

SECURING THE DATA IN FORENSIC DEPARTMENT BY USING CLOUD COMPUTING

Submitted in partial fulfillment of the requirements for the award
of
Bachelor of Engineering degree in Computer Science and Engineering
by

Supriya Sinha, (38110579)
Snigdha Anjali(38110545)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 100 119**

APRIL - 2020



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

” This is to certify that this Project Report is the bonafide work of **Supriya Sinha, (38110579)**

who carried out the project entitled as **SECURING DATA IN FORENSIC DEPARTMENT BY USING CLOUD COMPUTING** under my supervision from November 2020 to April 2021.

Internal Guide

Dr.sathyabama M.E ,Ph.D

Head of the Department

Dr. L. Lakshmanan M.E., Ph.D.,

Dr. S. Vigneshwari M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **Supriya Sinha, (38110579)**
, hereby declare that the Project Report entitled as **SECURING DATA IN
FORENSIC DEPARTMENT BY USING CLOUD
COMPUTING** project done by me under the guidance of **Dr.Sathyabama.**, and at
Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the
requirements for the award of Bachelor of Engineering degree in Computer Science and
Engineering.

DATE:

SIGNATURE OF THE CANDIDATE

PLACE:

ACKNOWLEDGEMENT

I am pleased to acknowledge our sincere thanks to Board of Management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. SASIKALA, M.E., Ph.D., Dean, School of Computing** and also **Dr. S. VIGNESHWARI, M.E., Ph.D., Dr. L. LAKSHMANAN, ME., Ph.D., Heads of the Department, Department of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr .sathaybama** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Client action logs can be an important wellspring of data in cloud criminological examinations; henceforth, guaranteeing the dependability and uprightness of such logs is urgent. Most existing answers for secure logging are intended for customary frameworks as opposed to the intricacy of a cloud situation. In this paper, we propose the Cloud Log Assuring Soundness and Secrecy (CLASS) process as an elective plan for the verifying of logs in a cloud situation. In CLASS, logs are encoded utilizing the individual client's open key with the goal that lone the client can unscramble the substance. So as to counteract unapproved alteration of the log, we produce evidence of past log (PPL) utilizing Rabin's unique finger impression and Bloom channel. Such a methodology diminishes confirmation time altogether. Discoveries from our investigations conveying CLASS in OpenStack exhibit the utility of CLASS in a genuine world context.

TABLE OF CONTENTS		
CHAPTER No	TITLE	PAGE No
	ABSTRACT	V
	LIST OF FIGURES	Vii
1	INTRODUCTION	1
	OVERVIEW	1
	CLOUD COMPUTING	
	SOFTWARE DEVELOPMENT ENVIRONMENT	3
	CLIENT SERVER	3
	WHAT IS CLIENT SERVER	3
	WHY CLIENT SERVER	4
2	LITERATURE SURVEY	5
	CLASS	5
	AN EFFICIENT PRIVACY OUTSOURCED	
	CALCULATION	5
	DATA STORAGE MANAGEMENT	6
	MULTILAYER CLOUD ARCHICTURAL MODEL	7
	CLOUD THROUGH SECURE LOGIN AS A	
	SERVICE	8
3	METHODOLOGY	9
	EXISTING SYSTEM	9
	PROPOSED SYSTEM	9
	TECHNOLOGIES	10
	FEATURES OF THE LANGUAGE USED	10
	ABOUT JAVA	10
	IMPORTANCE OF JAVA	10
	FEATURES OF JAVA	11
	SECURITY	11
	PORTABILITY	11
	BYTE CODE	11
	JAVA VIRTUAL MACHINE	12

	JAVA ARCHITECTURE	13
	COMPILATION OF CODE	13
	JAVASCRIPT	14
	JAVASCRIPT STATEMENTS	15
	JAVASCRIPT VS JAVA	15
	HYPER TEXT MARKUP LANGUAGE	16
	JAVA DATABASE CONNECTIVITY	17
	JAVA SERVER PAGES (JSP)	18
	FEATURE OF JSP	19
	JDBC CONNECTIVITY	20
	MODULE IMPLEMENTATION	21
	PRESERVATION OF LOGS	21
	GATHERER DESIGN	21
	CHECK	21
	MYSTERY KEY SHARING	22
	SYSTEM STUDY	22
	FEASIBILITY STUDY	22
	ALGORITHM	23
	SYSTEM ARCHITECTURE	24
	DATA FLOW DIAGRAM	24
	LEVEL 1	25
	LEVEL 2	25
	SEQUENCE DIAGRAM	26
	USECASE DIAGRAM	27
	CLASS DIAGRAM	28
	FLOW CHART DIAGRAM	29
4	RESULTS AND DISCUSSIONS	30
	WORKING PROCEDURE	30
	DASH BOARD	30
	DATA OWNER	31
	DATA USER	32

	CLOUD SERVICE PROVIDER	32
	PERFORMANCE ANALYSIS	33
	LOG PROCESSING TIME	34
	VERIFICATION TIME	35
5	CONCLUSION AND FUTURE WORK	36
	CONCLUSION	36
	FUTURE WORK	36
	REFERENCES	37
	APPENDIX	39
	A. SOURCE CODE	39
	B. PUBLICATION	

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
3.5	JAVA VIRTUAL MACHINE	13
3.5.2	COMPILATION OF CODE	14
3.13	SYSTEM ARCHITECTURE	24
3.14	DATA FLOW DIAGRAM (LEVEL 0)	24
3.14.1	DATA FLOW DIAGRAM (LEVEL 1)	25
3.14.2	DATA FLOW DIAGRAM (LEVEL 2)	25
3.15	SEQUENCE DIAGRAM	26
3.15.1	USECASE DIAGRAM	27
3.15.2	CLASS DIAGRAM	28
3.15.3	FLOW CHART DIAGRAM	29

LIST OF ABBREVIATIONS

ABBREVIATION

EXPANSION

JSP	JAVA SERVER PAGE
HTML	HYPER TEXT MARKUP LANGUAGE
JDBC	JAVA DATABASE CONNECTIVITY
CSP	CLOUD SERVICE PROVIDER
PPL	PROOF OF PAST LOG

CHAPTER 1

INTRODUCTION

OVERVIEW

CLOUD COMPUTING

CLOUD capacity, security and protection are genuinely settled research regions [1-7], which isn't astounding thinking about the across the board appropriation of cloud administrations and the potential for criminal abuse (for example bargaining cloud records and servers for the taking of delicate information). Curiously however, cloud criminology [8-10] is a generally less gotten subject. If a cloud administration, cloud server, or customer gadget has been undermined or engaged with malignant digital movement (for example used to have unlawful substance, for example, radicalization materials, or lead appropriated disavowal of administration (DDoS) assaults) [11, 12], agents should almost certainly direct criminological investigation so as to "answer the six key inquiries of an occurrence—what, why, how, who, when, and where" [13]. Because of the inborn idea of cloud advancements, regular computerized measurable techniques and devices should be refreshed to hold a similar handiness and relevance in a cloud domain [14]. In contrast to a traditional customer gadget, cloud virtual machines (VMs) can be bolstered by equipment that may be found remotely and, in this manner, would not be physically available (for example out of the jurisdictional region) to an agent. What's more, VMs can be appropriated over various physical gadgets in a grouped situation or they can exist inside a pool of VMs on the equivalent physical parts. In this manner, holding onto the machine for scientific examination isn't reasonable in many examinations. Besides, information dwelling in a VM might be unstable and could be lost once the power is off or the VM ends. Henceforth, the cloud specialist co-op (CSP) assumes a critical job in the accumulation of evidential information (for example cloud client's action log from the log). For instance, the CSP composes the movement log (cloud log) for every client. Along these lines, counteracting alteration of the logs, keeping up a legitimate chain of guardianship and guaranteeing information security is critical [15]. This examination considers "action log information" as any recorded PC occasion that compares to a particular client. Such

information must be kept up privately to preserve client security and to encourage potential insightful exercises.

