

A Project Report

on

**CLOUD LOG AND USER DATA RECORDS ASSURING
SOUNDNESS AND SECRECY**

Submitted for partial fulfilment of the requirements for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

By

Serupalli Mineesha (2451-16-733-080)

B. Rajesh Kumar (2451-16-733-102)

Nevuri Bhavya (2451-16-733-109)

Under the guidance of

Mrs. M. Anupama

Associate Professor

Department of CSE



MATURI VENKATA SUBBA RAO (MVSR) ENGINEERING COLLEGE

Department of Computer Science and Engineering

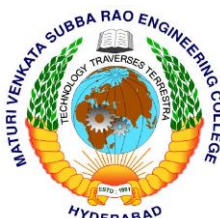
(Affiliated to Osmania University & Recognized by AICTE)

Nadergul, Saroor Nagar Mandal, Hyderabad – 501 510

Academic Year: 2019-20

MATURI VENKATA SUBBA RAO (MVSR) ENGINEERING COLLEGE

Department of Computer Science and Engineering
(Affiliated to Osmania University & Recognized by AICTE)
Nadergul, Saroor Nagar Mandal, Hyderabad – 501 510



CERTIFICATE

This is to certify that the project work entitled “Cloud Log And User Data Records Assuring Soundness And Secrecy” is a bonafide work carried out by **Ms. Serupalli Mineesha (2451-16-733-080), Mr. B. Rajesh Kumar (2451-16-733-102), Ms. Nevuri Bhavya (2451-16-733-109)** in partial fulfilment of the requirements for the award of degree of **Bachelor of Engineering in Computer Science And Engineering** from **Maturi Venkata Subba Rao Engineering College (MVSR)**, affiliated to **OSMANIA UNIVERSITY**, Hyderabad, during the Academic Year 2019-20. Under our guidance and supervision.

The results embodied in this report have not been submitted to any other university or institute for the award of any degree or diploma.

Internal Guide

Mrs. M. Anupama
Associate Professor
Department of CSE
MVSREC.

Head of the Department

Prof. J. Prasanna Kumar
Department of CSE
MVSREC.

DECLARATION

This is to certify that the work reported in the present project entitled “Cloud Log and user data records assuring soundness and secrecy” is a record of bonafide work done by us in the Department of Computer Science and Engineering, Maturi Venkata Subba Rao (MVSR) Engineering College, Osmania University. The reports are based on the project work done entirely by us and not copied from any other source. The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

Serupalli Mineesha
(2451-16-733-080)

B. Rajesh Kumar
(2451-16-733-102)

Nevuri Bhavya
(2451-16-733-109)

ACKNOWLEDGEMENT

We would like to express our sincere gratitude and indebtedness to our project guide **Mrs. M. Anupama, Associate Professor** for her valuable suggestions and interest throughout the course of this project.

We are also thankful to our principal **Dr. G. Kanaka Durga** and **Prof. J. Prasanna Kumar**, Head, Department of Computer Science and Engineering, Maturi Venkata Subba Rao (MVSR) Engineering College, Hyderabad for providing excellent infrastructure and a nice atmosphere for completing this project successfully as a part of our B.E. Degree (CSE).

We convey our heartfelt thanks to the lab staff for allowing us to use the required equipment whenever needed.

Finally, we would like to take this opportunity to thank our family for their support through the work. We sincerely acknowledge and thank all those who gave directly or indirectly their support in completion of this work.

Serupalli Mineesha (2451-16-733-080)

B. Rajesh Kumar (2451-16-733-102)

Nevuri Bhavya (2451-16-733-109)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

VISION

- To impart technical education of the highest standards, producing competent and confident engineers with an ability to use computer science knowledge to solve societal problems.

MISSION

- To make learning process exciting, stimulating and interesting.
- To impart adequate fundamental knowledge and soft skills to students.
- To expose students to advanced computer technologies in order to excel in engineering practices by bringing out the creativity in students.
- To develop economically feasible and socially acceptable software.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

The Program Educational Objectives of undergraduate program in Computer Science & Engineering are to prepare graduates who will:

PEO-1: Obtain strong fundamentals concepts, technical competency and problem solving skills to generate innovative solutions to engineering problems.

PEO-2: Continuously enhance their skills through training, independent inquiry, professional practices and pursue higher education or research by adapting to rapidly changing technology.

PEO-3: Advance in their professional careers including increased technical, multidisciplinary approach and managerial responsibility as well as attainment of leadership positions thus making them competent professionals at global level.

PEO-4: Exhibit commitment to ethical practices, societal contributions and lifelong learning.

(A) PROGRAM OUTCOMES (POs)

At the end of the program the students (Engineering Graduates) will be able to:

PO-1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO-2: Problem analysis: Identify, formulate, review research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO-3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO-4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO-5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

PO-6: The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO-7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO-8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO-9: Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO-10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO-11: Project management and finance: Demonstrate knowledge and understanding of the engineering and management principle and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO-12: Lifelong learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

(B) PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO-1: Demonstrate competence to build effective solutions for computational real-world problems using software and hardware across multidisciplinary domains.

PSO-2: Adapt to current computing trends for meeting the industrial and societal needs through a holistic professional development leading to pioneering careers or entrepreneurship.

PROJECT WORK – I
(PW 761 CS)
COURSE OBJECTIVES AND OUTCOMES

Course Objectives

- To enhance practical and professional skills.
- To familiarize tools and techniques of systematic literature survey and documentation
- To expose the students to industry practices and team work.
- To encourage students to work with innovative and entrepreneurial ideas

Course Outcomes

1. Demonstrate the ability to synthesize and apply the knowledge and skills acquired in the academic program to the real-world problems.
2. Evaluate different solutions based on economic and technical feasibility
3. Effectively plan a project and confidently perform all aspects of project management
4. Demonstrate effective written and oral communication skills

ABSTRACT

User activities can be a valuable source of information in cloud forensic investigations. Hence, ensuring there is liability and integrity of such data is crucial. Most existing solutions for secure logging are designed for conventional systems rather than the complexity of a cloud environment. In this project, we propose the Cloud Log and User Data Records Assuring Soundness and Secrecy process as an alternative scheme for the securing of logs and data in a cloud environment as well as for the user environment. In this project, data is encrypted using the individual user's public key so that only user is able to decrypt the content. In order to prevent unauthorized modification of the data, we generate proof of last log (PPL) using Rabin's fingerprint and Bloom filter. Such an approach reduces verification time significantly. We aim to deploy the findings from our project in Open Stack in a real-world context.

Serupalli Mineesha (2451-16-733-080)

B. Rajesh Kumar (2451-16-733-102)

Nevuri Bhavya (2451-16-733-109)

TABLE OF CONTENTS

CONTENTS	PAGE NO.S
Certificate	i
Declaration	ii
Acknowledgement	iii
Vision & Mission	iv
Course Objectives & Outcomes	vii
Abstract	viii
Table of Contents	ix
Chapters	
Chapter1 Introduction	
1.1 Problem statement	1
1.2 Objectives	1
1.3 Existing system	3
1.4 Proposed system	3
1.5 Scope	3
Chapter2 Literature Survey	
2.1 Existing system	5
2.2 Proposal	6
2.3 Applications	9
2.4 Summary	9
Chapter3 System Requirement Specifications	
3.1 Software Requirements	11
3.2 Hardware Requirements	11
Chapter4 System Design	
4.1 System Architecture/Block Diagram	12
4.2 Diagrams	14
Chapter5 Implementation	
5.1 Environmental Setup	18
5.1.1 Java	19
5.1.2 Java Virtual Machine	21
5.1.3 Java Script	22
5.1.4 HTML	23
5.1.5 Java Data Base Connectivity	24
5.1.6 JDBC Vs ODBC	25
5.1.7 JSP	28
5.2 Module Description	30
5.2.1 Data Owner	30
5.2.2 Data User	30
5.2.3 Cloud Server	31
5.2.4 Investigator	31

Chapter6 Tests & Results	
6.1 Test Cases	32
6.2 Results	33
Chapter7 Conclusion	40
References	41
Appendix	42

LIST OF FIGURES

Figure No.	Figure Name	Page No.
4.1	System Architecture	12
4.2.1	Flow Chart	14
4.2.2	Data Flow Diagrams	16
5.1.2	Compiling and Interpreting Java Source Code	22
5.1.6	Java Applet Architecture	26

CHAPTER 1

INTRODUCTION

1.1 Problem Statement

User activities can be a valuable source of information in cloud forensic investigations. Hence, ensuring the reliability and integrity of such data is crucial. We propose the Cloud Log and User Data Records Assuring Soundness and Secrecy process as an alternative scheme for the securing of logs and data in a cloud environment as well as for the User environment.

SecLaaS (Secure Log As A Service) encrypts the log(s) using the investigating agency's public key and stores the encrypted log(s) in a cloud server and ensures privacy and confidentiality. To facilitate log integrity, SecLaaS generates proof of past log (PPL) with the log chain and publishes it publicly after each predefined epoch.

In SecLaaS it is difficult to ensure or verify that the CSP is writing the correct information to the log or that any information pertinent to the investigation is not omitted or modified. Specifically, SecLaaS does not provide the user the ability to verify the accuracy of the log.

Extending SecLaaS, we propose a secure cloud logging scheme, Cloud Log designed to ensure CSP accountability and preserve the user's privacy. To avoid introducing unnecessary delays during user registration with the cloud service, both the CSP and the user will collectively choose a public/private key pair referred to as content concealing key (CC-key) for the user. The corresponding (content concealing) private key will be shared with other CSPs secret sharing schemes. This would allow the private key to be regenerated whenever necessary.

1.2 Objectives

Reliable cloud logs play a crucial role in forensic investigations. Log acquisition, maintaining confidentiality, integrity and forward secrecy, validity verification and accessibility by investigators (or another authorized party).

1.Preservation Of Log &Its Proof

Parser collects the log from log source. When a log file changes (i.e. new lines append) it triggers the parser to check the change and to start processing new log. Retrieving log from log source, the parser parses the log and gets the necessary information. Our goal is to keep log content secure given a parser that will provide the

system a log message in string format, regardless of content. The format of the log is out of the scope of this work.

2.Accumulator Design

Bloom filter as a proof of past data possession, which fails to account for Bloom filter's inherent potential for false positives. When using a Bloom filter technique, there is a trade-off between the number of false positives and the size of the filter. To mitigate this problem, a cryptographic one-way accumulator could be used. However, this requires significant computational overhead. In SecLaaS, they used their own data structure Bloom Tree that reduced the number of false positive incidents but requires an increased number of instances of logs and significant computational resources at verification time. This is true regardless of the number of entries being verified. In addition, it still remains vulnerable to false positives (albeit reduced).

3.Verification

Only a verification process that establishes authenticity will be able to determine the presence of log contamination. There are two types of verifications in our approach. First is verification where the user checks if the CSP is writing correct log entries or not. Second is verification by any party: user, investigator, law enforcement authority (LEA) or court of law to verify PPL to check for log modification. In both cases, the CSP can provide a small utility verification software or the user/investigator can buy it from individual software vendor (ISV) to verify.

4.Secret Key Sharing

In this project, we propose to encrypt the log with the user's private key (CC-key). In recognition that this might lead to permanent loss of log data (even for investigative entities), as the private key of a user's CC-key is known only to the user, we propose to share individual user's private key according to Shamir's or Blackley's secret key sharing strategy among multiple CSPs. This sharing scheme requires standardization. We can build sharing clouds for such a purpose when a user subscribes to a cloud service. The CSP and user together choose a pair of public/private key that is called the content concealing key (or CC-key) because it is used to hide user's log.

1.3 Existing System

SecLaaS encrypts the log(s) using the investigating agency's public key and stores the encrypted log(s) in a cloud server. This ensures privacy and confidentiality of the cloud user, unless the particular user is subject to an investigation (e.g. via a court order). To facilitate log integrity, SecLaaS generates proof of past log (PPL) with the log chain and publishes it publicly after each predefined epoch. A trust model was also suggested that stores the PPL in other clouds to minimize the risk of a malicious cloud entity altering the log. However, in SecLaaS, it is difficult to ensure or verify that the CSP is writing the correct information to the log, or that any information pertinent to the investigation is not omitted or modified. Specifically, SecLaaS does not provide the user the ability to verify the accuracy of the log (since the log is encrypted with the agency's public key).

1.4 Proposed System

Extending SecLaaS, we propose a secure cloud logging scheme, Cloud Log and User data Records Assuring Soundness and Secrecy, designed to ensure CSP accountability (i.e. writing the correct information to the log) and preserve the user's privacy. Specifically, we include the capability for the user to verify the accuracy of their log.

To do this, the log will be encrypted using the user's public key (rather than the agency's public key). To avoid introducing unnecessary delays to the forensic investigation, during user registration with the cloud service, both the CSP and the user will collectively choose a public/private key pair referred to as content concealing key (CC-key) for the user. The corresponding (content concealing) private key will be shared with other CSPs secret sharing schemes. This would allow the private key to be regenerated whenever necessary. We also demonstrate how we can leverage Rabin's fingerprint and bloom filter in PPL generation to establish log veracity. We then implement this in OpenStack and evaluate its performance.

1.5 Scope

It is designed based on the "trust no one" policy. Any party among the CSP, investigator, and user, should be capable of protecting its own security and privacy against another party or collusion between other parties. Potential challenges to designing forensic enabled cloud logging have been discussed in a number of previous

studies. For example, in an insecure cloud logging model, only the CSP can write to a log. An investigator or user can collude with the CSP to modify a log before or after publishing PPL. Thus, if a CSP falsely alters a log, whether in collusion with a malicious user or investigator or not, it can hinder the investigative process and conceal the truth of an event. This could result in an attacker failing to be identified or, more dangerously, attributing the attack to the wrong entity. Conversely, as the cloud is the host of multiple users, a malicious user can repudiate the log under investigation as his/her own log which can lead the criminal lawsuit to be dismissed. On the other hand, the log contains secretive data of the user and the user's privacy may be vulnerable due to this fact. A malicious investigator can alter the log before presenting to court authorities. Moreover, in collaboration with dishonest CSP or CSP employee(s), the investigator can violate the privacy of the user.

