes on the web, whether in business-to-consumer (B2C) or business-to-business (B2B) markets, conventional web site design technologies are not enough. The main issues, every developer faces when developing web applications, are:

1. Scalability - a successful site will have more users and as the number of users is increasing fastly, the web applications have to scale correspondingly.
2. Integration of data and business logic - the web is just another way to conduct business, and so it should be able to use the same middle-tier and data-access code.
3. Manageability - web sites just keep getting bigger and i need some viable mechanism to manage the ever-increasing content and its interaction with business systems.
4. Personalization - adding a personal touch to the web page becomes an essential factor to keep our customer coming back again. Knowing their preferences, allowing them to configure the information they view, remembering their past transactions or frequent search keywords are all important in providing feedback and interaction from what is otherwise a fairly one-sided conversation. Apart from these general needs for a business-oriented web site, the necessity for new technologies to create robust, dynamic and compact server-side web applications has been realized. The main characteristics of today's dynamic web server applications are as follows:

1. Serve HTML and XML, and stream data to the web client

2. Separate presentation, logic and data

3. Interface to databases, other Java applications, CORBA, directory and mail services
4. Make use of application server middleware to provide transactional support.
5. Track client sessions.

5.5 Benefits of JSP

One of the main reasons why the Java Server Pages technology has evolved into what it is today and it is still evolving is the overwhelming technical need to simplify application design by separating dynamic content from static template display data. Another benefit of utilizing JSP is that it allows to more cleanly separate the roles of web application/HTML designer from a software developer. The JSP technology is blessed with a number of exciting benefits, which are chronicled as follows:

1. The JSP technology is platform independent, in its dynamic web pages, its web servers, and its underlying server components. That is, JSP pages perform perfectly without any hassle on any platform, run on any web server, and web-enabled application server. The JSP pages can be accessed from any web server.

2. The JSP technology emphasizes the use of reusable components. These components can be combined or manipulated towards developing more purposeful components and page design. This definitely reduces development time apart from the At development time, JSPs are very different from Servlets, however, they are precompiled into Servlets at run time and executed by a JSP engine which is installed on a Web-enabled application server such as BEA WebLogic and IBM WebSphere.

5.6 Servlets

Earlier in client- server computing, each application had its own client program and it worked as a user interface and need to be installed on each user's personal computer. Most web applications use HTML/XHTML that are mostly supported by all the browsers and web pages are displayed to the client as static documents. A web page can merely displays static content and it also lets the user navigate through the content, but a web application provides a more interactive experience. Any computer running Servlets or JSP needs to have a container. A container is nothing but a piece of software responsible for loading, executing and unloading the Servlets and JSP. While servlets can be used to extend the functionality of any Java- enabled server**.** They are mostly used to extend web servers, and are efficient replacement for CGI scripts. CGI was one of the earliest and most prominent server side dynamic content solutions, so before going forward it is very important to know the difference between CGI and the Servlets.

5.7 Java servlets

Java Servlet is a generic server extension that means a java class can be loaded dynamically to expand the functionality of a server. Servlets are used with web servers and run inside a Java Virtual Machine (JVM) on the server so these are safe and portable**.** Unlike applets they do not require support for java in the web browser. Unlike CGI, servlets don't use multiple processes to handle separate request. Servers can be handled by separate threads within the same process. Servlets are also portable and platform independent. A web server is the combination of computer and the program installed on it. Web server interacts with the client through a web browser. It delivers the web pages to the client and to an application by using the web browser and he HTTP protocols respectively. The define the web server as the package of large number of programs installed on a computer connected to Internet or intranet for downloading the requested files using File Transfer Protocol, serving e-mail and building,

Chapter 6 Implementation

6.1 Identity based distributed provable data possession in multi cloud storage.

Registrationpage.jsp

¡!DOCTYPE html PUBLIC ”-//W3C//DTD XHTML 1.0 Strict//EN” ”http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd”¿

¡html xmlns=”http://www.w3.org/1999/xhtml”¿

¡head¿

¡meta name=”keywords” content=”” /¿

¡meta name=”description” content=”” /¿

¡meta http-equiv=”content-type” content=”text/html; charset=utf-8” /¿

¡title¿Login¡/title¿

¡link href=”http://fonts.googleapis.com/css?family=Open+Sans:400,300,600” rel=”stylesheet” type=”text/css” /¿