In 2016, Zawoad et al. proposed a safe logging administration called "SecLaaS" [16] that is intended to gather information from at least one log sources, parse the information and after that store the parsed information in relentless capacity so as to moderate the hazard related with information unpredictability. Before the putting away of information, it encodes the log and creates a log chain to accomplish classification and respectability separately. SecLaaS scrambles the log(s) utilizing the examining office's open key and stores the encoded log(s) in a cloud server. This guarantees security and privacy of the cloud client, except if the specific client is liable to an examination (for example through a court request). To encourage log honesty, SecLaaS produces evidence of past log (PPL) with the log chain and distributes it openly after each predefined age. A trust model was additionally proposed that stores the PPL in different mists to limit the danger of a malignant cloud substance adjusting the log. In any case, in SecLaaS, it is hard to guarantee or confirm that the CSP is composing the right data to the log, or that any data relevant to the examination isn't precluded or altered. In particular, SecLaaS does not give the client the capacity to confirm the exactness of the log (since the log is encoded with the office's open key). At the end of the day, SecLaaS has impediments in tending to responsibility and straightforwardness authorized, particularly from the viewpoint of the client. Broadening SecLaaS, we propose a safe cloud logging plan, Cloud Log Assuring Soundness and Secrecy (CLASS), intended to guarantee CSP responsibility (for example composing the right data to the log) and save the client's security—for example our commitment in this paper. In particular, we incorporate the capacity for the client to check the precision of their log. To do this, the log will be encoded utilizing the client's open key (as opposed to the organization's open key). To abstain from acquainting pointless deferrals with the legal examination, during client enrollment with the cloud administration, both the CSP and the client will all things considered pick an open/private key pair alluded to as substance covering key (CC-key) for the client. The comparing (content disguising) private key will be imparted to different CSPs utilizing Shamir's [17] or Blakley's [18] mystery sharing plans. This would enable the private key to be recovered at whatever point fundamental. We likewise exhibit how

we can use Rabin's unique finger impression [19] and sprout channel in PPL age to set up log veracity. We at that point execute CLASS in OpenStack and assess its exhibition.

SOFTWARE DEVELOPMENT ENVIRONMENT

CLIENT SERVER

With the varied topic in existence in the fields of computers, Client Server is one, which has generated more heat than light, and also more hype than reality. This technology has acquired a certain critical mass attention with its dedication conferences and magazines. Major computer vendors such as IBM and DEC, have declared that Client Servers is their main future market. A survey of DBMS magazine revealed that 76% of its readers were actively looking at the client server solution. The growth in the client server development tools from \$200 million in 1992 to more than \$1.2 billion in 1996. 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 to a more powerful PC, Workstation, Midrange or Main Frames server usually capable of 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 physical 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 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 heterogeneous table joins.

WHAT IS 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 simply 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, Record modifying, Insert, Delete with full relational integrity backup/ restore performance for high volume of transactions, etc. the client server middleware provides a flexible interface between client and server, who does what, when and to whom.

WHY CLIENT SERVER

Client server has evolved to solve a problem that has been around since the earliest days of computing: how best to distribute your computing, data generation and data storage resources in order to obtain efficient, cost effective departmental an 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 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 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 therefore could best be characterized as **"SLAVE-MASTER"**. Time-sharing changed the picture. Remote terminal could view and even change the central data, subject to access permissions. And, as the central data banks evolved in to sophisticated relational database with non-programmer query languages, online users could formulate adhoc queries and produce local reports without adding to the MIS applications software backlog. However remote access was through dumb terminals, and the client server remained subordinate to the Slave\Master.

CHAPTER 2

LITERATURE SURVEY

SURVEY 1

Publication: Cloud Log Assurance Soundness and Secrecy Scheme

Author: M A Manazir Ahsan, Ainuddin Wahid Bin Abdul Wahab, Mohd Yamani Idna Bin Idris, Suleman Khan, Eric Bachura.

- 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.
- propose the Cloud Log Assuring Soundness and Secrecy (CLASS) process as an alternative scheme for the securing of logs in a cloud environment. In CLASS, 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 CLASS in OpenStack demonstrate the utility of CLASS in a real-world context.
- It will provide better security.

SURVEY 2

Publication: An efficient privacy preserving outsourced calculation toolkit with multiple keys.

Author: X. Liu, R. H. Deng, K.-K. R. Choo, and J. Weng.

- we propose a toolkit for efficient and privacy-preserving outsourced calculation under multiple encrypted keys (EPOM).

- Using EPOM, a large scale of users can securely outsource their data to a cloud server for storage. Moreover, encrypted data belonging to multiple users can be processed without compromising on the security of the individual user's (original) data and the final computed results.
- To reduce the associated key management cost and private key exposure risk in EPOM, we present a distributed two-trapdoor public-key cryptosystem, the core cryptographic primitive.
- We also present the toolkit to ensure that the commonly used integer operations can be securely handled across different encrypted domains.
- We then prove that the proposed EPOM achieves the goal of secure integer number processing without resulting in privacy leakage of data to unauthorized parties. Last, we demonstrate the utility and the efficiency of EPOM using simulations.

SURVEY 3

Publication: Data storage management in cloud environments: Taxonomy,survey, and future directions.

Author: Y. Mansouri, A. N. Toosi, and R. Buyya

- Storage as a Service (StaaS) is a vital component of cloud computing by offering the vision of a virtually infinite pool of storage resources.
- It supports a variety of cloud-based data store classes in terms of availability, scalability, ACID (Atomicity, Consistency, Isolation, Durability) properties, data models, and price options.
- Application providers deploy these storage classes across different cloud-based data stores not only to tackle the challenges arising from reliance on a single cloud-based data store but also to obtain higher availability, lower response time, and more cost efficiency.
- Hence, in this article, we first discuss the key advantages and challenges of data-intensive applications deployed within and across cloud-based data stores.

- Then, we provide a comprehensive taxonomy that covers key aspects of cloud-based data store: data model, data dispersion, data consistency, data transaction service, and data management cost.
- Finally, we map various cloud-based data stores projects to our proposed taxonomy to validate the taxonomy and identify areas for future research.

SURVEY 4

Publication: Multi-layer cloud architectural model and ontology-based security service framework for IoT-based smart homes.

Author: M. Tao, J. Zuo, Z. Liu, A. Castiglione

- The Smart Home concept, associated with the pervasiveness of network coverage and embedded computing technologies is assuming an ever-growing significance for people living in the highly developed areas.
- However, the heterogeneity of devices, services, communication protocols, standards and data formats involved in most of the available solutions developed by different vendors, is adversely affecting its widespread application.
- In this paper, promoted by several promising opportunities provided by the advances in [Internet of Things](#) (IoT) and Cloud Computing technologies for facing these challenges, a novel multi-layer cloud [architectural model](#) is developed to enable effective and seamless interactions/interoperations on heterogeneous devices/services provided by different vendors in IoT-based smart home.
- In addition, to better solve the heterogeneity issues in the presented layered cloud platform, ontology has been used as a promising way to address data representation, knowledge, and application heterogeneity, and an ontology-based security service framework is designed for supporting security and [privacy preservation](#) in the process of interactions/interoperations.
- A key benefit of connecting edge and cloud computing is the capability to achieve high-throughput under high concurrent accesses, mobility support, real-time processing guarantees, and data persistency.

- For example, the elastic provisioning and storage capabilities provided by cloud computing allow us to cope with scalability, persistency and reliability requirements and to adapt the infrastructure capacity to the exacting needs based on the amount of generated data.

SURVEY 5

Publication: Towards building forensics enabled cloud through secure logging-as-a-service.

Author: Shams Zawoad, Amit Kumar Dutta, and Ragib Hasan.

- Collection and analysis of various logs (e.g., process logs, network logs) are fundamental activities in computer forensics. Ensuring the security of the activity logs is therefore crucial to ensure reliable forensics investigations.
- However, because of the black-box nature of clouds and the volatility and co-mingling of cloud data, providing the cloud logs to investigators while preserving users' privacy and the integrity of logs is challenging.
- The current secure logging schemes, which consider the logger as trusted cannot be applied in clouds since there is a chance that cloud providers (logger) collude with malicious users or investigators to alter the logs.
- In this paper, we analyze the threats on cloud users' activity logs considering the collusion between cloud users, providers, and investigators. Based on the threat model, we propose Secure-Logging-as-a-Service (SecLaaS), which preserves various logs generated for the activity of virtual machines running in clouds and ensures the confidentiality and integrity of such logs.

CHAPTER 3

METHODOLOGY

EXISTING SYSTEM

SecLaaS scrambles the log(s) utilizing the examining office's open key and stores the encoded log(s) in a cloud server. This guarantees security and secrecy of the cloud client, except if the specific client is liable to an examination (for example through a court request). To encourage log uprightness, SecLaaS creates confirmation of past log (PPL) with the log chain and distributes it openly after each predefined age. A trust model was likewise proposed that stores the PPL in different mists to limit the danger of a vindictive cloud element adjusting the log. Nonetheless, in SecLaaS, it is hard to guarantee or check that the CSP is composing the right data to the log, or that any data relevant to the examination isn't discarded or changed. In particular, SecLaaS does not give the client the capacity to check the precision of the log (since the log is scrambled with the organization's open key).