Cloud Log and User data Records Assuring Soundness and Secrecy , designed to ensure CSP accountability (i.e. writing the correct information to the log) and preserve the user's privacy. Specifically, we include the capability for the user to verify the accuracy of their log. To do this, the log will be encrypted using the user's public key (rather than the agency's public key). To avoid introducing unnecessary delays to the forensic investigation, during user registration with the cloud service, both the CSP and the user will collectively choose a public/private key pair referred to as content concealing key (CC-key) for the user. The corresponding (content concealing) private key will be shared with other CSPs using Shamir's or Blakley's secret sharing schemes. This would allow the private key to be regenerated whenever necessary. We also demonstrate how we can leverage Rabin's fingerprint and bloom filter in PPL generation to establish log veracity. We then implement this in OpenStack and evaluate its performance.

CHAPTER 2

LITERATURE SURVEY

2.1 Existing System

SecLaaS scheme is highly sophisticated and able to solve many problems regarding security, privacy, and admissibility of cloud logs. However, a thorough review reveals some limitations of SecLaaS. SecLaaS scheme preserves the log in the persistent database after encrypting it with law enforcement agency's public key so that no one (not even a malicious cloud employee) apart from the correct authority can decrypt it, regardless of access to the log database. Thus, cloud user privacy is ensured and no unauthorized access can take place. However, if someone who knows the private key of the law enforcement agency colludes with a malicious cloud employee and the cloud employee provides some portion of or the entire log database, then user privacy will be compromised. Thus, in the SecLaaS scheme, user privacy can be violated as a result of collusion between malicious cloud employees and law enforcement agencies.

SecLaaS scheme presents an efficient and robust procedure for cloud log management, from collecting logs to making their proof of veracity publicly available. At the same time, it preserves their integrity, privacy, and forward secrecy. SecLaaS preserves the encrypted log in storage then generates and publishes its proof so that no party can modify it. The following are notations used in this scheme:

- $H(m)$ = Collision resistance one-way hash function of message m .
- $E_{Pa}(m)$ = Encryption of m with the public key of law enforcement authority (LEA).
- $Signature_{Scsp}(m)$ = Signature of message m with private key of CSP.

It is assumed that a CSP will generate a correct log and publish its proof of past log (PPL) honestly. However, this does not account for log alteration prior to PPL publication. SecLaaS can only account for modification or contamination after publishing PPL. Thus, SecLaaS cannot mitigate log modification if the CSP writes false entries prior to publishing its PPL.

Because the CSP acts as a logger it has the sole responsibility of writing and publishing the log. The user has no knowledge of what the CSP is writing as his activity. This may lead to repudiation by a malicious user as the CSP writes log without user's knowledge. Moreover, because of the co-mingling nature of cloud environments, a

malicious cloud user can claim that logs (under investigation) belong to other users and the current scheme has no way to mitigate such claims. However, allowing the user to check his log (immediately after its generation) would enhance CSP accountability and diminish user repudiation.

2.2 Proposal

Normally logs are low-level data and hard for the common user to understand what exactly those logs signify. Thus, we will explore leveraging big data techniques to facilitate user retrieval and visualization of information from log data. Standardization of log format is also an associated research area.

To ease searching, we kept some crucial and sensitive information in plaintext format. This makes them vulnerable to be exposure. Thus, designing secure and efficient searchable encryption would extend this work.

There is also the need for an online credibility system designed to develop trust and credibility of a cloud user so that the CSP can enable stricter auditing policies for low-trust users in comparison to high-trust users.

Designing and implementing a prototype of the proposed scheme in collaboration with a realworld CSP, with the aim of evaluating its utility(e.g. performance and scalability) in a real-world environment.

Algorithms

Can be categorized into two major groups: One for Log Preservation and one for Proof Accumulation. The Log Preservation algorithm can take log entries individually or in a batch and performs processing prior to storage in a log database. This algorithm encrypts for secrecy and generates hash digest for consistency. The Proof Accumulator algorithm performs daily processing of all log entries corresponding to an IP address to prepare and publish proof of past log (PPL)

1.Preservation of Log & Its Proof

Parser collects the log from log source. For example, we collect log from log file stored in a specific directory. When a log file changes (i.e. new lines append) it triggers the parser to check the change and to start processing new log.

Retrieving log from log source, the parser parses the log and gets the necessary information. For the time being there is no standard format of a log and this is another

challenge of cloud forensics. The how, when, where, and who, aspects of logs are not yet standardized. Our goal is to keep log content secure given a parser that will provide the system a log message in string format, regardless of content. The format of the log is out of the scope of this work. For our example, we choose originating IP (FromIP), destination IP (ToIP), user id (UserID) and log time (TL). With this information, the parser generates log entry (LE).

$$LE = \langle \text{From IP}, \text{ToIP}, \text{UserID}, \text{TL} \rangle$$

Asymmetric encryption of log with individual user's public key PKU is computed to conceal user's content. This helps to resolve the problem mentioned in Fig.4.2.1. If we use public key encryption with investigator's public key PKA and store the log database in CSP's custody then neither cloud employee nor investigator can violate the privacy of user alone. This problem can be solved using Shamir or Blackley's secret key sharing scheme and is discussed in the following "secret key sharing" section. After content concealment, the scheme will generate an encrypted log entry (ELE) and this ELE will be available to the user so that user can cross check if CSP is writing a correct log entry.

Because searching through encrypted data is expensive, we keep some data in plain text format for filtering purposes.

$$ELE = EP(\text{ToIP} \parallel \text{UserID}, \text{From IP}, \text{TL})$$

After that, log chain (LC) is created in order to protect the integrity of the log and prevent potential manipulation. Cloud log and user data creates LC using a hash function with current ELE and previous LC.

$$LC = \langle H(\text{ELE} \parallel \text{LC}(\text{previous})) \rangle$$

At this stage, the payload is ready to be stored in the database. Cloud log and user data generates database log entry (DBLE) with ELE and LC and stores it in the log database. $DBLE = \langle ELE, LC \rangle$. For each DBLE, the Cloud log and user data scheme requires the generation of proof of past log (PPL) which is then made publicly available. Cloud log and user data generates the PPL in a manner that is designed to minimize the usage of memory space. In this project, we propose to generate PPL in a batch of the logs of a certain period or a certain amount of logs (e.g. n number of logs). At the end of each epoch, for each DBLE in a batch, Cloud log and user data concatenates each of the logs in a chain of logs in chronological order, derives the fingerprint, FP using Rabin's fingerprint. Then the Cloud log and user data scheme constructs an accumulator entry (AE) which is bloom filter membership information of the fingerprint FP. For

each static IP, Cloud log and user data retrieves accumulator entry (AE), epoch time TE, a signature using CSP's private key and concatenates AE, TE, with its signature, generating PPL.

Finally, the Cloud log and user data scheme publishes PPL to the web via RESTful API or RSS feed. After PPL is publicly available, it can be shared among multiple CSPs to build a trust model. This mitigates the potential for forgery so long as a single CSP is honest. After the publishing of PPL, the parser can verify their log, which is readily available in each epoch. If the user finds any discrepancy then the user can take steps accordingly. In this case, the user has to check two tests: backward check and forward check as described in Section 5.5. A backward check is to verify if logs accurately depict user activity and a forward check is to verify if the cloud server publishes the correct PPL for the corresponding logs.

2. Shamir's secret sharing algorithm

Small information will be extracted from private key following Shamir's secret sharing algorithm and each portion will be shared to one CSP. After getting secret portions of a particular user, the host cloud can reconstruct the private key to decrypt the log of that user using Shamir's secret sharing algorithm.

In this project, we propose to encrypt the log with the user's private key (CC-key). In recognition that this might lead to permanent loss of log data (even for investigative entities), as the private key of a user's CC-key is known only to the user.

We propose to share individual user's private key according to Shamir's or Blackley's secret key sharing strategy among multiple CSPs. This sharing scheme requires standardization. We can build sharing clouds for such a purpose when a user subscribes to a cloud service. The CSP and user together choose a pair of public/private key that is called the content concealing key (or CC-key) because it is used to hide user's log content. For usage of CC-key following rules are followed.

The user owns CC-key.

- CSP keeps only the public portion of the CC-key for encrypting the user's log.
- Small information will be extracted from private key following Shamir's secret sharing algorithm and each portion will be shared to one CSP. In the keys represent secret portions of the private key that is shared between five different cloud servers.
- Sharing cloud servers should have no communication/collusion between them.

- Sharing cloud servers should provide secret information of a particular user only when the court of law orders them
- When it requires investigating a cloud user's log, court-of-law will order a specific number of cloud servers to provide their secret portions for a particular user.
- After getting secret portions of a particular user, the host cloud can reconstruct the private key to decrypt the log of that user using Shamir's secret sharing algorithm.

2.3 Applications

Correctness: Cloud logs should reflect the correct history of a system's event with the occurring time. Any distortion to it is considered a violation of the correctness property.

Tamper Resistance: No one except real logger can introduce an invalid log entry as a valid one. Any sort of contamination such as the addition of new log entries, modification or deletion of existing log entries or even reordering of log entries requires prevention. At a minimum a tamper resistant scheme prevents an attacker from modification of logs without detection.

Verifiability: Verification should be possible by both the user whose activity is represented in the log and the investigating entity. The auditor or any other party involved in the related litigation need to be able to establish log veracity.

Confidentiality: Log data contains sensitive user information and requires privacy protection. For example, if a user mistakenly puts their password into the username field, the system will record this as a failed sign-in attempt and store the password as username in the log. This illustrates the need for confidentiality for all logged data in addition to data more traditionally viewed as requiring privacy protections.

Admissibility: A secure cloud log should be maintained in a way that allows it to be admissible in a court of law for criminal prosecution. The features of log integrity (correctness and tamper resistance), a chain of custody, and forward secrecy all help to achieve such admissibility.

2.4 Summary

The proposed secure logging mechanism consists of two phases, one is the log processing phase and the other is the log verification phase. The log processing phase has three computation steps: content concealment, log chain generation and PPL

generation. Content concealment is ensured by asymmetric encryption via the user's public key. Log chain generation with current log and previous log chain come after content concealment. This is as simple as collecting the previous log, concatenating it with current encrypted log and computing (MD5) hash digest. PPL generation is putting fingerprint of all log chains of an epoch into bloom filter. We prefer to compute Rabin's fingerprint whenever we need a hash digest of a one-way hash function (i.e. for PPL Generation) since Rabin's fingerprint has faster computation time.

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATIONS

H/W System Configuration

- | | |
|-----------|--------------------------------|
| Processor | - Pentium –IV or Later Version |
| RAM | - 4 GB (min) |
| Hard Disk | - 40 GB |

Software Requirements

- | | |
|------------------|-------------------------------|
| Operating System | - Windows XP or Later Version |
| Coding Language | - Java/J2EE(JSP, Servlet) |
| Back End | - MySQL |

CHAPTER 4

SYSTEM DESIGN

4.1 System Architecture/Block Diagram

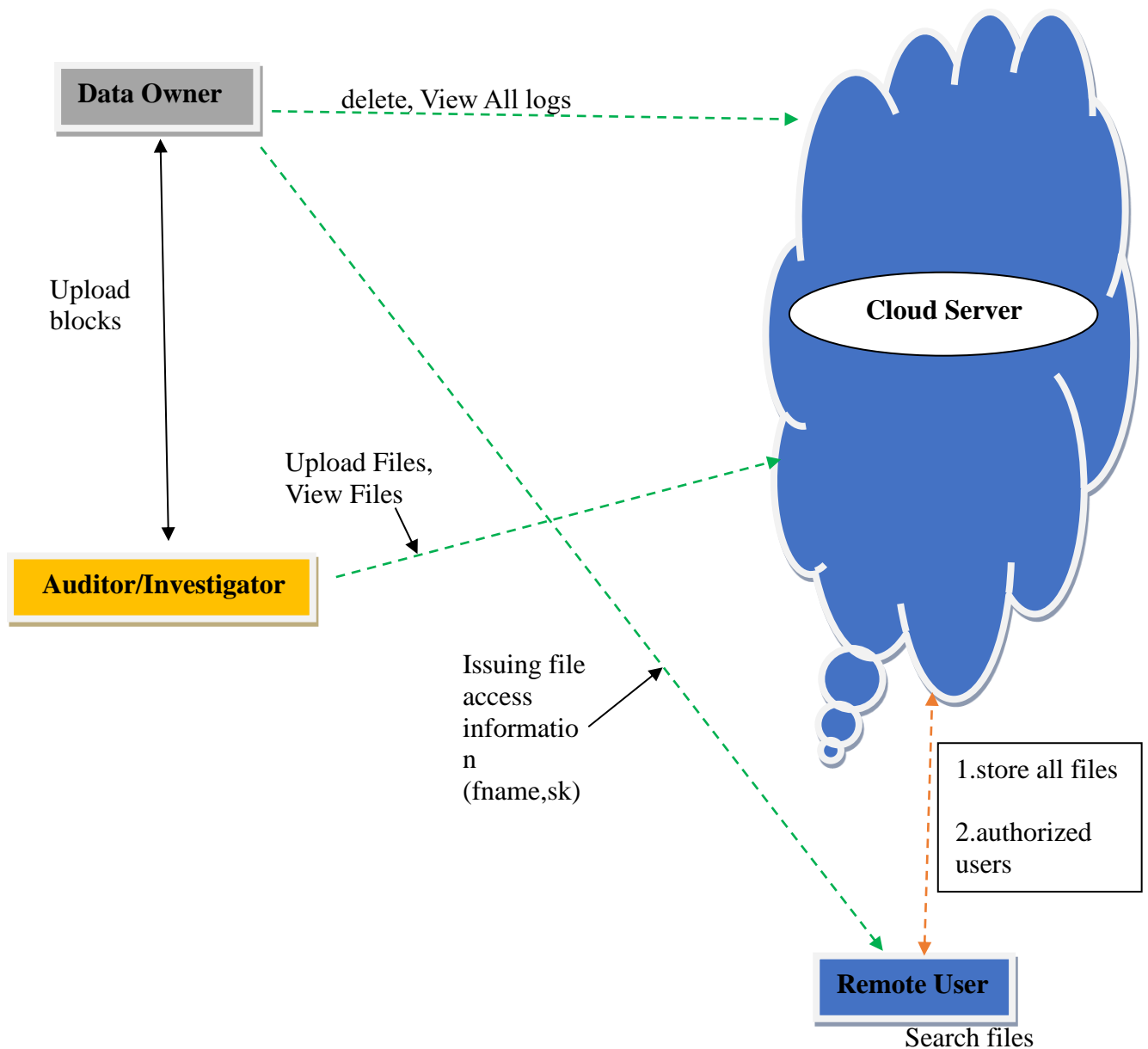


Fig 4.1 System Architecture

Input Design

Input Design plays a vital role in the life cycle of software development, it requires very careful attention of developers. The input design is to feed data to the application as accurate as possible. So inputs are supposed to be designed effectively so that the errors occurring while feeding are minimized. According to Software Engineering Concepts, the input forms or screens are designed to provide to have a validation control over the input limit, range and other related validations.