¡link href=’http://fonts.googleapis.com/css?family=Abel—Satisfy’ rel=’stylesheet’ type=’text/css’ /¿

¡link href=”style.css” rel=”stylesheet” type=”text/css” media=”screen”

/¿

¡/head¿

```
¡script type=”text/javascript”¿
function validateForm()
var x = document.forms[”myForm”][”Uname”].value; if
(x == null —— x == ””)
alert(”Name must be filled out”);
return false;
var x1 = document.forms[”myForm”][”Passwd”].value; if
(x1 == null —— x1 == ””)
alert(”Password must be filled out”);
return false;
var x2 = document.forms[”myForm”][”Confirm”].value; if
(x2 == null —— x2 == ””)
alert(”Confirm Passout must be filled out”); return
false;
var x3 = document.forms[”myForm”][”Emailid”].value; if
(x3 == null —— x3 == ””)
alert(”Email Id must be filled out”);
return false; var x4 =
document.forms[”myForm”][”City”].
value; if (x4 == null —— x4 ==
””)
  alert(”City must be filled
  out”); return false;
var x5 = document.forms[”myForm”]
[”UserType”].value;
if (x5 == null —— x5 == ””) alert(”User
Type must be filled out”); return false;
```

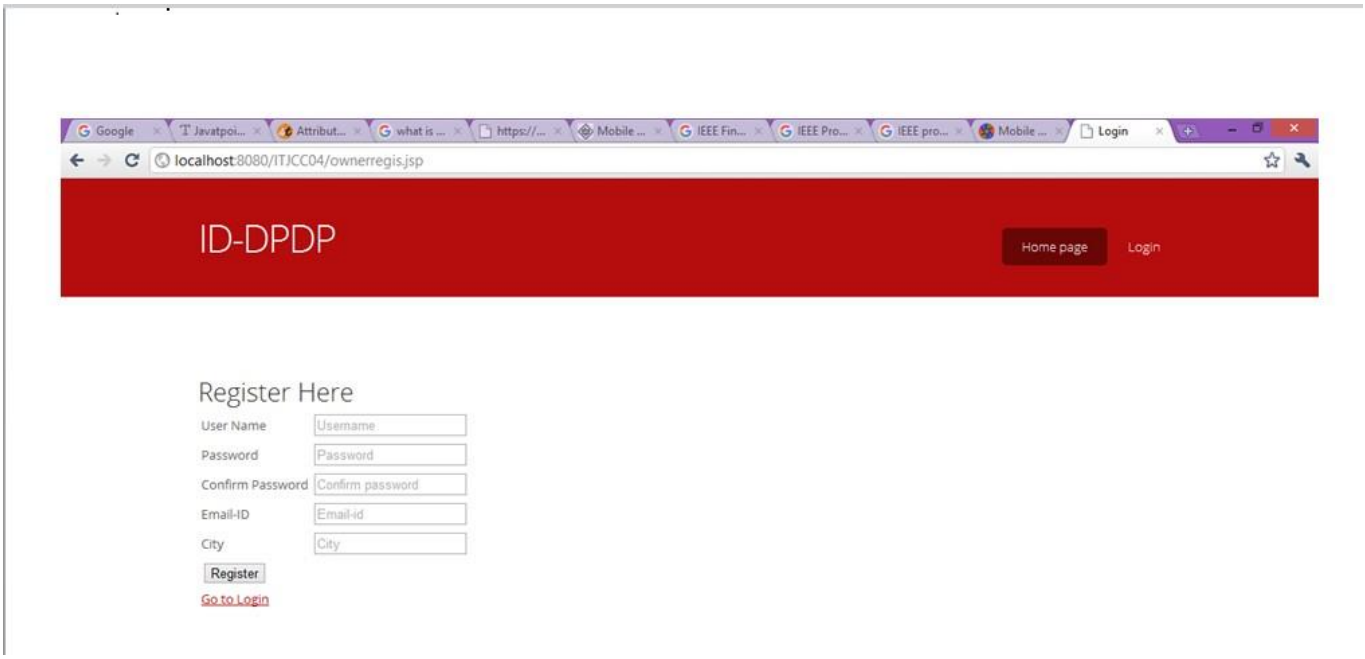

Figure 6.1: Registrationpage.jsp

Ownerreig.jsp

```
¡html xmlns=”http://www.w3.org/1999/xhtml”¿

¡head¿

¡meta name=”keywords” content=”” /¿

¡meta name=”description” content=”” /¿

¡meta http-equiv=”content-type” content=”text/html; charset=utf-8” /¿

¡title¿Login¡/title¿

¡link href=”http://fonts.googleapis.com/css?family=Open+Sans:400,300,600”
rel=”stylesheet” type=”text/css” /¿
¡link href=’http://fonts.googleapis.com/css?family=Abel—Satisfy’
rel=’stylesheet’ type=’text/css’ /¿
¡link href=”style.css” rel=”stylesheet” type=”text/css” media=”screen”

/¿

¡/head¿

¡script type=”text/javascript”¿
function validateForm()
var x = document.forms[”myForm”][”Uname”].value; if
(x == null —— x == ””)
alert(”Name must be filled out”); return false;
var x1 = document.forms[”myForm”][”Passwd”].value;
```