PROPOSED SYSTEM

Expanding SecLaaS, we propose a protected cloud logging plan, Cloud Log Assuring Soundness and Secrecy (CLASS), intended to guarantee CSP responsibility (for example composing the right data to the log) and protect the client's security. In particular, we incorporate the capacity for the client to check the precision of their log. To do this, the log will be scrambled utilizing the client's open key (as opposed to the organization's open key). To abstain from acquainting pointless postponements with the criminological examination, during client enlistment with the cloud administration, both the CSP and the client will aggregately pick an open/private key pair alluded to as substance disguising key (CC-key) for the client. The relating (content covering) private key will be imparted to different CSPs mystery sharing plans. This would enable the private key to be recovered at whatever point essential. We additionally show how we can use Rabin's unique finger impression and blossom channel in PPL age to set up log veracity. We at that point actualize CLASS in OpenStack and assess its presentation.

TECHNOLOGIES USED

To implement the project, some open source tools have been used such as cloud. The programming languages used to implement the project are JAVA / J2EE (JSP, Servlet), MySQL for database.

Features of The Language Used

In my project, I have chosen Java language for developing the code.

About Java

Initially the language was called as "oak" but it was renamed as "Java" in 1995. The primary motivation of this language was the need for a platform-independent (i.e., 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.

Importance of Java to the Internet

Java has had a profound effect on the Internet. This is because; Java expands the Universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the Server and the Personal computer. They are: Passive information and Dynamic active programs. The Dynamic, Self-executing programs cause serious problems in the areas of Security and probability. But Java addresses those concerns and by doing so, has opened the door to an exciting new form of program called the Applet.

Java can be used to create two types of programs:

Applications and Applets: An application is a program that runs on our Computer under the operating system of that computer. It is more or less like one creating using C or C++. Java's ability to create Applets makes it important. An Applet is an application designed to be transmitted over the Internet and executed by a

Java compatible web browser. An applet is actually a tiny Java program, dynamically downloaded across the network, just like an image. But the difference is, it is an intelligent program, not just a media file. It can react to the user input and dynamically change.

Features of Java

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.

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.

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.

Java Virtual Machine (JVM)

Beyond the language, there is the Java virtual machine. The 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 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 executing of Java code.

Overall Description

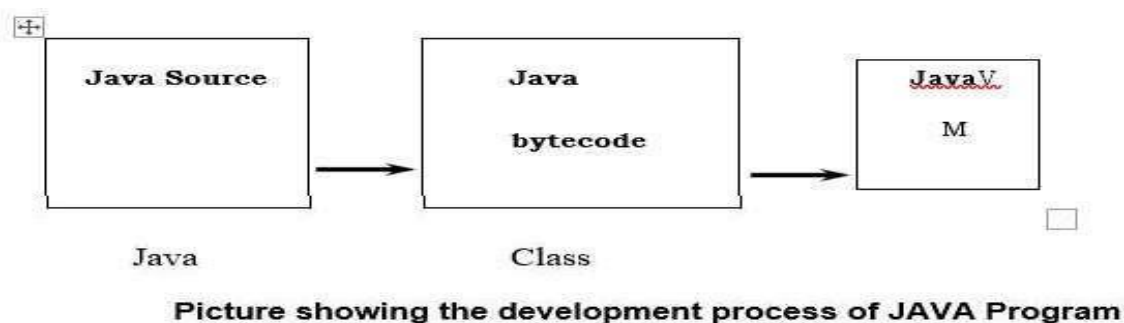


Fig. 3.5 JAVA VIRTUAL MACHINE

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.

Java Architecture

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

Compilation of Code

Compiling and interpreting Java Source Code

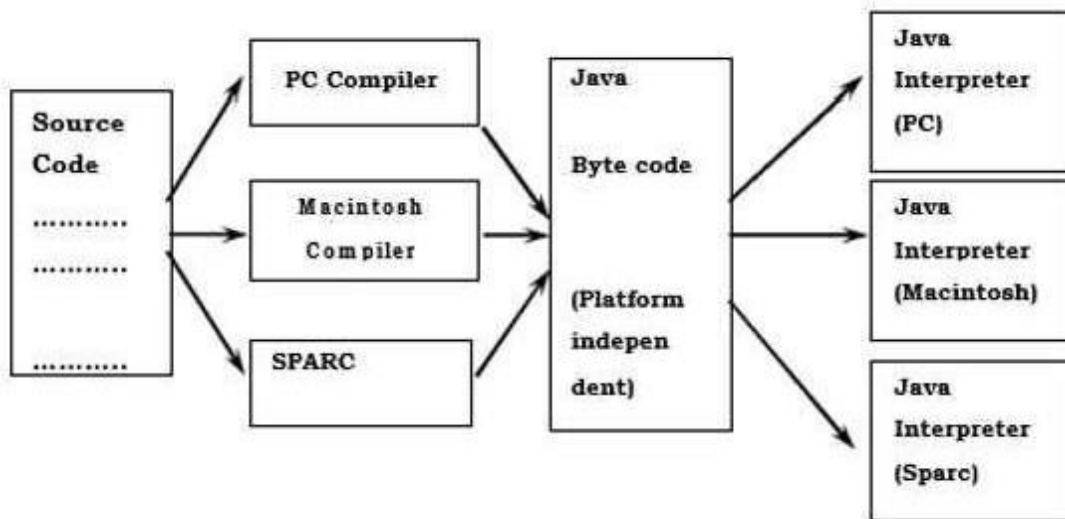


Fig. 3.5.2 COMPILATION OF CODE

During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be a Intel Pentium Windows 95 or Sun SARC station running Solaris or Apple Macintosh running system and all could receive code from any computer through Internet and run the Applets.

Simple

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ programmer, learning Java will be even easier. Because Java inherits the C/C++ syntax and many of the object-oriented features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

Object-Oriented

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank slate. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

Robust

The multi-platform environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and run time. Java virtually eliminates the problems of memory management and deallocation, which is completely automatic. In a well-written Java program, all run time errors can-and should be managed by your program.