This system has input screens in almost all the modules. Error messages are developed to alert the user whenever he commits some mistakes and guides him in the right way so that invalid entries are not made. Let us see deeply about this under module design.

Input design is the process of converting the user created input into a computer-based format. The goal of the input design is to make the data entry logical and free from errors. The error in the input are controlled by the input design. The application has been developed in user-friendly manner. The forms have been designed in such a way during the processing the cursor is placed in the position where must be entered. The user is also provided with in an option to select an appropriate input from various alternatives related to the field in certain cases.

Validations are required for each data entered. Whenever a user enters an erroneous data, error message is displayed and the user can move on to the subsequent pages after completing all the entries in the current page.

Output Design

For example, the output from the computer is required to mainly create an efficient method of communication within the company primarily among the project leader and his team members, in other words, the administrator and the clients. The output of VPN is the system which allows the project leader to manage his clients in terms of creating new clients and assigning new projects to them, maintaining a record of the project validity and providing folder level access to each client on the user side depending on the projects allotted to him. After completion of a project, a new project may be assigned to the client. User authentication procedures are maintained at the initial stages itself. A new user may be created by the administrator himself or a user

can himself register as a new user but the task of assigning projects and validating a new user rests with the administrator only.

The application starts running when it is executed for the first time. The server has to be started and then the internet explorer is used as the browser. The project will run on the local area network so the server machine will serve as the administrator while the other connected systems can act as the clients. The developed system is highly user friendly and can be easily understood by anyone using it even for the first time.

4.2 Diagrams

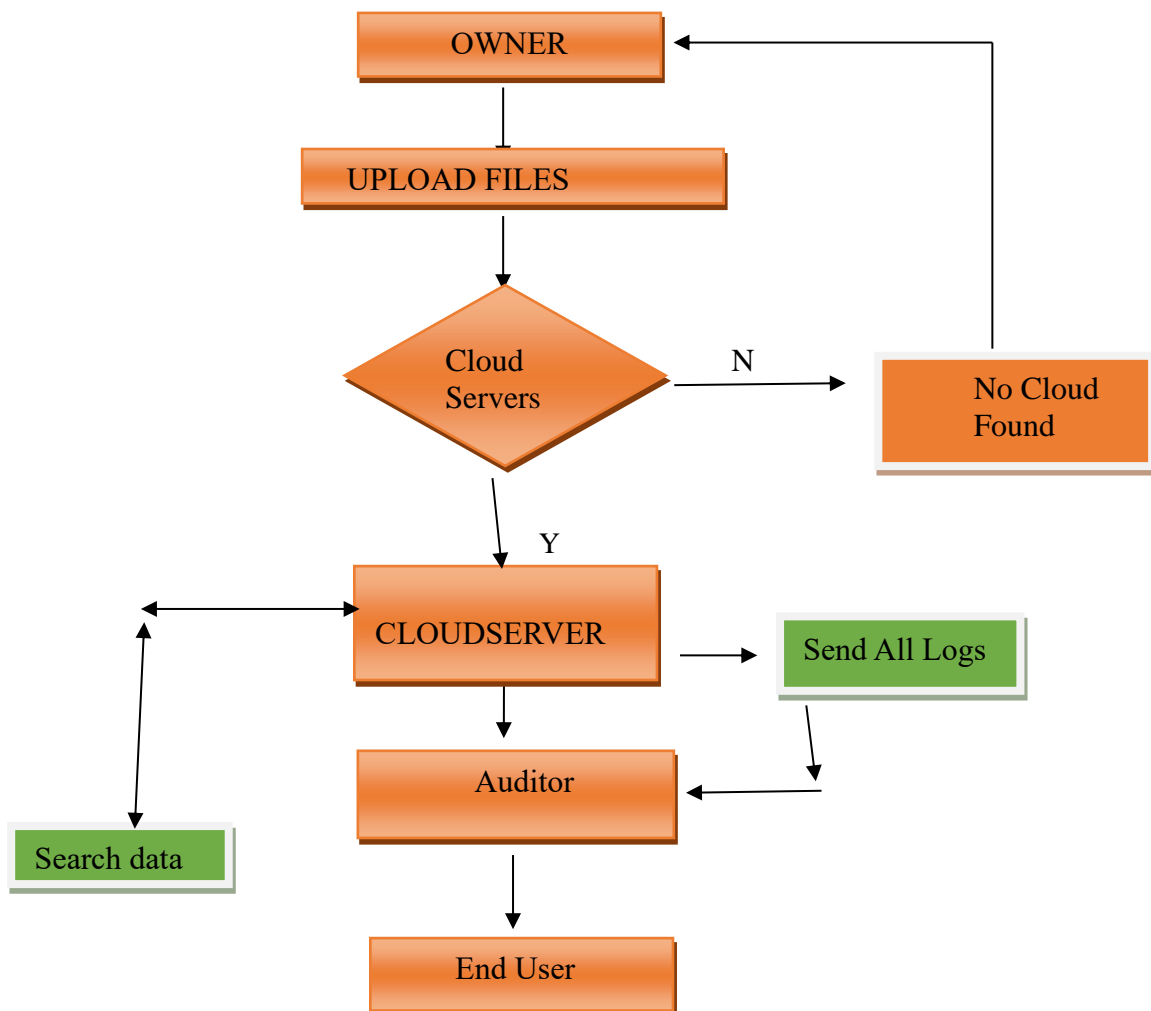
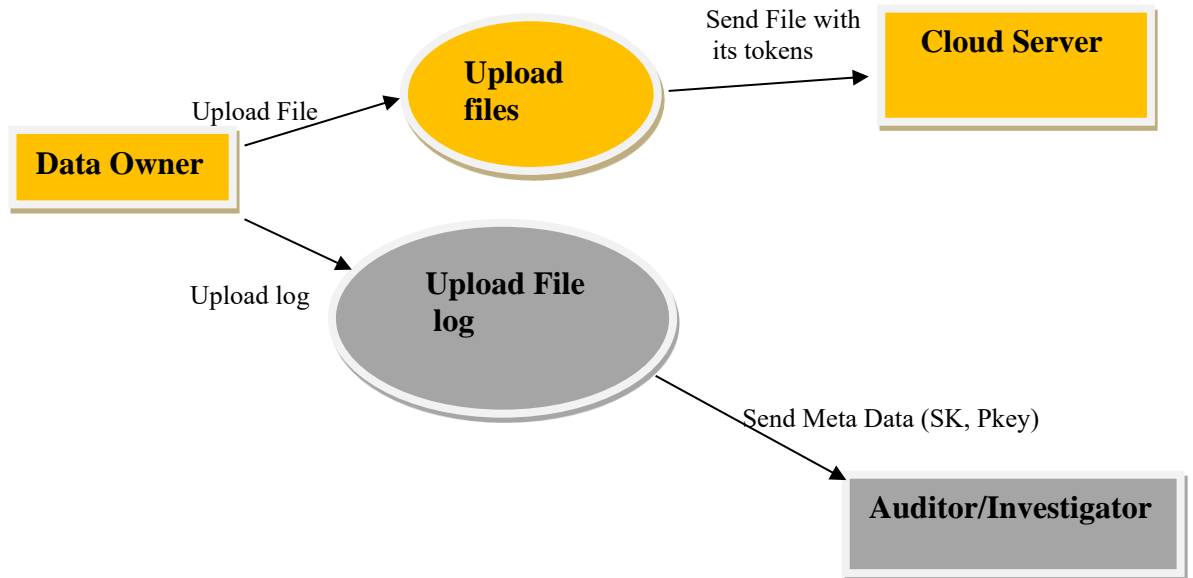


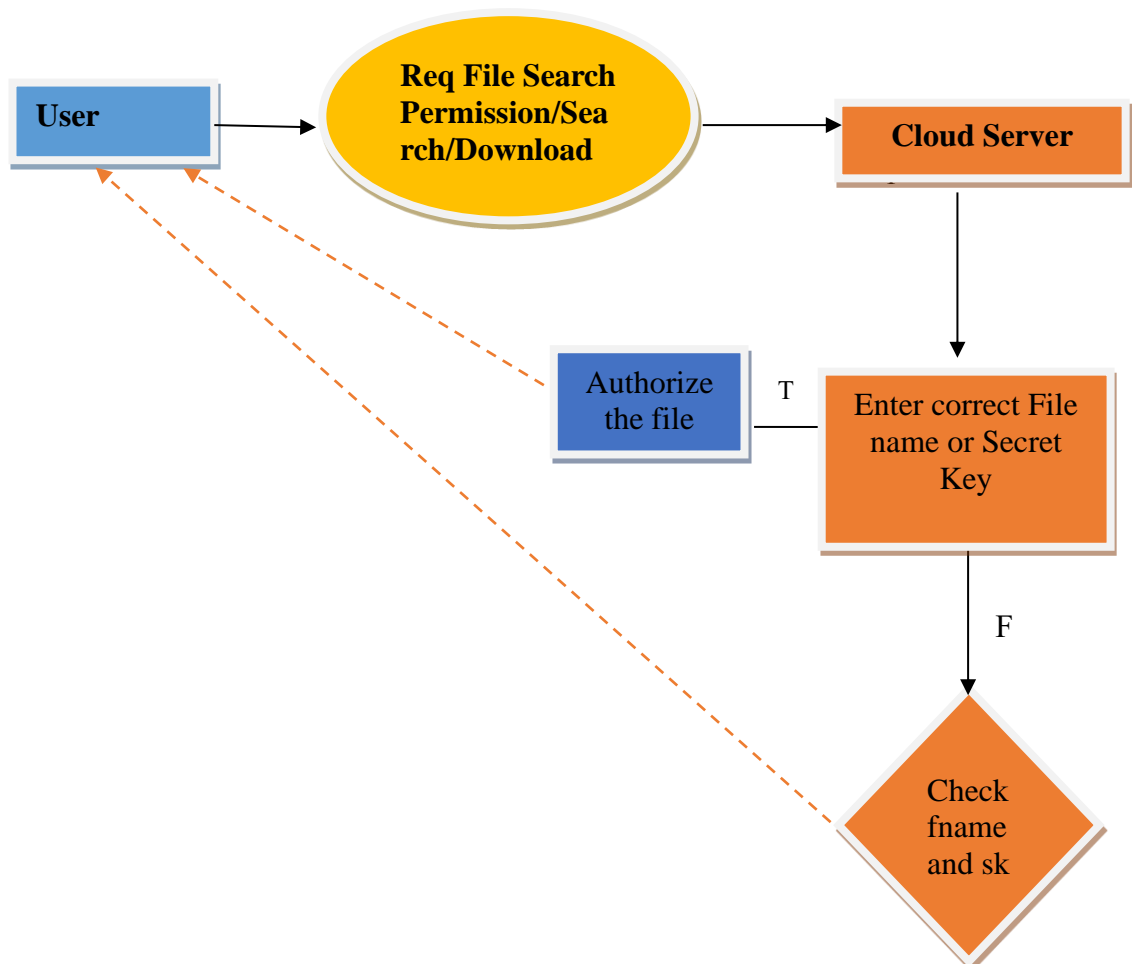
Fig 4.2.1 Flow Chart

Data Flow Diagrams

Level -0



Level -1



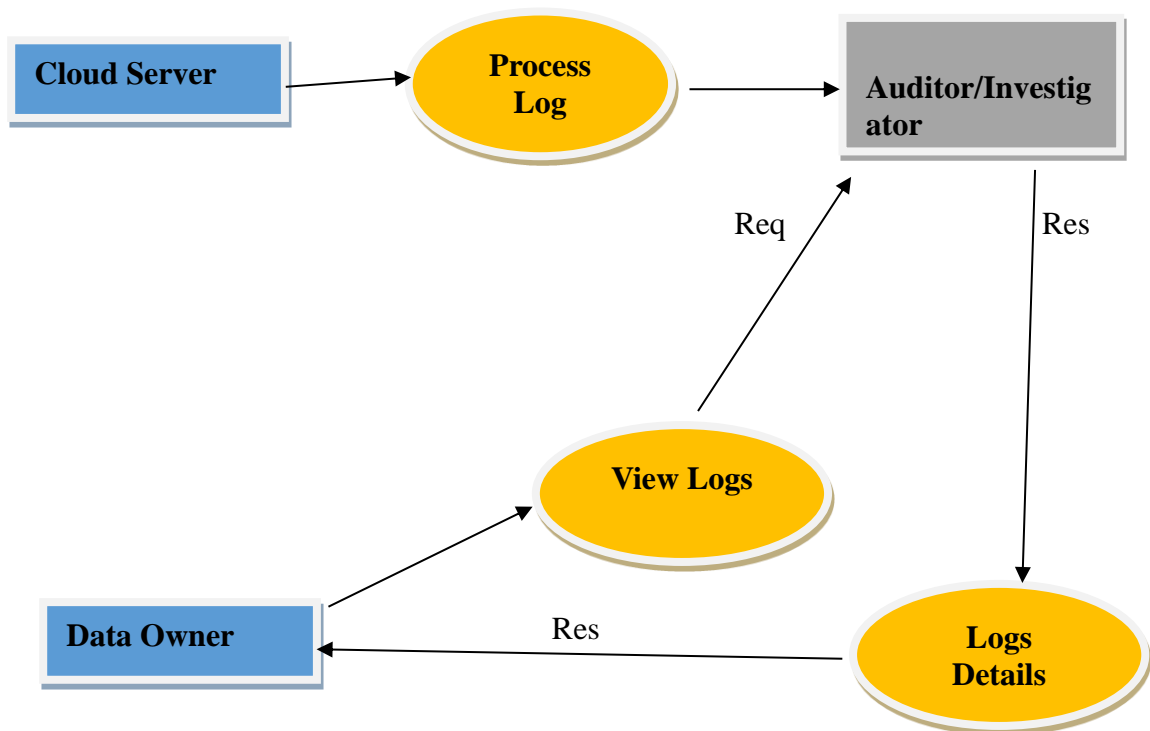
Level -2

Fig 4.2.2 Data Flow Diagrams

The first and foremost strategy for development of a project starts from the thought of designing a mail enabled platform for a small firm in which it is easy and convenient of sending and receiving messages, there is a search engine ,address book and also including some entertaining games. When it is approved by the organization and our project guide the first activity, i.e. preliminary investigation begins.