if (x1 == null —— x1 == ””) alert(”Password must be filled out”); return false;
var x2 = document.forms[”myForm”][”Confirm”].value; if (x2 == null —— x2 == ””)
alert(”Confirm Passout must be filled out”); return false;
var x3 = document.forms[”myForm”][”Emailid”].value; if (x3 == null —— x3 == ””)
alert(”Email Id must be filled out”); return false;
var x4 = document.forms[”myForm”][”City”].value; if (x4 == null —— x4 == ””)
alert(”City must be filled out”); return false;
var x5 = document.forms[”myForm”][”UserType”].value; if (x5 == null —— x5 == ””)
alert(”User Type must be filled out”); return false;
¡/script¿

¡body¿

¡div id=”wrapper”¿
¡div id=”header-wrapper”¿

¡div id=”header” class=”container”¿

¡div id=”logo”¿

¡h1¿¡a href=””¿ID-DPDP¡/a¿¡/h1¿

¡/div¿

¡div id=”menu”¿

¡ul¿

¡li class=”current$_{p}$*age*$_{i}$*tem*” >< *ahref* = ”*index.jsp*” > *Homepage* < */a* >< */li* >

¡li¿¡a href=”loginpage.jsp”¿Login¡/a¿¡/li¿

¡/ul¿

¡/div¿

¡/div¿

¡/div¿

¡div id=”page”¿

¡div style=”clear: both;”¿nbsp;¡/div¿

¡/div¿

¡div class=”container”¿

¡section id=”content”¿

¡form action=”ownerregis” method=”post” name=”myForm” ¿

¡table¿

¡h1¿ Register Here¡/h1¿

¡div¿
¡tr¿¡td¿ User Name ¡/td¿¡td¿ ¡input type=”text” placeholder=”Username” name=”Uname” /¿¡/td¿¡/tr¿

¡/div¿

¡div¿

¡tr¿¡td¿Password¡/td¿¡td¿¡input type=”password” placeholder=”Password” name=”Passwd” /¿¡/td¿¡/tr¿
¡/div¿

¡div¿

¡tr¿¡td¿Confirm Password¡/td¿¡td¿¡input type=”password” placeholder=”Confirm password” name=”Confirm”/¿¡/td¿
¡/div¿

¡div¿

¡tr¿¡td¿Email-ID¡/td¿¡td¿¡input type=”text” placeholder=”Email-id” name=”Emailid” /¿¡/td¿¡/tr¿
¡/div¿

¡div¿

¡tr¿¡td¿City¡/td¿¡td¿ ¡input type=”text” placeholder=”City” name=”City” /¿¡/td¿¡/tr¿
¡/div¿

¡div¿

¡div¿

¡tr¿¡td¿¡input type=”submit” value=”Register” onclick=”return validateForm()” /¿¡/td¿¡/tr¿

¡/div¿¡div¿¡td¿¡a href=”loginpage.jsp”¿Go to Login¡/a¿¡/td¿¡/tr¿¡/div¿¡/form¿¡/table¿
¡/div¿