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>
```

Here are a few things we can do with JavaScript:

- Validate the contents of a form and make calculations.
- Add scrolling or changing messages to the Browser's status line.
- Animate images or rotate images that change when we move the mouse over them.
- Detect the browser in use and display different content for different browsers.
- Detect installed plug-ins and notify the user if a plug-in is required.

We can do much more with JavaScript, including creating entire application.

JavaScript Vs Java

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.

Advantages

- JavaScript can be used for Server-side and Client-side scripting.
- It is more flexible than VBScript.
- JavaScript is the default scripting language at Client-side since all the browsers support it.

HYPER TEXT MARKUP LANGUAGE

Hypertext Markup Language (HTML), the language of the World Wide Web (WWW), allows users to produce 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 words that lead 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

<code><! -- --></code>	Specifies comments
<code><A>..... </code>	Creates hypertext links
<code>..... </code>	Formats text as bold
<code><BIG>..... </BIG></code>	Formats text in large font
<code><BODY>...</BODY></code>	Contains all tags and text in the HTML document
<code><CENTER>...</CENTER></code>	Creates text
<code><DD>...</DD></code>	Definition of a term
<code><DL>...</DL></code>	Creates definition list
<code>...</code>	Formats text with a particular font
<code><FORM>...</FORM></code>	Encloses a fill-out form
<code><FRAME>...</FRAME></code>	Defines a particular frame in a set of frames
<code><H#>...</H#></code>	Creates headings of different levels
<code><HEAD>...</HEAD></code> document	Contains tags that specify information about a
<code><HR>...</HR></code>	Creates a horizontal rule
<code><HTML>...</HTML></code>	Contains all other HTML tags

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.

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.

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

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 currently 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 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:

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

2. This request is transferred to the Java Webserver. At the server-side Java Webserver receives the request and if it is a request for a JSP file server gives this request to the JSP engine.
3. JSP engine is program which can understand 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 Java Webserver and then it is transferred back to the result is given back to the Java Webserver 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
- Manage 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.

Module Implementation

Preservation of Log & Its Proof

Parser gathers the log from log source. At the point when a log document changes (for example new lines affix) it triggers the parser to check the change and to begin handling new log. Retrieving log from log source, the parser parses the log and gets the important information. Our objective is to keep log substance secure given a parser that will give the framework a log message in string design, paying little mind to content. The arrangement of the log is out of the extent of this work.

Gatherer Design