Request Clarification

After the approval of the request to the organization and project guide, with an investigation being considered, the project request must be examined to determine precisely what the system requires.

Here our project is basically meant for users within the company whose systems can be interconnected by the Local Area Network(LAN). In today's busy schedule man

need everything should be provided in a readymade manner. So taking into consideration of the vastly use of the net in day to day life, the corresponding development of the portal came into existence.

An important outcome of preliminary investigation is the determination that the system request is feasible. This is possible only if it is feasible within limited resource and time.

CHAPTER 5

IMPLEMENTATION

5.1 Environmental Setup

Client server implementations are complex but the underlying concept is simple and powerful. A client is an application running with local resources but able to request the database and relate the services from separate remote server. The software mediating this client server interaction is often referred to as MIDDLEWARE.

The typical client either a PC or a Work Station connected through a network powerful PC, Workstation, Midrange or Main Frames server usually capable of their by handling request from more than one client. However, with some configuration server may also act as client. A server may need to access other server in order to process the original client request.

The key client server idea is that client as user is essentially insulated from the location and formats of the data needs for their application. With the proper middleware, a client input from or report can transparently access and manipulate both local global database on the client machine and remote databases on one or more servers. An added bonus is the client server opens the door to multi-vendor database access indulging the heterogeneous table joins.

What is a Client Server

Two prominent systems in existence are client server and file server systems. It is essential to distinguish between client servers and file server systems. Both provide shared network access to data but the comparison dens there! The file server simplifies provides a remote disk drive that can be accessed by LAN applications on a file by file basis. The client server offers full relational database services such as SQL-Access and Record modifying, Insert, Delete with full relational integrity backup/ restore.

The key client server idea is that client as user is essentially insulated from the physical location and formats of the data needs for their application. With the proper use of the middleware, a client input from or report can transparently access and manipulate both local database on the client machine and remote databases on one or more servers. An added bonus is the client server opens the door to multi-vendor database access to the indulging heterogeneous table joins.

Why Client Server

Client server has evolved to solve a problem that has been around since the near earliest days of computing: how best to distribute your computing, data generation and data storage resources in order to obtain efficient, cost effective departmental a basest enterprise wide data processing. During mainframe era choices were quite limited. A central machine housed both the CPU and DATA (cards, tapes, drums and later disks). Access to these resources was initially confined to batched runs that produced in their departmental reports at the appropriate intervals. A strong central information service department ruled the corporation. The role of the rest of the corporation limited to the requesting new or more frequent reports and to provide hand written forms from which the central data banks were created and updated. The earliest client server solutions by therefore could best be characterized as “SLAVE-MASTER”.

The entire user interface is planned to be developed in browser specific environment with a touch of Intranet-Based Architecture for achieving the Distributed Concept.

The browser specific components are designed by using the HTML standards, and the dynamism of the designed by concentrating on the constructs of the Java Server Pages.

The Communication architecture is designed by concentrating on the Standards of Servlets and Enterprise Java Beans. The database connectivity is established by using the Java Data Base Connectivity .The standards of three-tire architecture are given major concentration to keep the standards of higher cohesion and limited coupling for effectiveness of the operations. In our project, We have chosen Java language for developing the code.

5.1.1 Java

Initially the language was called as “oak” but it was renamed as “Java” in 1995. Primary motivation of this language was the need for a platform-independent (i.e.,that

architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.

Java is a programmer's language.

Java is cohesive and consistent.

Except for those constraints imposed by the Internet environment, Java gives the programmer, full control. Finally, Java is to Internet programming where C was to system programming.

Features Of Java

1.Security

Every time you that you download a “normal” program, you are risking a viral infection. Prior to Java, most users did not download executable programs frequently, and those who did scanned them for viruses prior to execution. Most users still worried about the possibility of infecting their systems with a virus. In addition, another type of malicious program exists that must be guarded against. This type of program can gather private information, such as credit card numbers, bank account balances, and passwords. Java answers both these concerns by providing a “firewall” between a network application and your computer. When you use a Java-compatible Web browser, you can safely download Java applets without fear of virus infection or malicious intent.

2.Portability

For programs to be dynamically downloaded to all the various types of platforms connected to the Internet, some means of generating portable executable code is needed .As you will see, the same mechanism that helps ensure security also helps create portability. Indeed, Java's solution to these two problems is both elegant and efficient.

3.The Byte code

The key that allows the Java to solve the security and portability problems is that the output of Java compiler is Byte code. Byte code is a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine (JVM). That is, in its standard form, the JVM is an interpreter for byte code. Translating a Java program into byte code helps makes it much easier to run

a program in a wide variety of environments. The reason is, once the run-time package exists for a given system, any Java program can run on it. Although Java was designed for interpretation, there is technically nothing about Java that prevents on-the-fly compilation of byte code into native code. Sun has just completed its Just In Time (JIT) compiler for byte code. When the JIT compiler is a part of JVM, it compiles byte code into executable code in real time, on a piece-by-piece, demand basis. It is not possible to compile an entire Java program into executable code all at once, because Java performs various run-time checks that can be done only at run time. The JIT compiles code, as it is needed, during execution.

5.1.2 Java Virtual Machine (JVM)

Beyond the language, there is the Java virtual machine. The JVM -Java virtual machine is an important element of the Java technology. The virtual machine can be embedded within a web browser or an operating system. Once a piece of Java code is loaded onto a machine, it is verified. As part of the loading process, a class loader is invoked and does byte code verification makes sure that the code that's has been then generated by the compiler will not corrupt the machine that it's loaded on. Byte code verification takes place at the end of the compilation process to make sure that is all accurate and correct. So byte code verification is integral to the compiling and along executing of Java code. Java programming uses to produce byte codes and executes them. The first box indicates that the Java source code is located in a. Java file that is processed with a Java compiler called javac. The Java compiler produces a file called a. class file, which contains the byte code. The. Class file is then loaded across the network or loaded locally on your machine into the execution environment is the Java virtual machine, which interprets and executes the byte code.

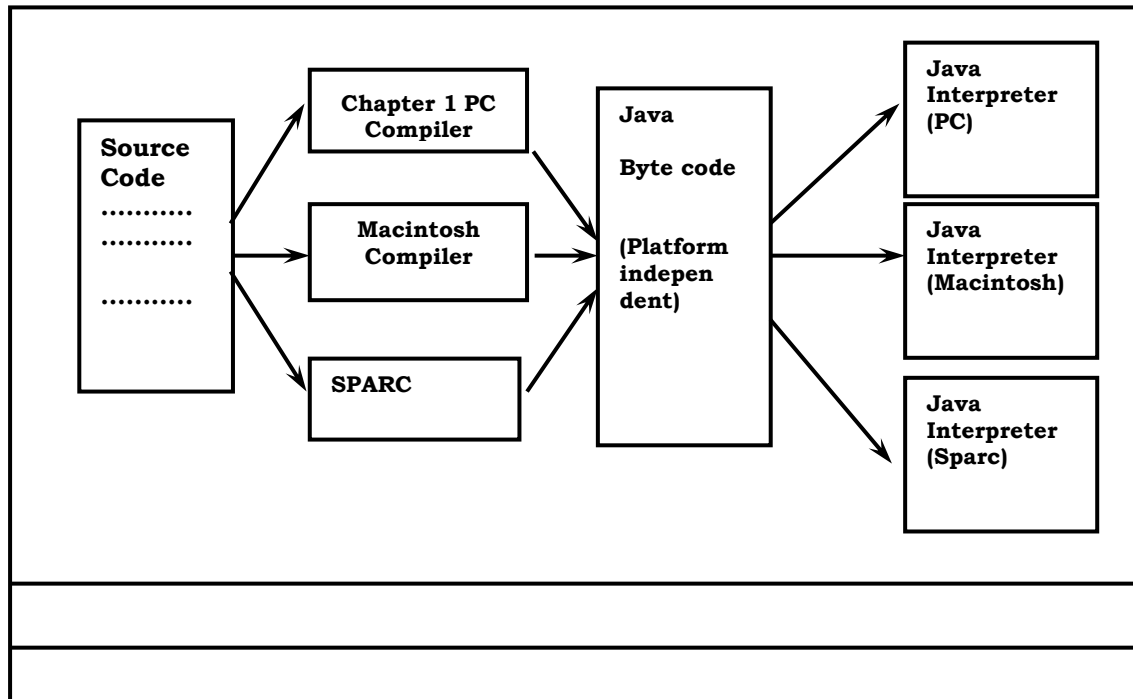


Fig 5.1.2 Compiling and interpreting Java Source Code

5.1.3 JAVASCRIPT

JavaScript is a script-based programming language that was developed by Netscape Communication Corporation. JavaScript was originally called Live Script and renamed as JavaScript to indicate its relationship with Java. JavaScript supports the development of both client and server components of Web-based applications. On the client side, it can be used to write programs that are executed by a Web browser within the context of a Web page. On the server side, it can be used to write Web server programs that can process information submitted by a Web browser and then updates the browser's display accordingly.

Even though JavaScript supports both client and server Web programming, we prefer JavaScript at Client side programming since most of the browsers supports it. JavaScript is almost as easy to learn as HTML, and JavaScript statements can be included in HTML documents by enclosing the statements between a pair of scripting tags

```
<SCRIPTS>..  
</SCRIPT>.
```

```
<SCRIPT LANGUAGE = "JavaScript">
```

JavaScript statements

</SCRIPT>

We can do much more with JavaScript, including creating entire application. JavaScript and Java are entirely different languages. A few of the most glaring differences are:

Java applets are generally displayed in a box within the web document; JavaScript can affect any part of the Web document itself.

While JavaScript is best suited to simple applications and adding interactive features to Web pages; Java can be used for incredibly complex applications.

There are many other differences but the important thing to remember is that JavaScript and Java are separate languages. They are both useful for different things; in fact they can be used together to combine their advantages.

5.1.4 Hyper Text Markup Language

Hypertext Markup Language (HTML), the languages of the World Wide Web (WWW), allows users to produces Web pages that include text, graphics and pointer to other Web pages (Hyperlinks).

HTML is not a programming language but it is an application of ISO Standard 8879, SGML (Standard Generalized Markup Language), but specialized to hypertext and adapted to the Web. The idea behind Hypertext is that instead of reading text in rigid linear structure, we can easily jump from one point to another point. We can navigate through the information based on our interest and preference. A markup language is simply a series of elements, each delimited with special characters that define how text or other items enclosed within the elements should be displayed. Hyperlinks are underlined or emphasized works that load to other documents or some portions of the same document.

HTML can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop.

HTML provides tags (special codes) to make the document look attractive. HTML tags are not case-sensitive. Using graphics, fonts, different sizes, color, etc., can enhance the presentation of the document. Anything that is not a tag is part of the document itself.

Basic HTML Tags :

<!-- -->	Specifies comments
<A>.....	Creates hypertext links
.....	Formats text as bold
<BIG>.....</BIG>	Formats text in large font.
<BODY>...</BODY>	Contains all tags and text in the HTML document
<CENTER>...</CENTER>	Creates text
<DD>...</DD>	Definition of a term
<DL>...</DL>	Creates definition list
...	Formats text with a particular font
<FORM>...</FORM>	Encloses a fill-out form
<FRAME>...</FRAME>	Defines a particular frame in a set of frames
<H#>...</H#>	Creates headings of different levels
<HEAD>...</HEAD>	Contains tags that specify information about a document
<HR>...</HR>	Creates a horizontal rule
<HTML>...</HTML>	Contains all other HTML tags
<META>...</META>	Provides meta-information about a document
<SCRIPT>...</SCRIPT>	Contains client-side or server-side script
<TABLE>...</TABLE>	Creates a table
<TD>...</TD>	Indicates table data in a table
<TR>...</TR>	Designates a table row
<TH>...</TH>	Creates a heading in a table

ADVANTAGES

A HTML document is small and hence easy to send over the net. It is small because it does not include formatted information.

HTML is platform independent.

HTML tags are not case-sensitive.

5.1.5 Java DataBase Connectivity

JDBC is a Java API for executing SQL statements. (As a point of interest, JDBC is a trademarked name and is not an acronym; nevertheless, JDBC is often thought of

as standing for Java Database Connectivity. It consists of a set of classes and interfaces written in the Java programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API.

Using JDBC, it is easy to send SQL statements to virtually any relational database. One can write a single program using the JDBC API, and the program will be able to send SQL statements to the appropriate database. The combinations of Java and JDBC lets a programmer write it once and run it anywhere.