¡/body¿

¡/html¿

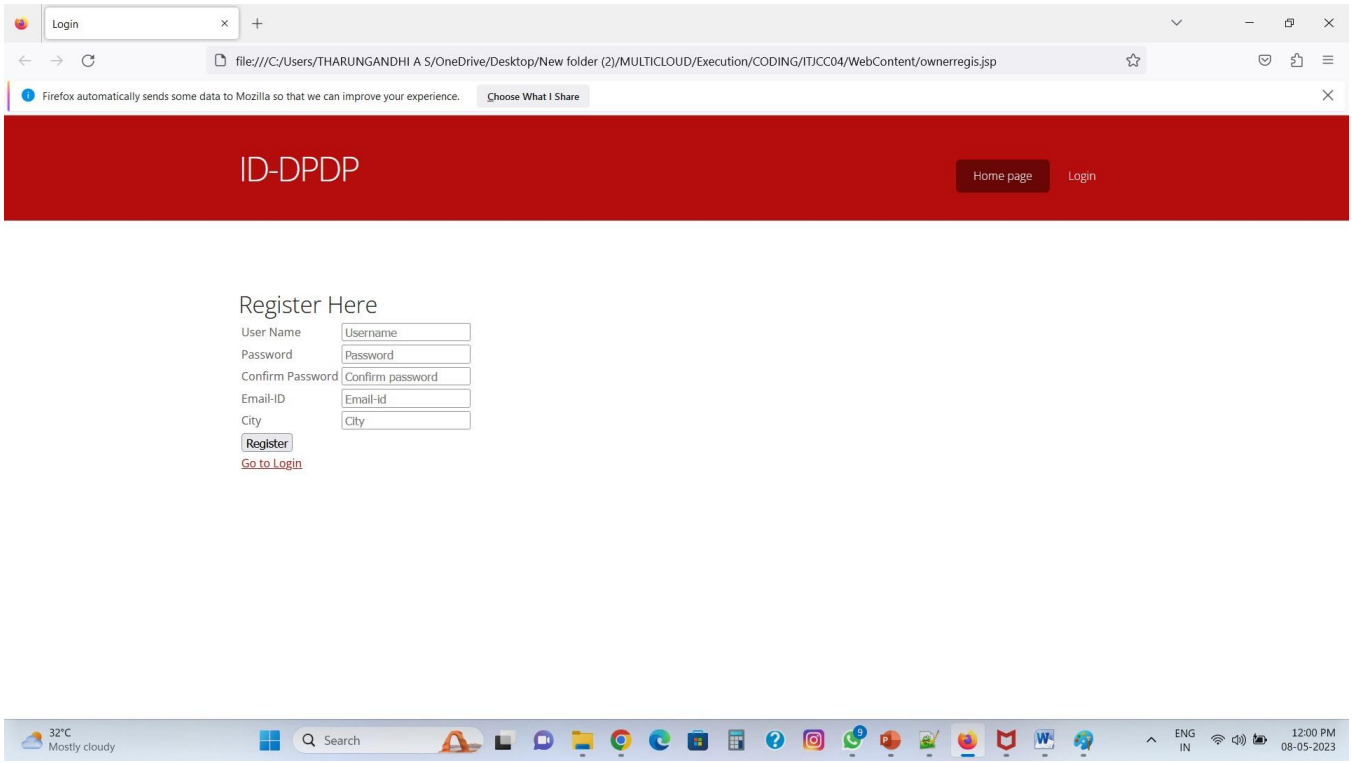

Figure 6.2: owner login

Serverhomepage.jsp

```
 ¡html xmlns=”http://www.w3.org/1999/xhtml”¿
¡head¿
 ¡meta name=”keywords” content=”” /¿
¡meta name=”description” content=”” /¿
¡meta http-equiv=”content-type” content=”text/html; charset=utf-8” /¿
¡title¿Combiner¡/title¿
¡link href=”http://fonts.googleapis.com/css?family=Open+Sans:400,300,600”     rel=”stylesheet”
     type=”text/css” /¿
 ¡link href=’http://fonts.googleapis.com/css?family=Abel—Satisfy’ rel=’stylesheet’ type=’text/css’ /¿
 ¡link href=”style.css” rel=”stylesheet” type=”text/css” media=”screen”
 /¿
 ¡/head¿
¡script type=”text/javascript”¿ function validateForm()
 var x = document.forms[”myForm”][”Uname”].value; if (x == null —— x == ””)
 alert(”Name must be filled out”); return false;
 var x1 = document.forms[”myForm”][”Passwd”].value;
 if (x1 == null —— x1 == ””) alert(”Password must be filled out”); return false;
 var x1 = document.forms[”myForm”][”Ctype”].value; if (x1 == null —— x1 == ””)
 alert(”Type must be filled out”); return false;
 ¡/script¿

 ¡body¿
 ¡div id=”wrapper”¿
 ¡div id=”header-wrapper”¿

 ¡div id=”header” class=”container”¿

 ¡div id=”logo”¿
```