Sprout channel as a proof of past information ownership, which is neglects to represent Bloom channel's characteristic potential for false positives. When utilizing a Bloom channel procedure, there is an exchange off between the quantity of false positives and the size of the channel. To relieve this issue, a cryptographic single direction collector could be used. In any case, this requires huge computational overhead. In SecLaaS, they utilized their own information structure Bloom Tree that diminished the quantity of false positive occurrences however requires an expanded number of examples of logs and critical computational assets at confirmation time. This is genuine paying little mind to the quantity of sections being checked. Furthermore, despite everything it stays powerless against false positives (though decreased).

Check

Just a check procedure that builds up credibility will most likely decide the nearness of log defilement. There are two kinds of confirmations in our methodology. First is confirmation where the client checks if the CSP is composing right log passages or not. Second is confirmation by any gathering: client, specialist, law implementation expert (LEA) or official courtroom to check PPL to check for log change. In the two cases, the CSP can give a little utility confirmation programming or the client/examiner can get it from individual programming merchant (ISV) to check.

Mystery Key Sharing

We propose, in CLASS, to encode the log with the client's private key (CC-key). In acknowledgment this may prompt lasting loss of log information (notwithstanding for insightful substances), as the private key of a client's CC-key is known uniquely to the client, we propose to share singular client's private key as per Shamir's or Blackley's mystery key sharing procedure among various CSPs. This sharing plan requires institutionalization. We can assemble sharing mists for such a reason when a client buys in to a cloud administration. The CSP and client together pick a couple of open/private key that is known as the substance hiding key (or CC-key) since it is utilized to conceal client's log content.

SYSTEM STUDY

Feasibility Study

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

Three key considerations involved in the feasibility analysis are:

- ☐ ECONOMICAL FEASIBILITY
- ☐ TECHNICAL FEASIBILITY
- ☐ SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

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

technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

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

SOCIAL FEASIBILITY

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

ALGORITHM

CLASS calculations: can be classified into two noteworthy gatherings: One for Log Preservation and one for Proof Accumulation. The Log Preservation calculation can take log passages separately or in a cluster and performs handling before capacity in a log database. This calculation encodes for mystery and produces hash digest for consistency. The Proof Accumulator calculation performs day by day handling of all log passages relating to an IP address to plan and distribute verification of past log (PPL). Shamir's mystery sharing algorithm: Small data will be separated from private key after Shamir's mystery sharing calculation and each part will be shared to one CSP. Subsequent to getting mystery bits of a specific client, the host cloud can recreate the private key to decode the log of that client utilizing Shamir's mystery sharing calculation.

SYSTEM ARCHITECTURE