Simply put, JDBC makes it possible to do three things:

Establish a connection with a database

Send SQL statements

Process the results.

5.1.6 JDBC versus ODBC and other APIs

At this point, Microsoft's ODBC (Open Database Connectivity) API is that probably the most widely used programming interface for accessing relational databases. It offers the ability to connect to almost all databases on almost all platforms. So why not just use ODBC from Java? The answer is that you can use ODBC from Java, but this is best done with the help of JDBC in the form of the JDBC-ODBC Bridge, which we will cover shortly. The question now becomes "Why do you need JDBC?" There are several answers to this question:

ODBC is not appropriate for direct use from Java because it uses a C interface. Calls from Java to native C code have a number of drawbacks in the security, implementation, robustness, and automatic portability of applications.

A literal translation of the ODBC C API into a Java API would not be desirable. For example, Java has no pointers, and ODBC makes copious use of them, including the notoriously error-prone generic pointer "void *". You can think of JDBC as ODBC translated into an object-oriented interface that is natural for Java programmers.

ODBC is hard to learn. It mixes simple and advanced features together, and it has complex options even for simple queries. JDBC, on the other hand, was designed to keep simple things simple while allowing more advanced capabilities where required.

A Java API like JDBC is needed in order to enable a "pure Java" solution. When ODBC is used, the ODBC driver manager and drivers must be manually installed on every client machine. When the JDBC driver is written completely in Java, however, JDBC code is automatically installable, portable, and secure on all Java platforms from network computers to mainframes.

Two-tier and Three-tier Models

The JDBC API supports both two-tier and three-tier models for database access.

In the two-tier model, a Java applet or application talks directly to the database. This requires a JDBC driver that can communicate with the particular database management system being accessed. A user's SQL statements are delivered to the database, and the results of those statements are sent back to the user. The database may be located on another machine to which the user is connected via a network. This is referred to as a client/server configuration, with the user's machine as the client, and the machine housing the database as the server. The network can be an Intranet, which, for example, connects employees within a corporation, or it can be the Internet.

In the three-tier model, commands are sent to a "middle tier" of services, which then send SQL statements to the database. The database processes the SQL statements and sends the results back to the middle tier, which then sends them to the user. MIS directors find the three-tier model very attractive because the middle tier makes it is

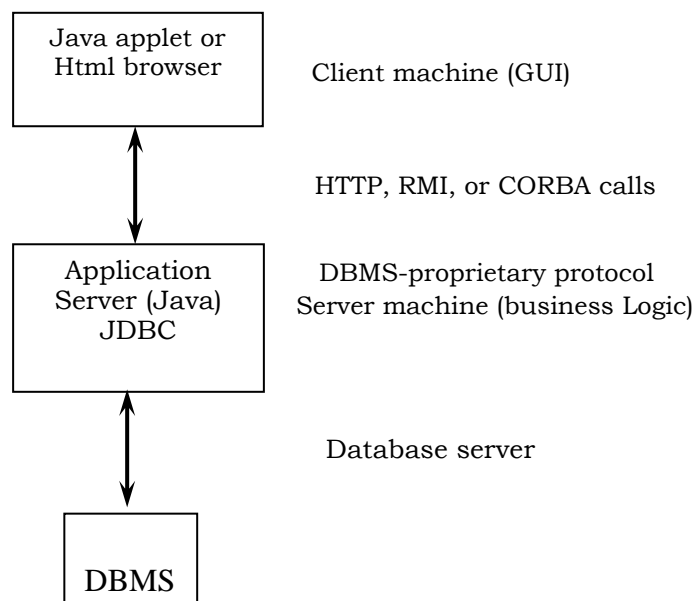


Fig 5.1.6 Java Applet Architecture

possible to maintain control over access and the kinds of updates that can be made to corporate data. Another advantage is that when there is a middle tier, the user can employ an easy-to-use higher-level API which is translated by the middle tier into the appropriate low-level calls. Finally, in many cases the three-tier architecture can provide performance advantages.

Until now the middle tier has typically been written in languages such as C or C++, which offer fast performance. However, with the introduction of optimizing compilers that translate Java byte code into efficient machine-specific code, it is becoming practical to implement the middle tier in Java. This is a big plus, making it possible to take advantage of Java's robustness, multithreading, and security features. JDBC is important to allow database access from a Java middle tier.

JDBC Driver Types

The JDBC drivers that we are aware of at this time fit into one of four categories:

JDBC-ODBC bridge plus ODBC driver

Native-API partly-Java driver

JDBC-Net pure Java driver

Native-protocol pure Java driver

JDBC-ODBC Bridge

If possible, use a Pure Java JDBC driver instead of the Bridge and an ODBC driver. This completely eliminates the client configuration required by ODBC. It also eliminates the potential that the Java VM could be corrupted by an error in the native code brought in by the Bridge (that is, the Bridge native library, the ODBC driver manager library, the ODBC driver library, and the database client library).

The JDBC-ODBC Bridge is a JDBC driver, which implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge implements JDBC for any database for which an ODBC driver is available. The Bridge is implemented as the `sun.jdbc.odbc` Java package and contains a native library used to access ODBC. The Bridge is a joint development of Intersolv and JavaSoft.

5.1.7 Java Server Pages (JSP)

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

Features of JSP

Portability

Java Server Pages files can be run on any web server or web-enabled application server that provides support for them. Dubbed the JSP engine, this support involves recognition, translation, and management of the Java Server Page lifecycle and its interaction components.

Components

It was mentioned earlier that the Java Server Pages architecture can include reusable Java components. The architecture also allows for the embedding of a scripting language directly into the Java Server Pages file. The components current supported include Java Beans, and Servlets.

Processing

A Java Server Pages file is essentially an HTML document with JSP scripting or tags. The Java Server Pages file has a JSP extension to the server as a Java Server Pages file. Before the page is served, the Java Server Pages syntax is parsed and processed into a Servlet on the server side. The Servlet that is generated outputs real content in straight HTML for responding to the client.

Access Models

A Java Server Pages file may be accessed in at least two different ways. A client's request comes directly into a Java Server Page. In this scenario, suppose the page accesses reusable Java Bean components that perform particular well-defined

computations like accessing a database. The result of the Beans computations, called result sets is stored within the Bean as properties. The page uses such Beans to generate dynamic content and present it back to the client.

In both of the above cases, the page could also contain any valid Java code. Java Server Pages architecture encourages separation of content from presentation.

Steps in the execution of a JSP Application:

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

This request is transferred to the JavaWebServer. At the server side JavaWebServer receives the request and if it is a request for a jsp file server gives this request to the JSP engine.

JSP engine is program which can understands the tags of the jsp and then it converts those tags into a Servlet program and it is stored at the server side. This Servlet is loaded in the memory and then it is executed and the result is given back to the JavaWebServer and then it is transferred back to the result is given back to the JavaWebServer and then it is transferred back to the client.

JDBC connectivity

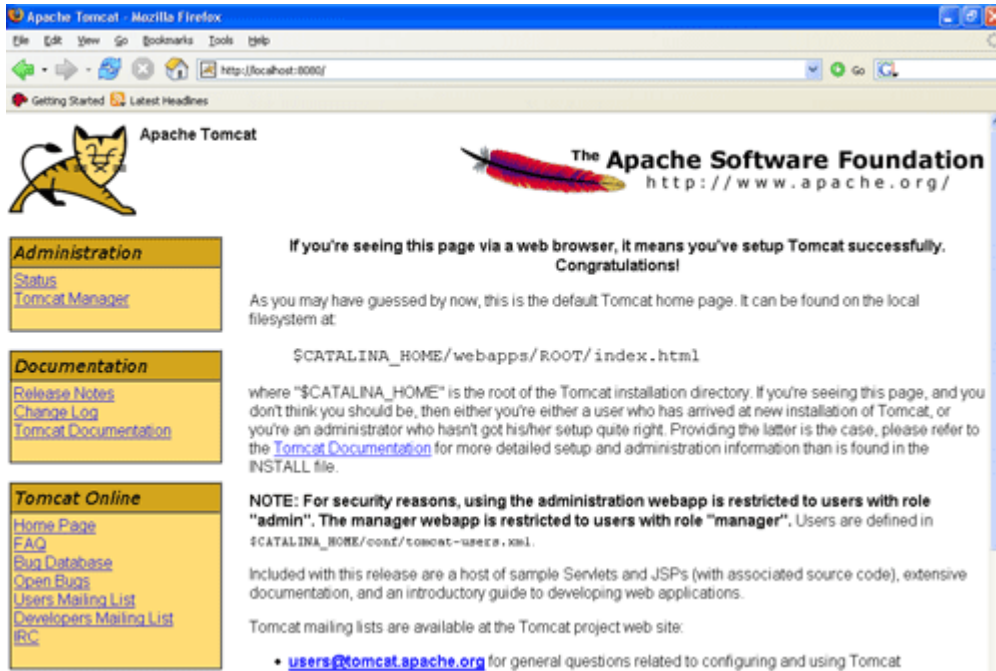
The JDBC provides database-independent connectivity between the J2EE platform and a wide range of tabular data sources. JDBC technology allows an Application Component Provider to:

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

Tomcat 6.0 web server

Tomcat is an open source web server developed by Apache Group. Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server

Pages specifications are developed by Sun under the Java Community Process. Web Servers like Apache Tomcat support only web components while an application server supports web components as well as business components (BEAs Weblogic, is one of the popular application server). To develop a web application with jsp/servlet install any web server like JRun, Tomcat etc to run your application.



5.2 Module Description

5.2.1 Data Owner

In this module, the data owner uploads their data in the cloud server. For the security purpose the data owner encrypts the file and will do the following operations like view Attackers, View Files, Upload File, Delete Files, View Proof Of Past Log(PPL)

5.2.2 Data User

In this module, user logs in by using his/her user name and password. After Login user requests search control to cloud and will Search for files based on the index keyword with the Score of the searched file and downloads the file. User can view the search of the files and also do some operations like View My Profile, View Files ,Search Files, View Searched Ratio, View Top K Documents, Request Search Permission.

5.2.3 Cloud Server

The cloud server manages a cloud to provide data storage service. Data owners encrypt their data files and store them in the cloud for sharing with Remote User and will do the following operations like View Authorize Users, View Authorize Owners, View Files, View Generated PPL, View Top Searched, View Attackers ,View Search Requests, View Time Delay Results, View Throughput Results.

5.2.4 Investigator

In this module, investigator logs in by using his/her user name and password. After Login he will do some operations like Investigate Generated PPL, Investigate Search Ratio

CHAPTER 6

TESTS & RESULTS

6.1 Test Cases

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

1. Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. It is the testing of individual software units of the application which is done after the completion of an individual unit. This is a structural testing, that relies on knowledge of its construction. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Test strategy and approach

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

Test objectives

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

Features to be tested

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

2.Functional testing

Functional testing is centered on the following items:

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

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

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

Test Case 1 : User Interface

Description - Check all the text boxes, radio buttons, buttons etc

Input - Valid username, Check the text box, radio buttons, buttons etc.

Output - Error messages regarding username invalid details.

Test Case 2 : Password Validation

Description - Check the password when the data is valid.

Input - Alpha numeric value

Output - User should not login and should show a proper error message.

Test Case 3 : Login Details

Description - Check the precise user name and password.

Input - Valid username and password should be given.

Output - Clicking on signup link takes user to signup page successfully.

6.2 Results

After running the code on server we will get this home page in the cloud. Mainly there are 4 modules i.e., Data Owner, Data User, CSP(cloud service provider), Investigator(Auditor).



This is CSP login page, here the user gives the valid username and password to login to the server.



In this interface, the cloud service provider can view the authorized owners, authorized users, view files and some other activities. Here, CSP can only view the activities done by the authorizers but cannot modify the data. Additionally, he can also view the attackers and the also all the log details in the encrypted form.

Cloud Log And User Data Records Assuring Soundness And Secrecy



Cloud forensics, Cloud log, Cloud log assuring soundness and secrecy, Cloud security, Proof of past log, Sustainable computing

Welcome Cloud Service Provider



User activity logs can be a valuable source of information in cloud forensic investigations; hence, ensuring the reliability and integrity of such logs is crucial. Most existing solutions for secure logging are designed for conventional systems rather than the complexity of a cloud environment. In this, we propose the Cloud Log And User Data Records Assuring Soundness and Secrecy process as an alternative scheme for the securing of logs in a cloud environment. In this, logs are encrypted using the individual user's public key so that only the user is able to decrypt the content. In order to prevent unauthorized modification of the log, we generate proof of past log (PPL) using Rabin's fingerprint and Bloom filter. Such an approach reduces verification time significantly. Findings from our experiments deploying Cloud log and user data records assuring soundness and secrecy in OpenStack demonstrate the utility of this in a real-world context.

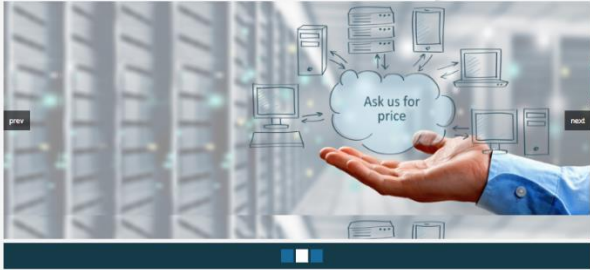
Menu

- [Home](#)
- [View Authorize Users](#)
- [View Authorize Owners](#)
- [View Files](#)
- [View Generated PPL](#)
- [View Top Searched](#)
- [View Attackers](#)
- [View Search Requests](#)
- [View Time Delay Results](#)
- [View Throughput Results](#)
- [Logout](#)

This is Data owner login page. If he/she is a authorized person then valid username and password should be given or else using the register link he can be an authorized member.