¡h1¿¡a href=””¿ID-DPDP¡/a¿¡/h1¿

¡/div¿

¡div id=”menu”¿

¡ul¿

¡li class=”current*page*¡*item*” >< *ahref* = ”*index.jsp*” > *Homepage* < */a* >< */li* >

¡/div¿

¡/div¿

¡/div¿¡div id=”page”¿
¡div style=”clear: both;”¿nbsp;¡/div¿

¡/div¿

¡div class=”container”¿

¡section id=”content”¿

¡h1¿

¡p¿WELLCOME!!! ¡/p¿ ¡/h1¿

¡div id=”menu”¿

¡form action=”ServerLogin” method=”post” name=”myForm”¿

¡font size=”3” color=”blue”¿¡h1¿Cloud ServerLogin ¡/h1¿¡/font¿

¡div¿¡table¿

¡tr¿¡td¿ServerName¡/td¿¡td¿¡input type=”text” placeholder=”Servername” name=”Servername” /¿¡/td¿¡/tr¿
¡tr¿¡td¿Password¡/td¿¡td¿¡input type=”password” placeholder=”Password” name=”Passwd” /¿¡/td¿¡/tr¿
¡tr¿¡td¿Server Type ¡/td¿¡td¿

¡select name=”ServerType”¿

¡option ¿ Select
¡/option¿

¡option value=”server1”¿ server1¡/option¿

¡option value=”server2”¿server2¡/option¿

¡option value=”server3”¿server3¡/option¿
¡/select¿¡/td¿¡/tr¿
¡a href=”Verifierpage.jsp”¿Go to Verifier¡/a¿
¡tr¿¡td¿¡input type=”submit” value=”submit” onclick=”return validateForm()” /¿¡/td¿¡/tr¿
¡/div¿¡/table¿
¡/form¿
¡/section¿
¡/div¿