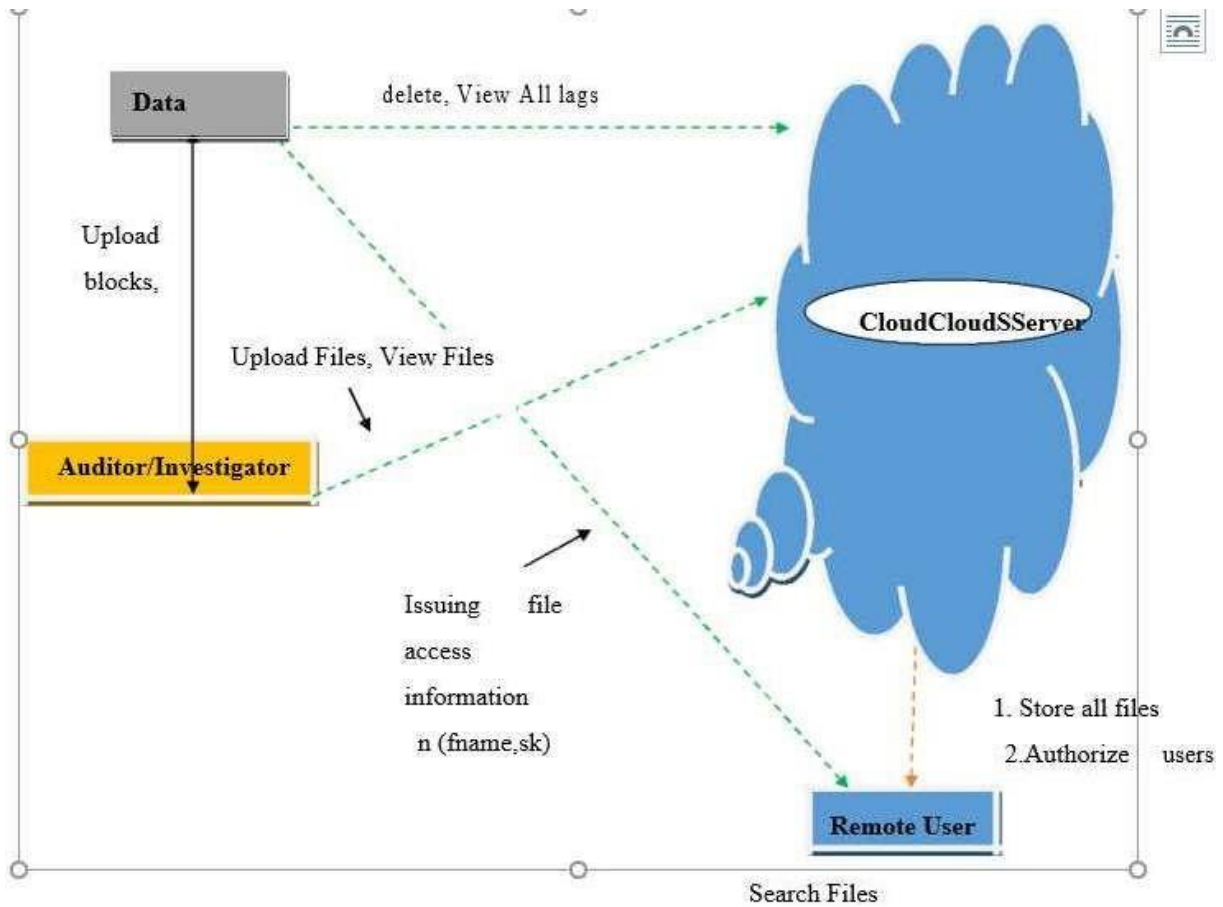


Fig. 3.13 SYSTEM ARCHITECTURE

DATA FLOW DIAGRAM

LEVEL- 0

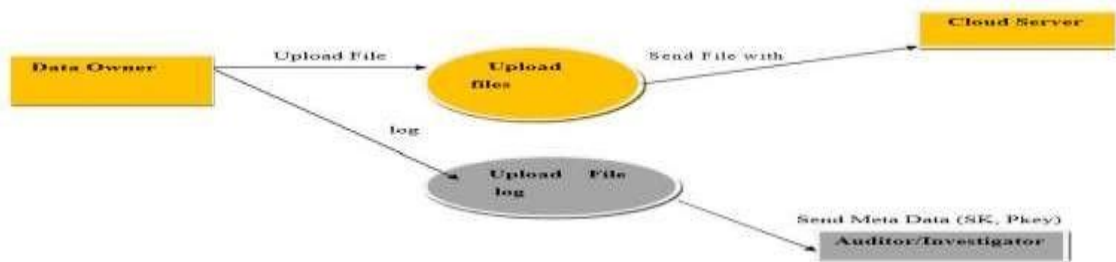


Fig. 3.14 DATA FLOW DIAGRAM

LEVEL- 1

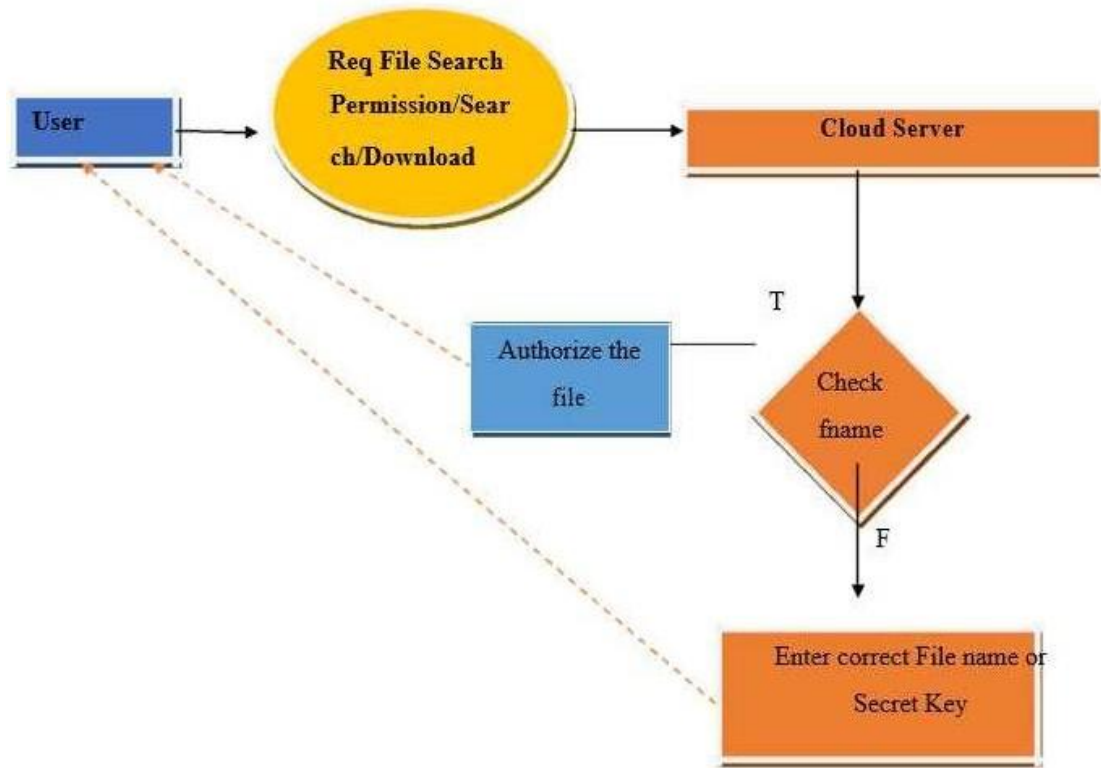


Fig. 3.14.1 LEVEL-1

Level- 2

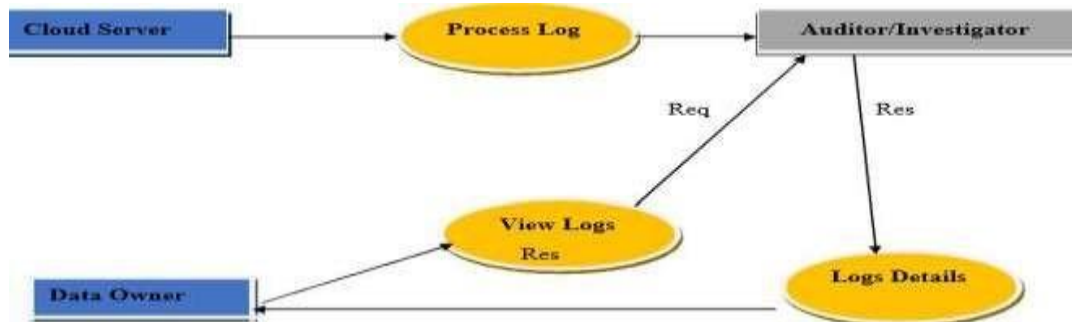


Fig. 3.14.2 LEVEL-2

SEQUENCE DIAGRAM

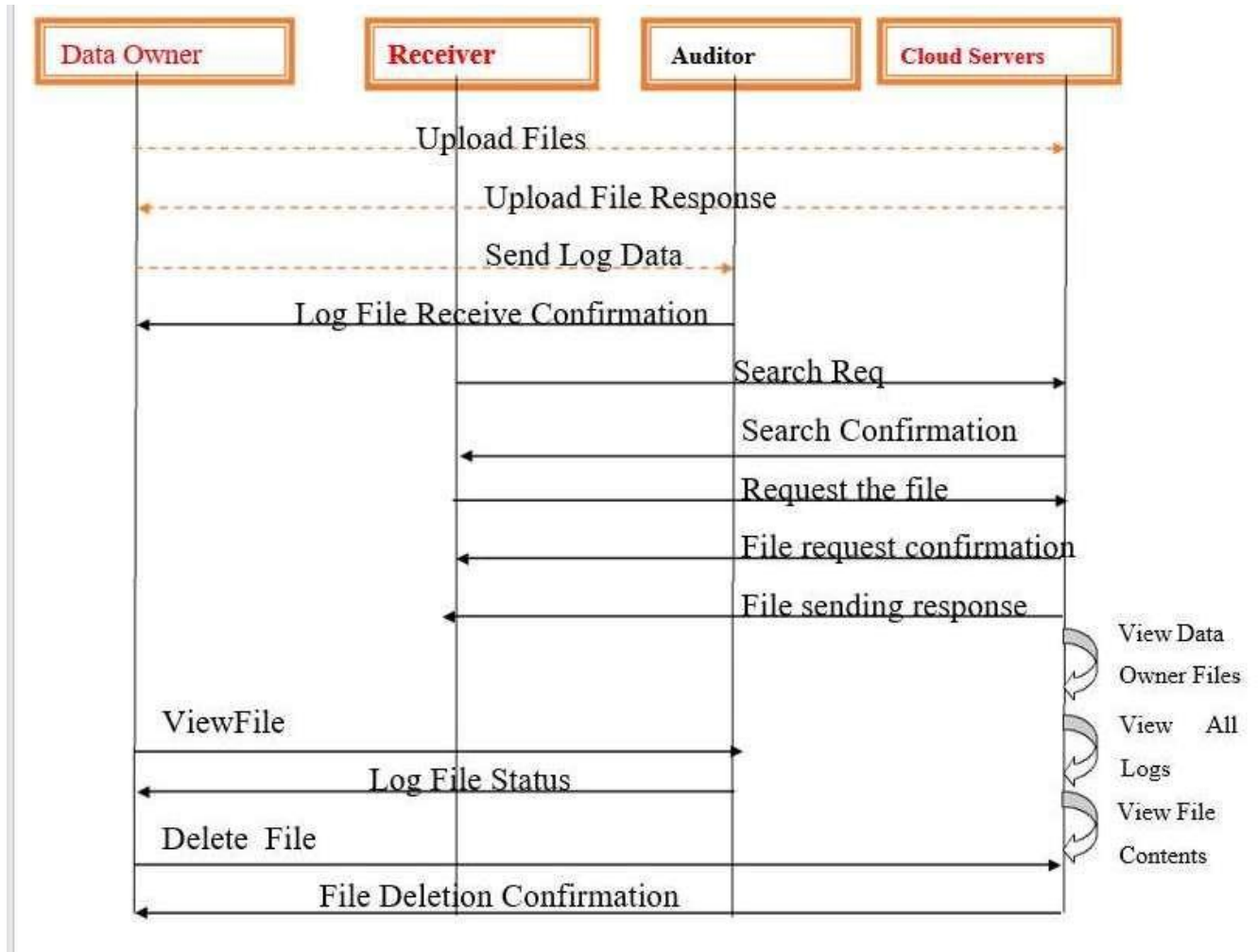


Fig. 3.15 SEQUENCE DIAGRAM

USECASE DIAGRAM

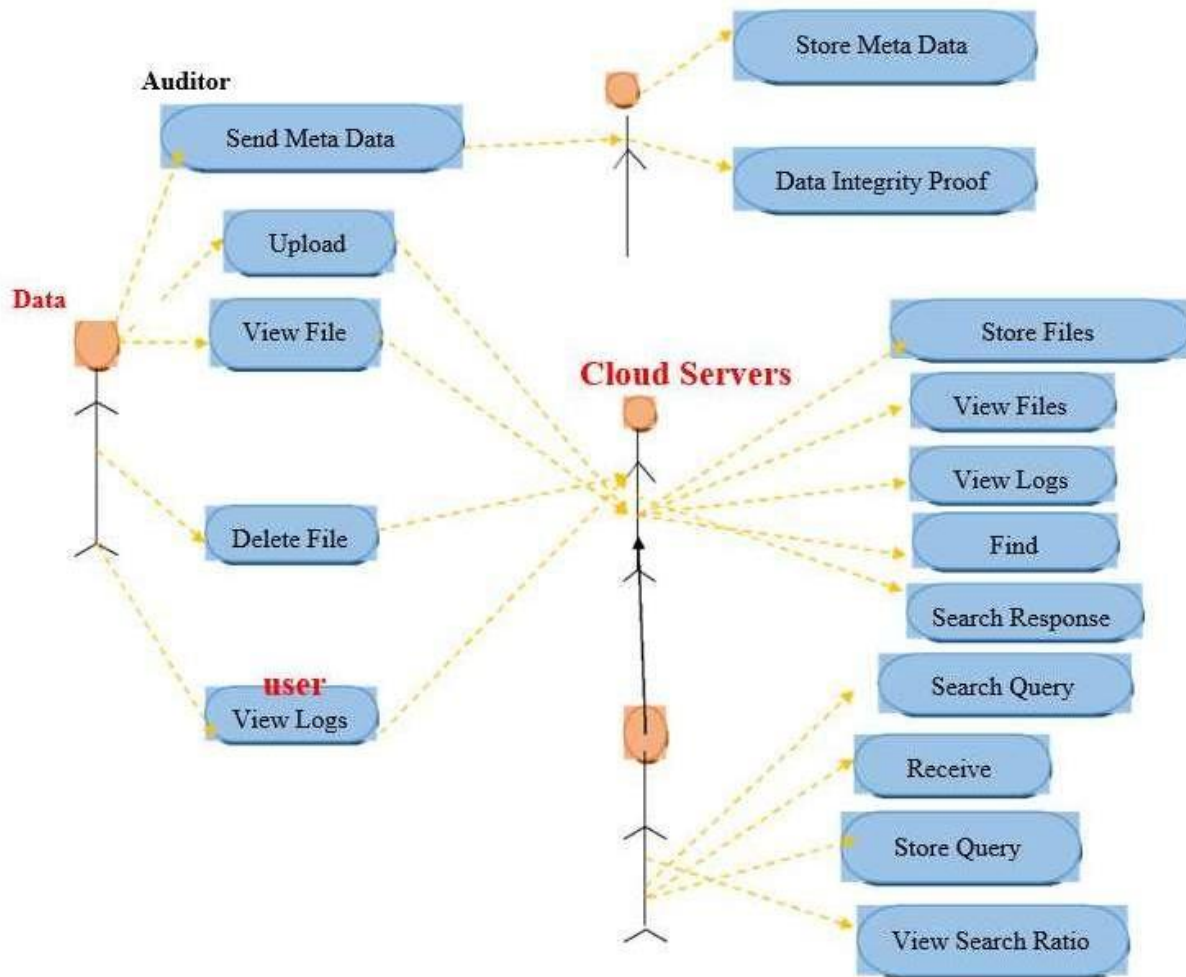


Fig3.15.1 USECASE DIAGRAM

CLASS DIAGRAM

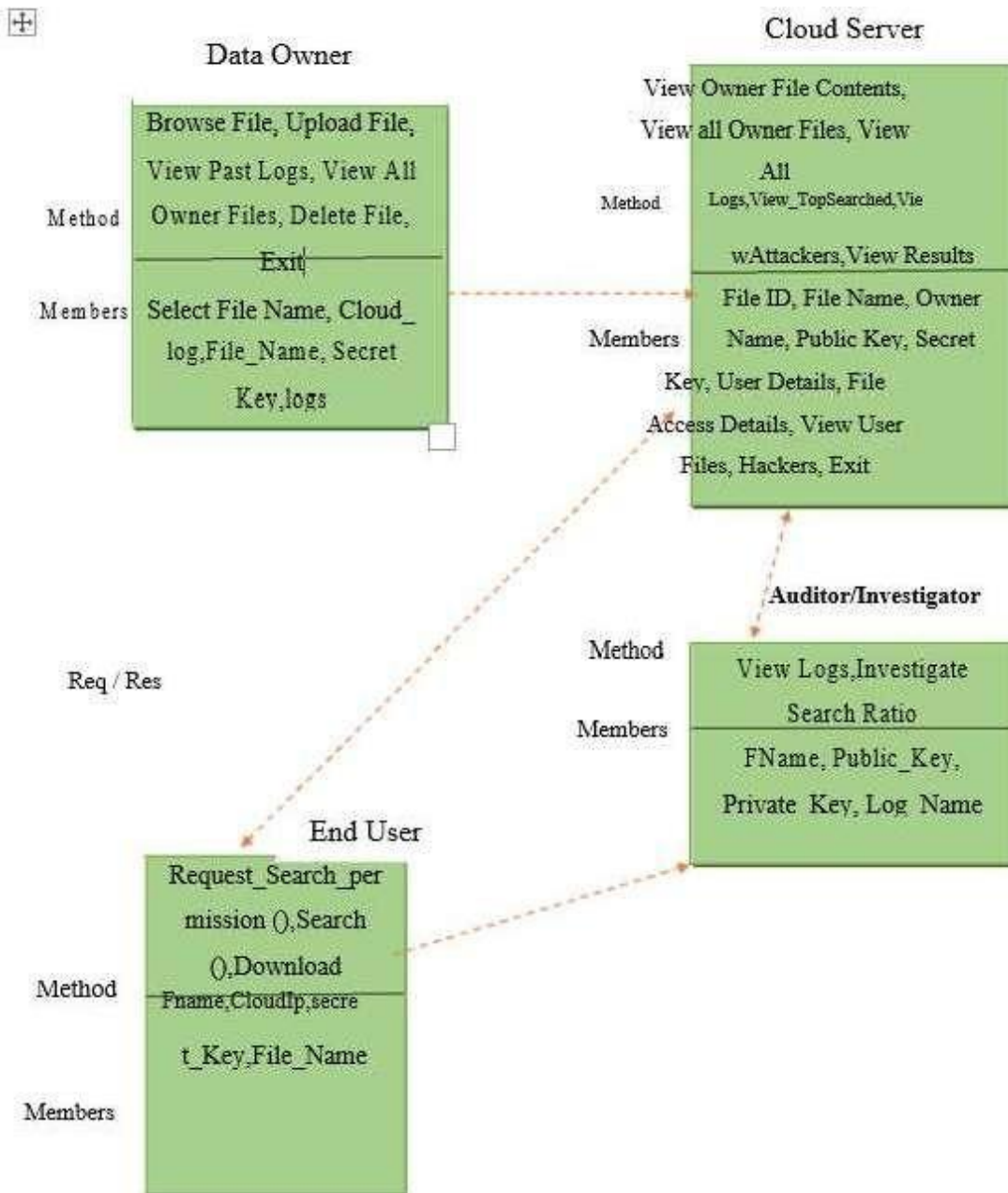


Fig 3.15.2 CLASS DIAGRAM

FLOW CHART DIAGRAM

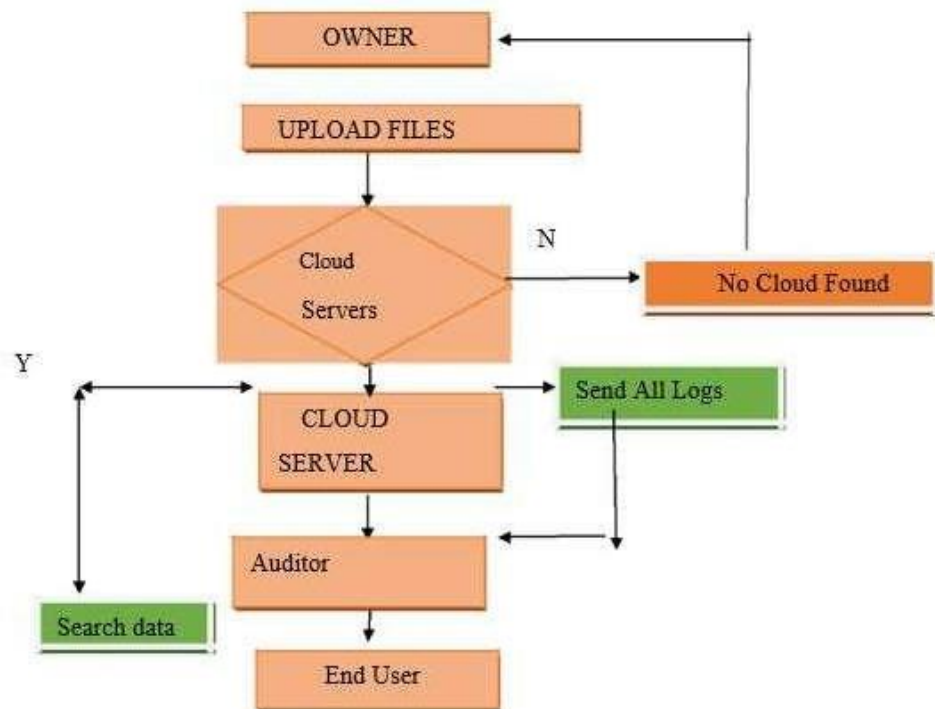


Fig 3.15.3 FLOW CHART DIAGRAM

CHAPTER 4

RESULTS AND DISCUSSIONS

This system consists of home page, Data Owner, Data user, Cloud Service Provider (CSP), Investigator (Auditor) modules. Every module has Authorization for security purpose. First, we need to register for every module, with that register id we need to login into that particular module.

WORKING PROCEDURE

In this project, we have some modules such as Data Owner, Data user, Cloud Service Provider (CSP), Investigator (Auditor).

DASH BOARD

This shows the overview of the contents that are present in the project.



Fig 4.1.1 Dash Board

DATA OWNER

Data Owner module can't perform without registering. After registering we need to login with that register id. Data owner can perform operations like data uploading, data receiving, he can view the attackers, he can delete the files, he can also view the files. With the help of Proof of past Log (PPL) he can also we the attackers.



Fig. 4.1.2 DATA OWNER

DATA USER

Data user module can't perform without registering. We can login with that register id to use that data. Data User module can perform operations like he can search the files, view the files, he can also search the top item who search for more, also he sent request to cloud service provider (CSP).



Fig. 4.1.3 DATA OWNER

CLOUD SERVICE PROVIDER (CSP)

Cloud Service Provider (CSP) module can perform operations like view the authorized owners, view the authorized users, grant the request to the data user, he can also view the time delay documents.

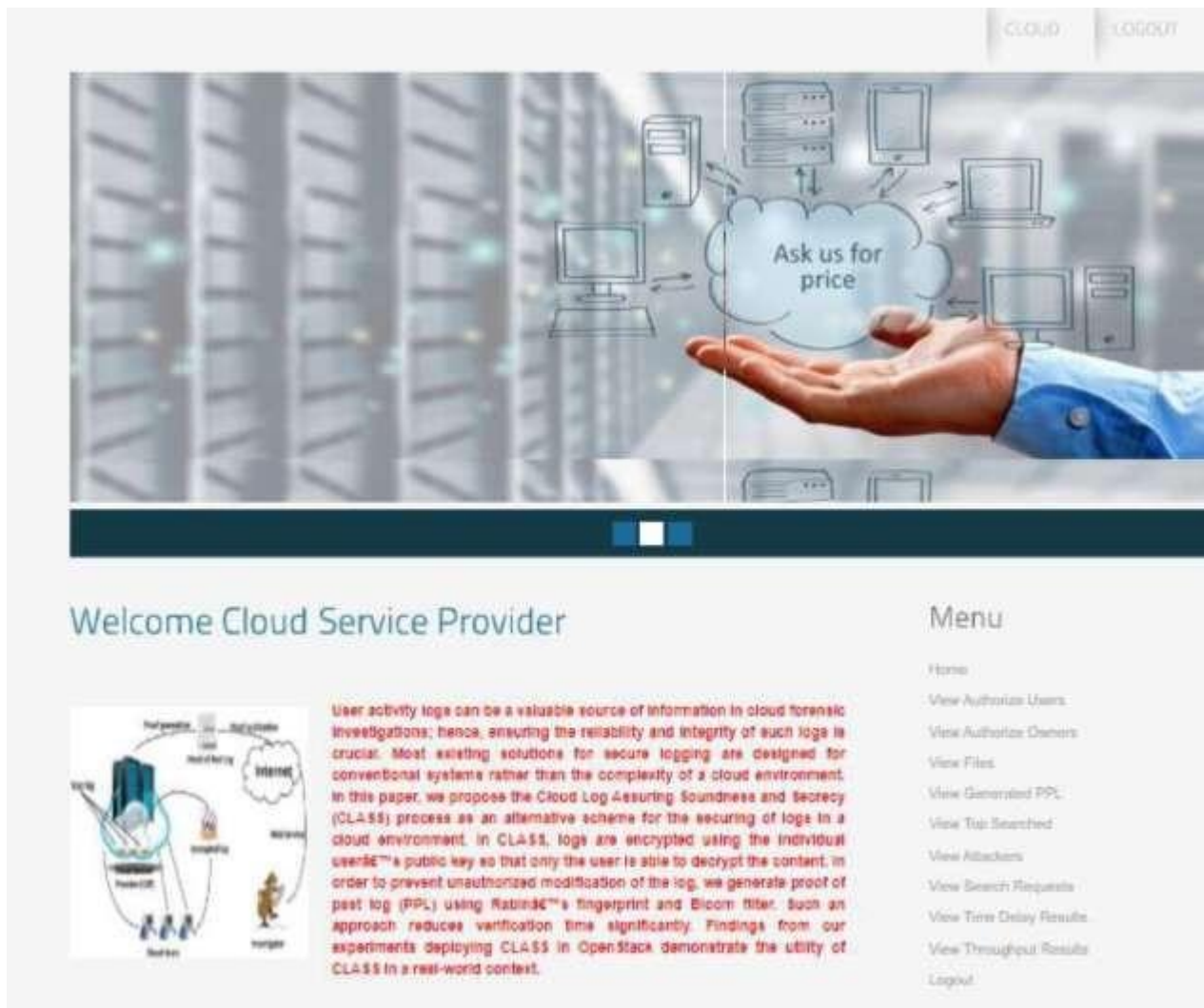


Fig. 4.1.4 CLOUD SERVICE PROVIDER (CSP)

PERFORMANCE ANALYSIS

CLASS differs from SecLaaS scheme mainly in two different ways. Firstly, during processing of log, CLASS incorporates asymmetric encryption with individual user's public key (instead of investigator's public key like in SecLaaS) to avoid data leakage. Secondly, during proof accumulation, CLASS incorporates fingerprinting of all log entries of a single IP address in an epoch to minimize false positive in bloom filter. Our experiment setup suggested that it is feasible to implement the proposed secure cloud logging mechanism. We implemented both schemes and collected some metrics for comparison.

Log Processing Time

We have taken 512KB of the log file with almost 100,000 log entries and have executed both schemes. Similarly, we run on files of 1MB, 1.5MB, 2MB, 2.5MB, 3MB, 3.5MB, 4MB, 4.5MB, 5MB and so on. Our system is performing the same number of operations as the SecLaaS scheme does. Thus, in the execution phase, there is no difference between the CLASS and SecLaaS schemes.