CLOUD LOG AND USER DATA RECORDS ASSURING SOUNDNESS AND SECRECY

HOME DATA OWNER DATA USER CLOUD



prev next

Data Owner Login



Name (required):

Password (required):

New Data Owner? click here to [Register](#)

Menu

- Home
- Cloud
- Data User
- Data Owner

After login, the data owner gets menu page which consists the activities like uploading files, view files, delete files and log details. In order to upload a file data owner should browse file and create an index number and by clicking on the encrypt the file will be encrypted and uploaded.

CLOUD LOG AND USER DATA RECORDS ASSURING SOUNDNESS AND SECRECY

DATA OWNER LOGOUT



prev next

Select File To Upload

Select File: No file chosen

Index:

File Name:

[Back](#)

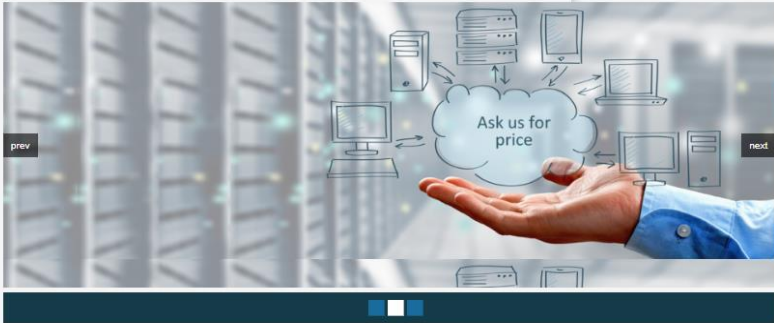
Menu

- Home
- Uploads
- View Files
- Delete Files
- Transactions
- Logout

This is the data user login page. If the data user is authorized they can give their valid username and password to login or else using the register link he can be an authorized member.


CLOUD LOG AND USER DATA RECORDS ASSURING SOUNDNESS AND SECRECY

[HOME](#)[DATA OWNER](#)[DATA USER](#)[CLOUD](#)



prevnext

Data User Login



Name (required)

Password (required)

New User? click here to [Register](#)

Menu

[Home](#)[Cloud](#)[Data User](#)[Data Owner](#)

Dept of CSE, MVSREC.

37

After login, the data user gets menu page which consists the activities like downloading files, view files, search files. In order to search file data user gets a dialog box to request search control to the csp. After authorization data user can search for required files.

Data user can also download file, by giving the required file name to download.

This is the home page of investigator(auditor).

CLOUD LOG AND USER DATA RECORDS ASSURING SOUNDNESS AND SECRECY

INVESTIGATOR | LOGOUT



Welcome Investigator or Auditor



User activity logs can be a valuable source of information in cloud forensic investigations; hence, ensuring the reliability and integrity of such logs is crucial. Most existing solutions for secure logging are designed for conventional systems rather than the complexity of a cloud environment. In this, we propose the Cloud Log And User Data Records Assuring Soundness and Secrecy process as an alternative scheme for the securing of logs in a cloud environment. In this, logs are encrypted using the individual user's public key so that only the user is able to decrypt the content. In order to prevent unauthorized modification of the log, we generate proof of past log (PPL) using Rabin's fingerprint and Bloom filter. Such an approach reduces verification time significantly. Findings from our experiments deploying this in OpenStack demonstrate the utility of this in a real-world context.

Menu

- Home
- Investigate Generated PPL
- Investigate Search Ratio
- Logout

This is the activity done by the auditor i.e., viewing list of proof of past log(PPL). That means the list of the activities done by the data owner or the user in the decrypted format.

CLOUD LOG AND USER DATA RECORDS ASSURING SOUNDNESS AND SECRECY

INVESTIGATOR | LOGOUT



Proof of Past Log(PPL)

1	View Generation PPL Based On File Operations
2	View Generation PPL Based On Search Keyword Operations

Menu

- Home
- Logout

[Back](#)

CHAPTER 7

CONCLUSION

In this project, a secure logging scheme for cloud computing along with features that facilitate the preservation of user privacy and mitigate the damaging effects of collision among other parties has been developed. Cloud log and user data preserves the privacy of cloud users by encrypting cloud logs with a public key of the respective user while also facilitating log retrieval in the event of an investigation. Moreover, it ensures accountability of the cloud server by allowing the user to identify any log modification. This has the additional effect of preventing a user from repudiating entries in his own log once the log has had its PPL established. Our implementation on OpenStack demonstrates the feasibility and practicality of the proposed scheme. The experimental results show an improvement in efficiency thanks to the features of this scheme. Findings from our project and deploying this in OpenStack demonstrate the utility of this in a real-world context.

REFERENCES

1. X. Liu, R. H. Deng, K.-K. R. Choo, and J. Weng, "An efficient privacy-preserving outsourced calculation toolkit with multiple keys," *IEEE Transactions on Information Forensics and Security*, vol. 11, pp. 2401-2414, 2016.
2. L. Zhou, Y. Zhu, and A. Castiglione, "Efficient k-NN query over encrypted data in cloud with limited key-disclosure and offline data owner," *Computers & Security*, vol. 69, pp. 84-96, 2017.
3. Q. Alam, S. U. Malik, A. Akhunzada, K.-K. R. Choo, S. Tabbasum, and M. Alam, "A Cross Tenant Access Control (CTAC) Model for Cloud Computing: Formal Specification and Verification," *IEEE Transactions on Information Forensics and Security*, vol. 12, pp. 1259-1268, 2017.

APPENDIX

Pseudo Code

Cloud module code:

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<% @ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
    <% @page import ="java.util.*"%>
<% @ include file="connect.jsp" %>
    <% @page import
="java.util.*,java.security.Key,java.util.Random,javax.crypto.Cipher,javax.crypto.spe
c.SecretKeySpec,org.bouncycastle.util.encoders.Base64"%>
    <% @ page
import="java.sql.*,java.util.Random,java.io.PrintStream,java.io.FileOutputStream,jav
a.io.FileInputStream,java.security.DigestInputStream,java.math.BigInteger,java.securi
ty.MessageDigest,java.io.BufferedInputStream" %>
<% @ page import
="java.security.Key,java.security.KeyPair,java.security.KeyPairGenerator,javax.crypt
o.Cipher"%>
    <% @page import
="java.util.*,java.text.SimpleDateFormat,java.util.Date,java.io.FileInputStream,java.i
o.FileOutputStream,java.io.PrintStream"%>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Cloud </title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style1 {font-size: 36px}
.style30 {color: #FF0000; font-size: 16px; font-family: "Times New Roman", Times,
serif; }
-->
</style>
</head>
<body>
<div class="main">

```

```

<div class="header">
<div class="header_resize">
<div class="logo">
  <h1><a href="index.html" class="style1">Cloud Log And User Data Records
Assuring Soundness And Secrecy</a></h1>
</div>
<div class="menu_nav">
  <ul>
    <li><a href="C_Main.jsp">Cloud</a></li>
    <li><a href="C_Login.jsp"><span>Logout </span></a></li>
    <li></li>
  </ul>
</div>
<div class="clr"></div>
<div class="slider">
  <div id="coin-slider"> <a href="#"></a> <a href="#"></a> <a href="#"></a> </div>
  <div class="clr"></div>
</div>
<div class="clr"></div>
</div>
<div class="content">
<div class="content_resize">
  <div class="mainbar">
    <div class="article">
      <h2><span>Proof of Past Log</span>(PPL)</h2>
      <p>&nbsp;</p>

```

```

    <table width="559" border="0">
      <tr>
        <td width="99" height="36" bgcolor="#FFFF00"><div align="center"
class="style30">1</div></td>
        <td width="442" bgcolor="#FFFF00"><div align="center"
class="style30"><a href="C_Transactions.jsp">View Generation PPL Based On File
Operations </a> </div></td>
      </tr>
      <tr>
        <td height="43" bgcolor="#FFFF00"><div align="center"
class="style30">2</div></td>
        <td bgcolor="#FFFF00"><div align="center" class="style30"><a
href="C_Transactions1.jsp">View Generation PPL Based OnSearch Keyword
Operations</a> </div></td>
      </tr>
    </table>

```

```

        <td height="20">&nbsp;</td>
        <td>&nbsp;</td>
    </tr>
</table>
<p>&nbsp;</p>
<p align="right"><a href="C_Main.jsp">Back</a></p>
</div>
</div>
<div class="sidebar">
<div class="gadget">
    <h2 class="star">Menu</h2>
    <div class="clr"></div>
    <ul class="sb_menu">
        <li><a href="C_Main.jsp">Home</a></li>
        <li><a href="C_ViewFiles.jsp">View Files</a></li>
        <li><a href="C_TopSearched.jsp">Top Searched</a></li>
        <li><a href="C_Attackers.jsp">View Attackers</a> </li>
        <li><a href="C_UserAutho.jsp">Authorize Users</a></li>
        <li><a href="C_OwnerAutho.jsp">Authorize Owners</a></li>
        <li><a href="C_GrantSearch.jsp">Search Requests</a></li>
        <li><a href="C_PersonalizedModel.jsp">Personalized Model</a></li>
        <li><a href="C_IntrestModel.jsp">User Interest Model</a></li>
        <li><a href="C_Login.jsp">Logout</a></li>
    </ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>

</div>
<div align=center></div>
</body>
</html>

//cloud main

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Cloud Main</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>

```



```

<style type="text/css">
<!--
.style1 {font-size: 36px}
.style2 {
    color: #FF0000;
    font-weight: bold;
}
-->
</style>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html" class="style1">Cloud Log And User Data Records
Assuring Soundness And Secrecy </a></h1>
</div>
<div class="menu_nav">
<ul>
<li><a href="C_Main.jsp">Cloud</a></li>
<li><a href="C_Login.jsp"><span>Logout </span></a></li>
<li></li>
</ul>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"></a> <a href="#"></a> <a href="#"></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>Welcome Cloud Service Provider </span></h2>
<p>&nbsp;</p>
<div class="img"></div>
<div class="post_content">
<p align="justify" class="style2">User activity logs can be a valuable source of
information in cloud forensic investigations; hence, ensuring the reliability and
integrity of such logs is crucial. Most existing solutions for secure logging are designed
for conventional systems rather than the complexity of a cloud environment. In this, we
propose the Cloud Log And User Data Records Assuring Soundness and Secrecy

```

process as an alternative scheme for the securing of logs in a cloud environment. In this, logs are encrypted using the individual userTMs public key so that only the user is able to decrypt the content. In order to prevent unauthorized modification of the log, we generate proof of past log (PPL) using Rabin^{€TM}s fingerprint and Bloom filter. Such an approach reduces verification time significantly. Findings from our experiments deploying Cloud log and user data records assuring soundness and secrecy in OpenStack demonstrate the utility of this in a real-world context.</p>

```

</div>
<p class="infopost">&nbsp;</p>
<div class="clr"></div>
</div>
</div>
<div class="sidebar">
<div class="gadget">
<h2 class="star">Menu</h2>
<div class="clr"></div>
<ul class="sb_menu">
<li><a href="C_Main.jsp">Home</a></li>
<li><a href="C_UserAutho.jsp">View Authorize Users</a></li>
<li><a href="C_OwnerAutho.jsp">View Authorize Owners</a></li>
<li><a href="C_ViewFiles.jsp">View Files</a></li>
<li><a href="C_ClassCloudLog.jsp">View Generated PPL</a></li>
<li><a href="C_TopSearched.jsp">View Top Searched</a></li>
<li><a href="C_Attackers.jsp">View Attackers</a> </li>
<li><a href="C_GrantSearch.jsp">View Search Requests</a></li>
<li><a href="C_TimeDelay_Results.jsp">View Time Delay
Results</a></li>
<li><a href="C_Throughput_Results.jsp">View Throughput
Results</a></li>

<li><a href="C_Login.jsp">Logout</a></li>
</ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>
</div>
<div align=center></div>
</body>
</html>

```

Data Owner Module Code:

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Data Owner </title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style1 { font-size: 36px }
.style14 { color: #0e4369; font-size: 16px; font-weight: bold; }
.style16 { font-size: 12px }
.style28 { color: #5f5f5f }
.style30 { color: #0d6887 }
-->
</style>
<script language="javascript" type='text/javascript'>
function loadFileAsText()
{
    var fileToLoad = document.getElementById("file").files[0];

    var fileReader = new FileReader();
    fileReader.onload = function(fileLoadedEvent)
    {
        var textFromFileLoaded = fileLoadedEvent.target.result;
        document.getElementById("textarea").value = textFromFileLoaded;
    };
    fileReader.readAsText(fileToLoad, "UTF-8");
}

</script>
</head>
<body>
<div class="main">
    <div class="header">
        <div class="header_resize">
            <div class="logo">
                <h1><a href="index.html" class="style1">Cloud Log And User Data Records
Assuring Soundness And Secrecy</a></h1>

```

```

</div>
<div class="menu_nav">
  <ul>
    <li class="active"><a href="DO_Main.jsp"><span>Data Owner
</span></a></li>
    <li><a href="DO_Login.jsp">Logout</a></li>
  </ul>
</div>
<div class="clr"></div>
<div class="slider">
  <div id="coin-slider"> <a href="#"></a> <a href="#"></a> <a href="#"></a> </div>
  <div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
  <div class="content_resize">
  <div class="mainbar">
  <div class="article">
    <h2>Select File To Upload </h2>
    <p class="infopost">&nbsp;</p>
    <form id="form1" name="form1" method="post" action="DO_Upload1.jsp">
      <table width="491" height="433" border="0" align="center" cellpadding="0"
cellspacing="0" style="border-collapse:collapse">
        <tr>
          <td width="133" height="35"><span class="odd style14 style28"><span
class="odd style11 style16 style30"> Select File :</span></span></td>
          <td width="423"><input required="required" type="file" name="t42"
id="file" onchange="loadFileAsText()" /></td>
        </tr>
        <tr>
          <td height="35" class="odd style14 style16 style30">Index :</td>
          <td><label>
            <input required="required" name="indexname" type="text" size="49" />
          </label></td>
        </tr>
        <tr>
          <td height="35" class="odd style14 style28"><span class="odd style11
style16 style30">File Name :</span> </td>
          <td><input required="required" name="tt" type="text" id="t42"
size="49"/></td>
        </tr>
        <tr>
          <td height="252"><span class="style30"></span></td>
          <td><textarea name="text" id="textarea" cols="50"
rows="15"></textarea></td>