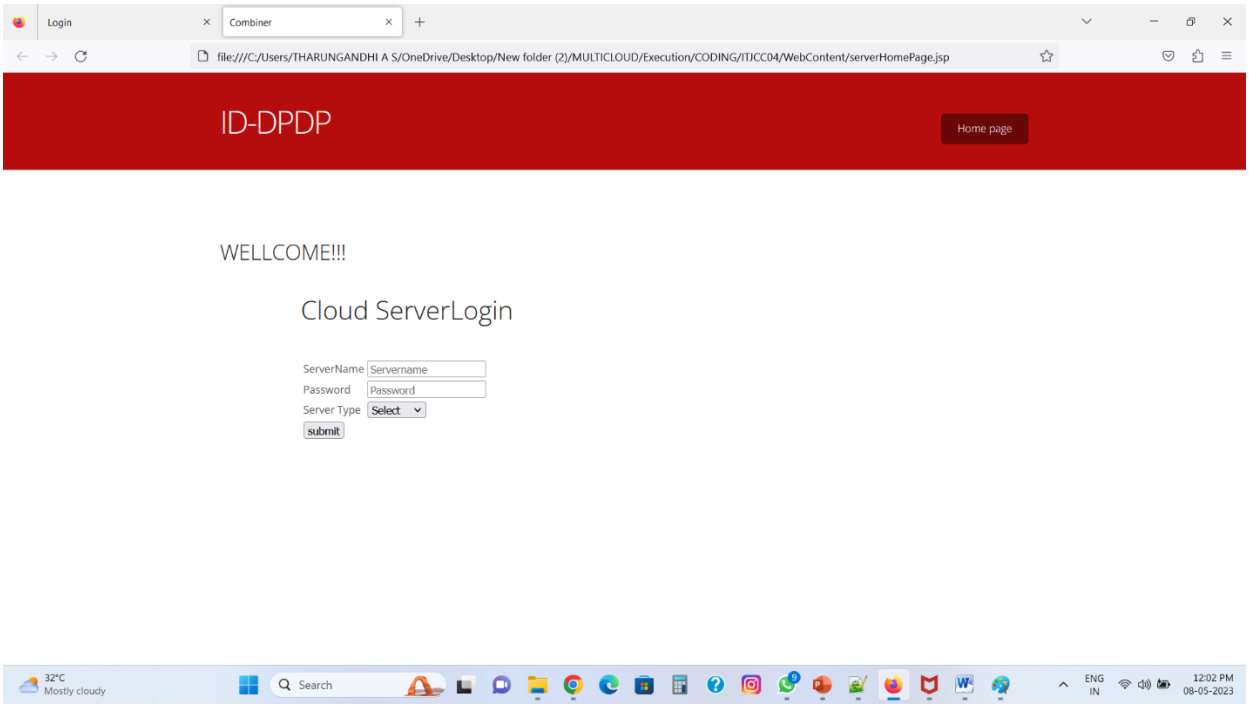

Figure 6.3: server login

keyverify.jsp

```
<%@page import="java.sql.Connection"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
  pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<%try{String key=request.getParameter("key");
String[] retrival=key.split(",");
 String secetkey=retrival[0];
    String file=retrival[1];
    String server=retrival[2];
System.out.println("i am key=="+secetkey+file+server);
Connection con=Dbcon.con();
PreparedStatement ps=con.prepareStatement("SELECT secretkey FROM
`outsource`.`serverfiles` where file=? and server=? ");
//System.out.print("select key from upload where  key='"+key+"'");
ps.setString(1,file);
ps.setString(2, server);
String keyvalue="";
ResultSet resultSet=ps.executeQuery();
if(resultSet.next())
{
keyvalue=resultSet.getString(1).toString().tri();
System.out.println("iam keyvalue==="+keyvalue);
}
```

```
if(secetkey.equals(keyvalue))
{
%><p id="msg">sucess</p><%
}
else
{%> <p id="msg">unsucess</p><%
}
%>
<% }catch(Exception e){
e.printStackTrace();
}
%>
<form action="Edit" id="iddown">
<input type="hidden" name="fname" id="filename">
</form>
<form action="edit" id="idup">
<input type="hidden" name="edit" id="Edit">
</form>
</body>
</html>
```

6.2 conclusion

In project developed PKI security and uploading data were separated stored in different server. ID-DPDP algorithm used to encrypt and decrypt the data in server and using jsp code separate used to divide the data and store in server and combiner used to combine the data easily. Additional security for viruses production for server there is one verifier to verify the data to store in server.

References

[1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner,Z. Peterson, and D. Song, "Provable Data Possession atUntrusted Stores," in Proc. CCS, 2007, pp. 598-609.

[2] G. Ateniese, R. DiPietro, L.V. Mancini, and G. Tsudik, "Scalableand Efficient Provable Data Possession," in Proc. SecureComm,2008, pp. 1-10.

[3] C.C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia,"Dynamic Provable Data Possession," in Proc. CCS, 2009,pp. 213-222.

[4] F. Sebe´, J. Domingo-Ferrer, A. Martı´nez-Balleste´, Y. Deswarte,and

J. Quisquater, "Efficient Remote Data Integrity Checkingin Critical Information Infrastructures," IEEE Trans. Knowl.Data Eng., vol. 20, no. 8, pp. 1034-1038, Aug. 2008.

[5] H.Q.Wang. (2013, Oct./Dec.). Proxy Provable Data Possession inPublic Clouds. IEEE Trans. Serv. Comput. [Online]. 6(4), pp. 551-559. Available.

[6] Y. Zhu, H. Hu, G.J. Ahn, andM. Yu, "Cooperative Provable DataPossession for Integrity Verification in Multicloud Storage,"IEEE
Trans. Parallel Distrib. Syst., vol. 23, no. 12, pp. 2231-2244,Dec. 2012.

[7] Y. Zhu, H.Wang, Z. Hu, G.J. Ahn, H. Hu, and S.S. Yau, "EfficientProvable Data Possession for Hybrid Clouds," in Proc. CCS, 2010,pp. 756-758.

[8] R. Curtmola, O. Khan, R. Burns, and G. Ateniese,
"MR-PDP:Multiple-Replica Provable Data Possession," in Proc. ICDCS,2008, pp. 411-420.

[9] A.F. Barsoum and M.A. Hasan, "Provable possession andreplication of data over cloud servers," Centre Appl. Cryptogr.Res., Univ. Waterloo, Waterloo, ON, Canada, Rep. 2010/32.

[10] Z. Hao and N. Yu, "A Multiple-Replica Remote Data PossessionChecking Protocol with Public Verifiability," in Proc. 2nd Int.Symp. Data, Privacy, E-Comm., 2010, pp. 84-89.