However, if we consider an extra symmetric key encryption (with individual user's secret key) before asymmetric key encryption with investigator's public key to hide data, then processing time would increase minimally in comparison to that of the SecLaaS scheme's time, which is depicted. This shows the difference in execution time between SecLaaS and CLASS is very small which is ratified by the fact that, symmetric key encryption takes a little time. However, we replaced SecLaaS's asymmetric key encryption that uses the investigator's public key with that of individual user's public key. Thus, a number of executed operations remain same and CLASS's time remains same as that of SecLaaS.

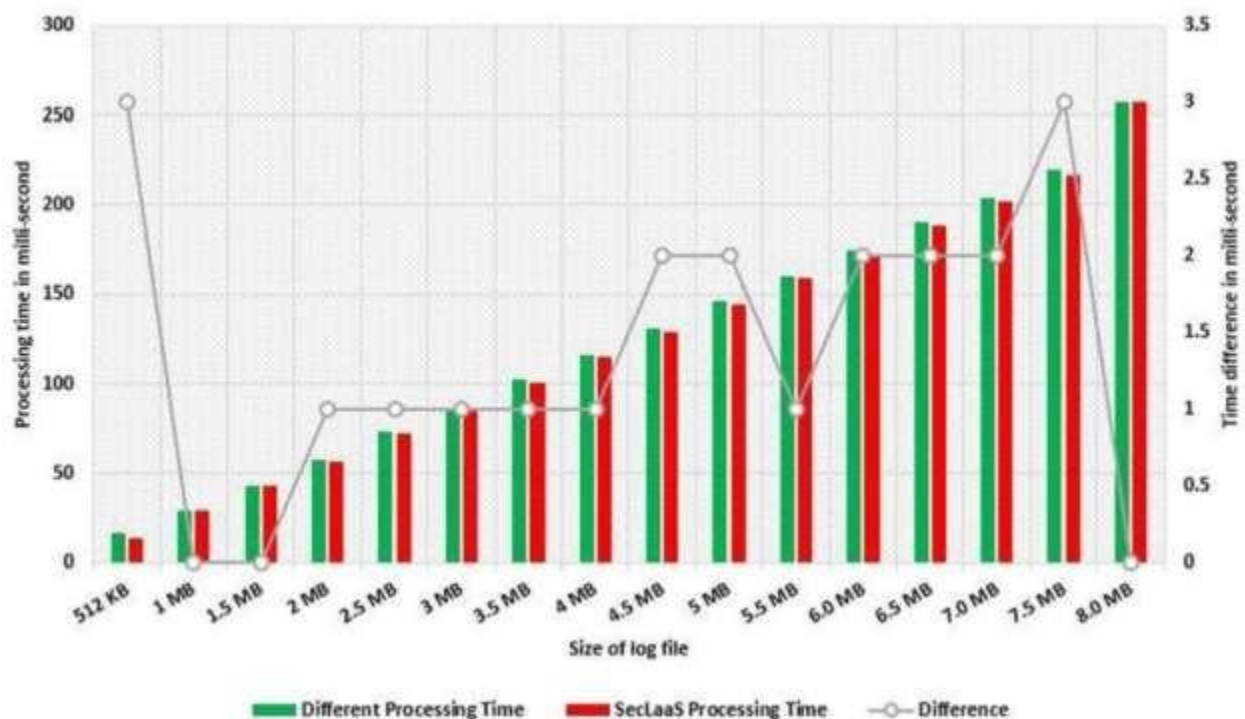


Fig. 4.2.1 log processing time

VERIFICATION TIME

In the case of testifying proof of log, we consider 5000 to 85000 log entries at each epoch with an increment of 5000 log entries at each step. In CLASS, we concatenate each log entries of an epoch and generate the fingerprint. Later, this fingerprint's membership in a bloom filter is calculated and published: for one epoch one fingerprint and one bloom filter. In this case, fingerprint reflects each log entry of an epoch and bloom filter's accuracy improves as only one member resides in a bloom filter. By comparison, SecLaaS generates bloom tree, which is a multi-level bloom filter and inserts each log entry separately into bloom filter. Thus, SecLaaS' verification time is longer than that of CLASS'.

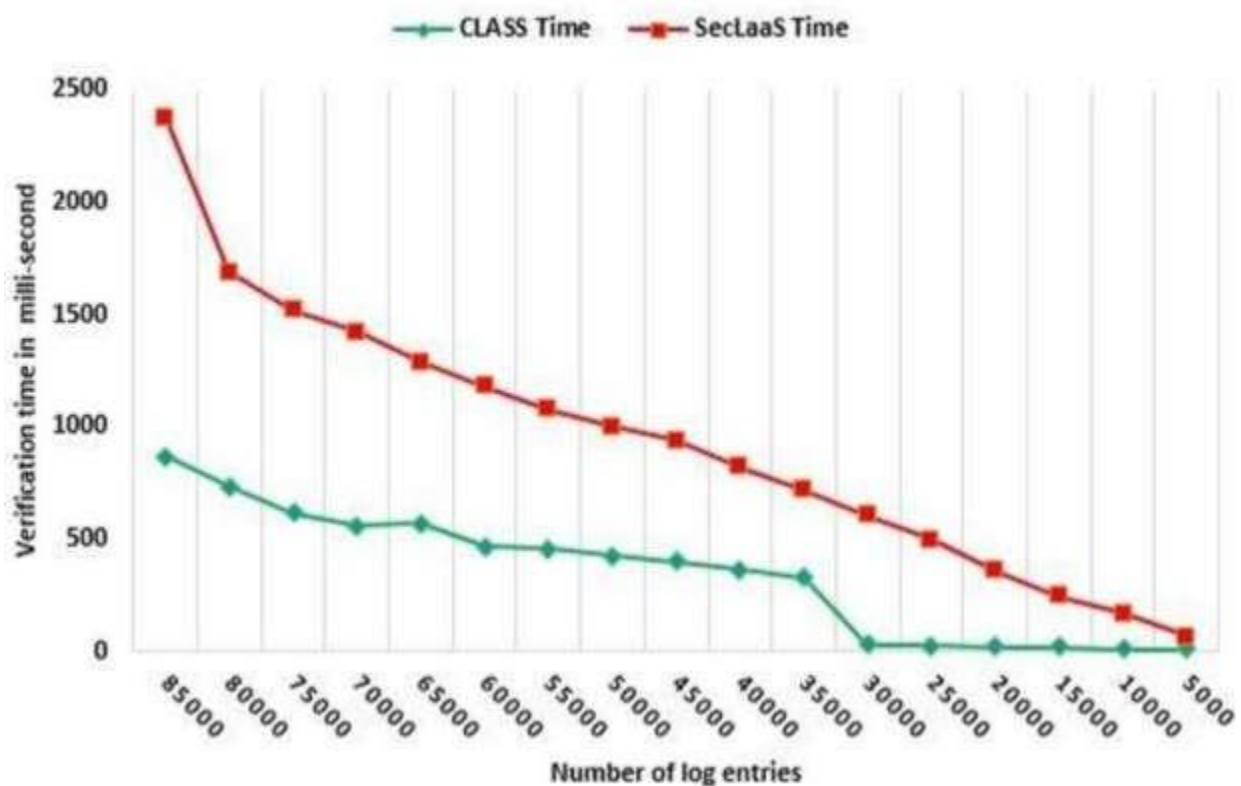


Fig. 4.2.2 VERIFICATION TIME

CHAPTER 5

CONCLUSION AND FUTURE WORK

CONCLUSION

In this paper, we proposed a protected logging plan (CLASS) for distributed computing with highlights that encourage the preservation of client privacy and that moderate the harming impacts of collusion among other parties. CLASS saves the protection of cloud users by encrypting cloud logs with an open key of the respective client while likewise encouraging log recovery in case of an examination. In addition, it guarantees responsibility of the cloud server by enabling the client to recognize any log change. This has the extra impact of anticipating a user from repudiating sections in his very own log once the log has had its PPL built up. Our usage on OpenStack exhibits the achievability and practicality of the proposed plot. The test results demonstrate an improvement in proficiency because of the highlights of the CLASS scheme, particularly in check stage.

FUTURE WORK

Ordinarily logs are low-level information and difficult for the regular client to comprehend what precisely those logs connote. Along these lines, we will investigate utilizing enormous information procedures to encourage client recovery and representation of data from log information. Institutionalization of log arrangement is additionally a related research region. To simplicity looking, we kept some essential and delicate data in plaintext design. This makes them powerless against be introduction. Along these lines, structuring secure and effective accessible encryption would broaden this work. There is likewise the requirement for an online believability framework intended to create trust and validity of a cloud client with the goal that the CSP can empower stricter reviewing approaches for low-trust clients in contrast with high-trust clients. Planning and executing a model of the proposed plan as a team with a real world CSP, with the point of assessing its utility (e.g. execution and adaptability) in a genuine situation.

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] Y. Mansouri, A. N. Toosi, and R. Buyya, "Data storage management in cloud environments: Taxonomy, survey, and future directions," *ACM Computing Surveys (CSUR)*, vol. 50, p. 91, 2017.