```

```

        </tr>

        <tr>
        <td height="35"><div align="right"></div></td>
        <td><input type="submit" name="Submit" value="Encrypt" /></td>
        </tr>
    </table>
    <p align="right"><a href="DO_Main.jsp">Back</a></p>
    </form>
    </div>
    </div>
    <div class="sidebar">
        <div class="gadget">
            <h2 class="star">Menu</h2>
            <div class="clr"></div>
            <ul class="sb_menu">
                <li><a href="DO_Main.jsp">Home</a></li>
                <li><a href="DO_Attackers.jsp">Attackers</a></li>
                <li><a href="DO_ViewFiles.jsp">View Files</a></li>
                <li><a href="DO_Delete.jsp">Delete Files</a></li>
                <li><a href="DO_Transactions.jsp">Transactions</a></li>
                <li><a href="DO_Login.jsp">Logout</a></li>
            </ul>
        </div>
    </div>
    <div class="clr"></div>
</div>

</div>
<div align=center></div>
</body>
</html>

//Data Owner Main

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Data Owner Main</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">

```

```

<!--
.style1 {font-size: 36px}
.style2 {
    color: #FF0000;
    font-weight: bold;
}
-->
</style>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html" class="style1">Cloud Log And User Data Records
Assuring Soundness And Secrecy</a></h1>
</div>
<div class="menu_nav">
<ul>
<li class="active"><a href="DO_Main.jsp"><span>Data Owner
</span></a></li>
<li><a href="DO_Login.jsp">Logout</a></li>
</ul>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"></a> <a href="#"></a> <a href="#"></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2>Welcome to Data Owner:: <%=application.getAttribute("doname")%> </h2>
<p class="infopost">&nbsp;</p>
<div class="img"></div>
<div class="post_content">
<p align="justify" class="style2">User activity logs can be a valuable source of
information in cloud forensic investigations; hence, ensuring the reliability and
integrity of such logs is crucial. Most existing solutions for secure logging are designed
for conventional systems rather than the complexity of a cloud environment. In this, we
propose the Cloud Log And User Data Records Assuring Soundness and Secrecy
process as an alternative scheme for the securing of logs in a cloud environment. In

```

this, logs are encrypted using the individual user's public key so that only the user is able to decrypt the content. In order to prevent unauthorized modification of the log, we generate proof of past log (PPL) using Rabin's fingerprint and Bloom filter. Such an approach reduces verification time significantly. Findings from our experiments deploying this in OpenStack demonstrate the utility of this in a real-world context.

```

    </div>
    <div class="clr"></div>
  </div>
</div>
<div class="sidebar">
  <div class="gadget">
    <h2 class="star">Menu</h2>
    <div class="clr"></div>
    <ul class="sb_menu">
      <li><a href="DO_Main.jsp">Home</a></li>
      <li><a href="DO_Attackers.jsp">Attackers</a></li>
      <li><a href="DO_ViewFiles.jsp">View Files</a></li>
      <li><a href="DO_Upload.jsp">Upload File</a></li>
      <li><a href="DO_Delete.jsp">Delete Files</a></li>
      <li><a href="DO_Transactions.jsp">View Proof Of Past Log(PPL)</a></li>
      <li><a href="DO_Login.jsp">Logout</a></li>
    </ul>
  </div>
</div>
<div class="clr"></div>
</div>
</div>
<div align=center></div>
</body>
</html>

```

Data User Module Code:

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<% @ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
    <% @page import ="java.util.*"%>
<% @ include file="connect.jsp" %>
    <% @page import
="java.util.*,java.security.Key,java.util.Random,javax.crypto.Cipher,javax.crypto.spe
c.SecretKeySpec,org.bouncycastle.util.encoders.Base64"%>
    <% @ page
import="java.sql.*,java.util.Random,java.io.PrintStream,java.io.FileOutputStream,jav
a.io.FileInputStream,java.security.DigestInputStream,java.math.BigInteger,java.securi
ty.MessageDigest,java.io.BufferedInputStream" %>

```

```

<%@ page import
="java.security.Key,java.security.KeyPair,java.security.KeyPairGenerator,javax.crypt
o.Cipher"%>
<%@ page import
="java.util.*,java.text.SimpleDateFormat,java.util.Date,java.io.FileInputStream,java.i
o.FileOutputStream,java.io.PrintStream"%>

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Data User </title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style1 { font-size: 36px }
.style15 { color: #fff;
           font-weight: bold;
}
.style16 { font-size: 12px;
           color: #6d6d6d;
}
.style17 {
           color: #595f61;
           font-size: 16px;
           font-weight: bold;
}
.style18 {
           color: #595f61;
           font-weight: bold;
}
.style19 { font-size: 16px }
.style20 { font-size: 16; }
-->
</style>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html" class="style1">Cloud Log And User Data Records
Assuring Soundness And Secrecy</a></h1>
</div>
<div class="menu_nav">

```



```

<ul>
  <li class="active"><a href="DU_Main.jsp">Data User</a></li>
  <li><a href="DU_Login.jsp"> Logout </a><a href="about.html"></a></li>
</ul>
</div>
<div class="clr"></div>
<div class="slider">
  <div id="coin-slider"> <a href="#"></a> <a href="#"></a> <a href="#"></a> </div>
  <div class="clr"></div>
</div>
<div class="clr"></div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>Search Files</span></h2>
<p class="infopost">&nbsp;</p>

    <%
String user=(String)application.getAttribute("uname");

String strQuery1 = "select * from request where user='"+user+"'";
ResultSet rs = connection.createStatement().executeQuery(strQuery1);
if(rs.next()==true)
{

    String Searchper=rs.getString(3);
    if(Searchper.equalsIgnoreCase("Granted"))
    {
        %>
        <form id="form1" name="form1" method="post"
action="DU_Search1.jsp">
            <table width="572" border="0" align="center"
cellpadding="0" cellspacing="0" style="border-collapse:collapse">
                <tr>
                    <th width="236" height="40" scope="row"><div
align="left"><span class="style15"><span class="odd style16">Enter The Keyword
To Search : </span></span></div></th>
                    <td width="320"><span class="style15">
                        <input required="required" name="t14"
type="text" value="" size="30" />
                    </span></td>
                </tr>
            </table>
            <p align="center" class="style15">&nbsp;</p>

```

```

        <p align="center">
            <input type="submit" name="Submit2"
value="SEARCH" />
        </p>
    </form>
    <%
    }
    else
    {
        %>
        <h3 class="style33 style17">Search Control Not
Granted From Cloud !!</h3>
        <a href="DU_Main.jsp">Back</a><br />
        <%
    }
    }
    else
    {
        %>
        <h3 class="style33 style18"><span class="style19">You Have
Not Requested For Search Control</span> <a href="DU_SearchReq.jsp"
class="style20">Click Here To Request</a> !!</h3>
        <a href="DU_Main.jsp">Back</a><br />
        <%
    }
    %>

    <p class="infopost">&nbsp;</p>
    <div class="clr"></div>
</div>
</div>
<div class="sidebar">
<div class="gadget">
<h2 class="star">Menu</h2>
<div class="clr"></div>
<ul class="sb_menu">
    <li><a href="DU_Main.jsp">Home</a></li>
    <li><a href="DU_Profile.jsp">My Profile</a></li>
    <li><a href="DU_ViewFiles.jsp">View Files </a></li>
    <li><a href="DU_Ratio.jsp">Search Ratio</a></li>
    <li><a href="DU_TopkDoc.jsp">Top K Documents</a></li>
    <li><a href="DU_SearchReq.jsp">Req Search Control</a></li>
    <li><a href="DU_Login.jsp">Logout</a></li>
</ul>
</div>
</div>
<div class="clr"></div>
</div>
</div>

```

```

</div>
</div>
<div align=center></div>
</body>
</html>

```

// Data User main

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Data User Main</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style1 { font-size: 36px}
.style2 {
        color: #FF0000;
        font-weight: bold;
    }
-->
</style>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html" class="style1">Cloud Log And User Data Records
Assuring Soundness And Secrecy</a></h1>
</div>
<div class="menu_nav">
<ul>
<li class="active"><a href="DU_Main.jsp">Data User</a></li>
<li><a href="DU_Login.jsp"> Logout </a><a href="about.html"></a></li>
</ul>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"></a> <a href="#"></a> <a href="#"></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>Welcome to End User::
<%=application.getAttribute("uname")%></span></h2>
<p class="infopost">&nbsp;</p>
<div class="img"></div>
<div class="post_content">
<p align="justify" class="style2">User activity logs can be a valuable source of
information in cloud forensic investigations; hence, ensuring the reliability and
integrity of such logs is crucial. Most existing solutions for secure logging are designed
for conventional systems rather than the complexity of a cloud environment. In this, we
propose the Cloud Log And User Data Records Assuring Soundness and Secrecy
process as an alternative scheme for the securing of logs in a cloud environment. In
this, logs are encrypted using the individual user's public key so that only the user
is able to decrypt the content. In order to prevent unauthorized modification of the log,
we generate proof of past log (PPL) using Rabin's fingerprint and Bloom filter.
Such an approach reduces verification time significantly. Findings from our
experiments deploying this in OpenStack demonstrate the utility of this in a real-world
context.</p>
</div>
<div class="clr"></div>
</div>
</div>
<div class="sidebar">
<div class="gadget">
<h2 class="star">Menu</h2>
<div class="clr"></div>
<ul class="sb_menu">
<li><a href="DU_Main.jsp">Home</a></li>
<li><a href="DU_Profile.jsp">My Profile</a></li>
<li><a href="DU_ViewFiles.jsp">View Files </a></li>
<li><a href="DU_Search.jsp">Search Files</a></li>
<li><a href="DU_Ratio.jsp">View Searched Ratio</a></li>
<li><a href="DU_TopkDoc.jsp">View Top K Documents</a></li>
<li><a href="DU_SearchReq.jsp">Request Search Permission</a></li>
<li><a href="DU_Login.jsp">Logout</a></li>
</ul>
</div>
</div>
<div class="clr"></div>

```

```

</div>
</div>

</div>
<div align=center></div>
</body>
</html>

```

Investigator Module Code:

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<% @ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
    <% @page import = "java.util.*"%>
<% @ include file="connect.jsp" %>
    <% @page import
="java.util.*,java.security.Key,java.util.Random,javax.crypto.Cipher,javax.crypto.spe
c.SecretKeySpec,org.bouncycastle.util.encoders.Base64"%>
    <% @ page
import="java.sql.*,java.util.Random,java.io.PrintStream,java.io.FileOutputStream,jav
a.io.FileInputStream,java.security.DigestInputStream,java.math.BigInteger,java.securi
ty.MessageDigest,java.io.BufferedInputStream" %>
<% @ page import
="java.security.Key,java.security.KeyPair,java.security.KeyPairGenerator,javax.crypt
o.Cipher"%>
    <% @page import
="java.util.*,java.text.SimpleDateFormat,java.util.Date,java.io.FileInputStream,java.i
o.FileOutputStream,java.io.PrintStream"%>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Investigator</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style1 { font-size: 36px }
.style30 { color: #FF0000; font-size: 16px; font-family: "Times New Roman", Times,
serif; }
-->
</style>
</head>

```

```

<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html" class="style1">Cloud Log And User Data Records
Assuring Soundness And Secrecy</a></h1>
</div>
<div class="menu_nav">
<ul>
<li><a href="C_Main.jsp">Investigator</a></li>
<li><a href="CILogin.jsp"><span>Logout </span></a></li>
<li></li>
</ul>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"></a> <a href="#"></a> <a href="#"></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>Proof of Past Log</span>(PPL)</h2>
<p>&nbsp;</p>

<table width="559" border="0" bgcolor="#FF0000">
<tr>
<td width="99" height="36" bgcolor="#00FF00"><div align="center"
class="style30">1</div></td>
<td width="442" bgcolor="#00FF00"><div align="center"
class="style30"><a href="I_Transactions.jsp">View Generation PPL Based On File
Operations </a> </div></td>
</tr>
<tr>
<td height="43" bgcolor="#00FF00"><div align="center"
class="style30">2</div></td>
<td bgcolor="#00FF00"><div align="center" class="style30"><a
href="I_Transactions1.jsp">View Generation PPL Based OnSearch Keyword
Operations</a> </div></td>

```

```

        </tr>
        <tr>
            <td height="20">&nbsp;</td>
            <td>&nbsp;</td>
        </tr>
    </table>
    <p>&nbsp;</p>
    <p align="right"><a href="I_Main.jsp">Back</a></p>
</div>
</div>
<div class="sidebar">
    <div class="gadget">
        <h2 class="star">Menu</h2>
        <div class="clr"></div>
        <ul class="sb_menu">
            <li><a href="I_Main.jsp">Home</a></li>
            <li><a href="C_ILogin.jsp">Logout</a></li>
        </ul>
    </div>
</div>
<div class="clr"></div>
</div>s
</div>

</div>
<div align=center></div>
</body>
</html>

```

```
// Investigator main
```

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Investigator Main</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-titillium-250.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style1 { font-size: 36px}
.style2 {      color: #FF0000;

```

```

        font-weight: bold;
    }
-->
</style>
</head>
<body>
<div class="main">
<div class="header">
<div class="header_resize">
<div class="logo">
<h1><a href="index.html" class="style1">Cloud Log And User Data Records
Assuring Soundness And Secrecy</a></h1>
</div>
<div class="menu_nav">
<ul>
<li><a href="I_Main.jsp">Investigator</a></li>
<li><a href="C_ILLogin.jsp"><span>Logout </span></a></li>
<li></li>
</ul>
</div>
<div class="clr"></div>
<div class="slider">
<div id="coin-slider"> <a href="#"></a> <a href="#"></a> <a href="#"></a> </div>
<div class="clr"></div>
</div>
<div class="clr"></div>
</div>
<div class="content">
<div class="content_resize">
<div class="mainbar">
<div class="article">
<h2><span>Welcome Investigator or Auditor </span></h2>
<p>&nbsp;</p>
<div class="img"></div>
<div class="post_content">
<p align="justify"><span class="style2">User activity logs can be a valuable
source of information in cloud forensic investigations; hence, ensuring the reliability
and integrity of such logs is crucial. Most existing solutions for secure logging are
designed for conventional systems rather than the complexity of a cloud environment.
In this, we propose the Cloud Log And User Data Records Assuring Soundness and
Secrecy process as an alternative scheme for the securing of logs in a cloud
environment. In this, logs are encrypted using the individual user's public key so
that only the user is able to decrypt the content. In order to prevent unauthorized
modification of the log, we generate proof of past log (PPL) using Rabin's
fingerprint and Bloom filter. Such an approach reduces verification time significantly.

```


Findings from our experiments deploying this in OpenStack demonstrate the utility of this in a real-world context.

```

</div>
<p class="infopost">&nbsp;</p>
<div class="clr"></div>
</div>
</div>
<div class="sidebar">
<div class="gadget">
<h2 class="star">Menu</h2>
<div class="clr"></div>
<ul class="sb_menu">
<li><a href="I_Main.jsp">Home</a></li>
<li><a href="I_Investigate.jsp">Investigate Generated PPL</a></li>
<li><a href="I_Ratio.jsp">Investigate Search Ratio</a></li>

<li><a href="C_Login.jsp">Logout</a></li>
</ul>
</div>
</div>
</div>
<div class="clr"></div>
</div>
</div>
<div align=center></div>
</body>
</html>

```

MVSR Engineering College, Dept. of CSE**COURSE NAME: PROJECT****COURSE CODE: PW 961 CS****Course Objectives:**

1. Understand the significance of survey in necessary domains for problem identification.
2. Understand to map requirements into software specification.
3. Learn to apply concepts of software engineering for design of the identified real world problem
4. Improve the coding capabilities by implementing the various modules of project
5. Comprehend the suitable documentation procedure for a technical project.
6. Exhibit effective communication and presentation skills.
7. Learn the process of planning the complete lifecycle of a project

Course Outcomes:

Code No.	Student will be able to
PW_961_CS.01	Review recent advancements to formulate a precise problem statement with applications towards society.
PW_961_CS.02	Identify system requirements, explore design alternatives to design software based solution within the scope of project.
PW_961_CS.03	Implement, test and build the solution using contemporary technologies and tools.
PW_961_CS.04	Exhibit effective communication to present ideas clearly and produce well-structured technical report.
PW_961_CS.05	Demonstrate qualities necessary to work in a team and execute project as per plan.

Table 1: Relevance of CO-PO/PSO

CO	PO addressed	PSO addressed	Cognitive levels
PW_961_CS.01	1,2,3,4,5,6,7,8	1	Analyze, Evaluate
PW_961_CS.02	1,2,3,4,5,6,7,8	1	Analyze
PW_961_CS.03	1,2,3,4,5,6,7,8	1,2	Apply, Evaluate, Analyze
PW_961_CS.04	1,2,3,4,5,6,8,10	1,2	Apply, Evaluate, Analyze
PW_961_CS.05	8,9,10,11,12	1	Apply, Evaluate, Analyze

Table 2: CO-PO/PSO matrix

CO	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO10	PO11	PO12	PSO 1	PSO2
PW_961_CS.01	3	2	3	2	2	2	2	1	-	-	-	-	1	-
PW_961_CS.02	3	3	3	2	2	2	2	2	-	-	-	-	3	-
PW_961_CS.03	3	3	3	2	2	2	2	2	-	-	-	-	3	3
PW_961_CS.04	3	3	3	2	2	2	-	1	-	1	-	-	2	3
PW_961_CS.05	-	-	-	-	-	-	-	2	2	2	2	2	2	-
CS414	3	3	3	2	2	2	1	2	1	1	1	1	3	2

Table 3: Justification for CO-PO/PSO Level – through number of sessions

CO	No of sessions	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
PW_961_CS.01	6	4	4	4	2	2	2	2	1	-	-	-	-	1	-
PW_961_CS.02	12	6	7	7	4	4	4	4	4	-	-	-	-	7	-
PW_961_CS.03	18	8	8	8	7	7	6	5	6	-	-	-	-	10	8
PW_961_CS.04	12	6	5	5	4	4	4	-	2	-	1	-	-	4	8
PW_961_CS.05	6	-	-	-	-	-	-	-	3	2	2	2	2	2	-
Total	60	24	24	24	17	17	16	11	16	6	5	4	4	26	16

Table 4: % of classroom session and Correlation level

% of classroom sessions addressing a particular PO/ PSO	Level
>=40% of classroom sessions addressing a particular PO	3 : substantial(high)
25 to 40% of classroom sessions addressing a particular PO	2 : moderate(medium)
5 to 25% of classroom sessions addressing a particular PO	1 : slight (low)
< 5% of classroom sessions addressing a particular PO	- : no correlation

Table 5: PO/PSO addressed by the Project

Project Name	Domain	In-house/ Industry	PO/PSO addressed	Internal Guide

Table 6: Rubrics Evaluation

PO/PSO	PO1,PO2,PO6,PO7				PO3	PO4,PO5, PSO1		PO4,PO5, PSO2	PO8	PO9				PO10			PO11	PO12
Rubrics	R1				R2	R3		R4	R5	R6				R7			R8	R9
Roll. No.	CI	CII	CIII	Total	CIV	CV		CVI	CVII	CVIII	CIX	CX	Total	CXI	CXII	Total	CXIII	CIV
	4	4	4	12	4	4		4	4	4	4	4	12	4	4	8	4	4

Rubrics for project

Focus Areas:

1. **Problem Formulation (PO1,PO2, PO6, PO7)**
2. **Project Design (PO3)**
3. **Build (PO4,PO5, PSO1)**
4. **Test & Deploy (PO4, PO5, PSO2)**
5. **Ethical responsibility (PO8)**
6. **Team Skills (PO9)**
7. **Project Presentation (P10)**
8. **Project management (PO11)**
9. **Lifelong Learning (PO12)**

Focus Areas	Criterion [c]	Exemplary 4	Satisfactory 3	Developing 2	Unsatisfactory 1
Problem Formulation (PO1,PO2, PO6, PO7)	I - Identify/Define Problem Ability to identify a suitable problem and define the project objectives.	Demonstrates a skillful ability to identify / articulate a problem and the objectives are well defined and prioritized.	Demonstrates ability to Identify / articulate a problem and All major objectives are identified.	Demonstrates some ability to identify / articulate a problem that is partially connected to the issues and most major objectives are identified but one or two minor ones are missing or priorities are not established.	Demonstrates minimal or no ability to identify / articulate a problem and many major objectives are not identified.
	II - Collection of Background Information: Ability to gather background Information (existing knowledge, research, and/or indications of the problem)	Collects sufficient relevant background information from appropriate sources, and is able to identify pertinent/critical information;	Collects sufficient relevant background information from appropriate sources;	Collects some relevant background information from appropriate Sources.	Minimal or no ability to collect relevant background information
	III- Define scope of the problem Ability to identify problem scope suitable to the degree considering the impact on society and environment	Demonstrates a skillful ability to define the scope of problem accurately mentioning the relevant fields of engineering precisely. Considers, explains and evaluates the impact of engineering interventions on society and environment.	Demonstrates ability to define problem scope mentioning the relevant fields of engineering broadly. Considers and explains the impact of engineering interventions on society and environment	Demonstrates some ability to define problem scope mentioning some of the relevant fields. Some consideration of the impact of engineering interventions on society and environment.	Demonstrates minimal or no ability to define problem scope and fails to mention relevant fields of engineering. Minimal or no consideration of the impact of engineering interventions on society and environment
Project Design (PO3)	IV- Understanding the Design Process and Problem Solving: Ability to explain the design process including the importance of needs, specifications, concept generation and to	Demonstrates a comprehensive ability to understand and explain a design process. Considers multiple approaches to solving a problem, and can articulate reason for choosing solution	Demonstrates an ability to understand and explain a design process. Considers multiple approaches to solving a problem, which is justified and considers consequences.	Demonstrates some ability to understand and explain a design process. Considers a few approaches to solving a problem; doesn't always consider consequences.	Demonstrates minimal or no ability to understand and explain a design process. Considers a single approach to solving a problem. Does not consider consequences.

	develop an approach to solve a problem.				
Build (PO4,PO5, PSO1)	V- Implementing Design Strategy: Ability to execute a solution taking into consideration design requirements using appropriate tool (software/hardware);	Demonstrates a skillful ability to execute a solution taking into consideration all design requirements using the most relevant tool.	Demonstrates an ability to execute a solution taking into consideration design requirements using relevant tool.	Demonstrates some ability to execute a solution but not using most relevant tool.	Demonstrates minimal or no ability to execute a solution. Solution does not directly attend to the problem.
Test & Deploy (PO4, PO5, PSO2)	VI- Evaluating Final Design: To evaluate/confirm the functioning of the final design. To deploy the project on the target environment	Demonstrates a skillful ability to evaluate/confirm the functioning of the final design skillfully, with deliberation for further Improvement after deployment.	Demonstrates an ability to evaluate/confirm the functioning of the final design. The evaluation is complete and has sufficient depth.	Ability to evaluate/confirm the functioning of the final design, but the evaluation lacks depth and/or is incomplete.	Demonstrates minimal or no ability to evaluate/confirm the functioning of the final design.
Ethical responsibility (PO8)	VII - Proper Use of Others' Work: Ability to recognize, understand and apply proper ethical use of intellectual property, copyrighted materials, and research.	Always recognizes and applies proper ethical use of intellectual property, copyrighted materials, and others' research.	Recognizes and applies proper ethical use of intellectual property, copyrighted materials, and others' research.	Some recognition and application of proper ethical use of intellectual property, copyrighted materials, and others' research.	Minimal or no recognition and/or application of proper ethical use of intellectual property, Copyrighted materials, or others' research.
Team Skills (PO9)	VIII - Individual Work Contributions and Time Management: Ability to carry out individual Responsibilities and manage time (estimate, prioritize, establish deadlines/ milestones, follow timeline, plan for contingencies, adapt to change).	Designated jobs are accomplished by deadline; completed work is carefully and meticulously prepared and meets all requirements.	Designated jobs are accomplished by deadline; completed work meets requirements.	Designated jobs are accomplished by deadline; completed work meets most requirements.	Some Designated jobs are accomplished by deadline; completed work meets some requirements.
	IX - Leadership Skills: Ability to lead a team. (i) Mentors and accepts mentoring from others. (ii) Demonstrates capacity for initiative while respecting others' roles. (iii) Facilitates others' involvement. (iv) Evaluates team Effectiveness and	Exemplifies leadership skills.	Demonstrates leadership skills.	Demonstrates some leadership skills at times.	Demonstrates minimal or no leadership skills.

	plans for improvements				
	X - Working with Others: Ability to listen to, collaborate with, and champion the efforts of others.	Skillfully listens to, collaborates with, and champions the efforts of others.	Listens to, collaborates with, and champions the efforts of others.	Sometimes listens to, collaborates with, and champions others' efforts.	Rarely listens to, collaborates with, or champions others' efforts.
Project Presentation (P10)	XI - Technical Writing Skills Ability to communicate the main idea with clarity. Ability to use illustrations properly to support ideas (citations, position on page etc)	Main idea is clearly and precisely stated. Materials are seamlessly arranged in a logical sequence. Illustrations are skillfully used to support ideas	Main idea is understandable. Material moves logically forward. Illustrations are properly used to support ideas	Main idea is somewhat understandable. Material has some logical order and is somewhat coherent or easy to follow. Illustrations are for the most part properly used to support ideas	Main idea is difficult to understand. Material has little logical order, and is often unclear, incoherent. Illustrations are used, but minimally support ideas. (not properly cited etc)
	XII - Communication Skills for Oral Reports Ability to present strong key ideas and supporting details with clarity and concision. Maintain contact with audience, and ability to complete in the allotted time	Presentation is logically and skillfully structured. Key ideas are compelling, and articulated with exceptional clarity and concision. Introduction, supporting details and summary are clearly evident and memorable, and ascertain the credibility of the speaker. Presentation fits perfectly within time constraint.	Presentation has clear structure and is easy to follow. Key ideas are clearly and concisely articulated, and are interesting. There is sufficient detail to ascertain speaker's authority, and presentation includes an introduction and summary. Presentation fits within time constraint, though presenter might have to subtly rush or slow down.	Presentation has some structure. Key ideas generally identifiable, although not very remarkable. Introduction, supporting details and/or summary may be too broad, too detailed or missing. Credibility of the speaker may be questionable at times. Presentation does not quite fit within time constraint; presenter has to rush or slow down at end	Presentation rambles. Not organized; key ideas are difficult to identify, and are unremarkable. No clear introduction, supporting details and summary. Speaker has no credibility. Presentation is unsuitably short or unreasonably long.
Project management (PO11)	XIII - Monitoring and Controlling the Project	Monitors timelines and progress toward project goals on a daily basis. Provides accurate, complete reports of project progress.	Monitors timelines and progress toward project goals most of the time. Provides relatively accurate, complete reports of project progress with only minor errors or omissions	Seldom monitors timelines and progress toward project goals. Provides relatively accurate, yet clearly incomplete, reports of project progress	Does not monitor timelines and progress toward project goals. Provides inaccurate, incomplete reports of project progress
Lifelong Learning (PO12)	XIV - Extend Scope of Work: Ability to extend the project through implementation in other study areas	Demonstrates a skillful ability to explore a subject/topic thoroughly, discusses the road map to extend the project in other areas.	Demonstrates an ability to explore a subject/topic, and shows possible areas in which project can be extended	Demonstrates some ability to explore a subject/topic, providing some knowledge of areas in which project can be extended	Demonstrates minimal or no ability to explore a subject/topic, and does not discuss future work clearly mentioning other areas