- [3] M. Tao, J. Zuo, Z. Liu, A. Castiglione, and F. Palmieri, "Multi-layer cloud architectural model and ontology-based security service framework for IoT-based smart homes," *Future Generation Computer Systems*, vol. 78, pp. 1040-1051, 2018.
- [4] Z. Xia, Y. Zhu, X. Sun, Z. Qin, and K. Ren, "Towards privacy-preserving content-based image retrieval in cloud computing," *IEEE Transactions on Cloud Computing*, pp. 276-286, 2018.
- [5] 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.
- [6] 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.
- [7] L. Li, R. Lu, K.-K. R. Choo, A. Datta, and J. Shao, "Privacy-preserving-outsourced association rule mining on vertically partitioned databases," *IEEE Transactions on Information Forensics and Security*, vol. 11, pp. 1847-1861, 2016.
- [8] K.-K. R. Choo, M. Herman, M. Iorga, and B. Martini, "Cloud forensics: State-of-the-art and future directions," *Digital Investigation*, pp. 77-78, 2016.
- [9] C. Esposito, A. Castiglione, F. Pop, and K.-K. R. Choo, "Challenges of Connecting Edge and Cloud Computing: A Security and Forensic Perspective," *IEEE Cloud Computing*, vol. 4, pp. 13-17, 2017.

- [10] Z. Qi, C. Xiang, R. Ma, J. Li, H. Guan, and D. S. Wei, "ForenVisor: A tool for acquiring and preserving reliable data in cloud live forensics," *IEEE Transactions on Cloud Computing*, vol. 5, pp. 443-456, 2017.
- [11] C. Hooper, B. Martini, and K.-K. R. Choo, "Cloud computing and its implications for cybercrime investigations in Australia," *Computer Law & Security Review*, vol. 29, pp. 152-163, 2013.
- [12] O. Osanaiye, K.-K. R. Choo, and M. Dlodlo, "Distributed denial of service (DDoS) resilience in cloud: review and conceptual cloud DDoS mitigation framework," *Journal of Network and Computer Applications*, vol. 67, pp. 147-165, 2016.
- [13] N. H. Ab Rahman, W. B. Glisson, Y. Yang, and K.-K. R. Choo, "Forensic-by-design framework for cyber-physical cloud systems," *IEEE Cloud Computing*, vol. 3, pp. 50-59, 2016.
- [14] B. Martini and K.-K. R. Choo, "An integrated conceptual digital forensic framework for cloud computing," *Digital Investigation*, vol. 9, pp. 71-80, 2012.
- [15] S. Khan, A. Gani, A. W. A. Wahab, M. A. Bagiwa, M. Shiraz, S. U. Khan, et al., "Cloud log forensics: foundations, state of the art, and future directions," *ACM Computing Surveys (CSUR)*, vol. 49, p. 7, 2016.
- [16] S. Zawoad, A. K. Dutta, and R. Hasan, "Towards building forensics enabled cloud through secure logging-as-a-service," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, pp. 148-162, 2016.
- [17] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, pp. 612-613, 1979.
- [18] G. R. Blakley, "Safeguarding cryptographic keys," *Proc. of the National Computer Conference* 1979, vol. 48, pp. 313-317, 1979.
- [19] M. O. Rabin, *Fingerprinting by random polynomials: Center for Research in Computing Techn., Aiken Computation Laboratory, Univ.*, 1981.
- [20] F. Anwar and Z. Anwar, "Digital forensics for eucalyptus," in *Frontiers of Information Technology (FIT)*, 2011, 2011, pp. 110-116.

APPENDIX

A. SOURCE CODE

Authentication.jsp Code

```
<title>Authentication Page</title>

<%@      page      language="java"      contentType="text/html;
      charset=ISO-
8859pageEncoding="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.spec.SecretKeySpec,org.bouncycastle.util.encoders.Base64"%> <%@page
import="java.sql.*,java.util.Random,java.io.PrintStream,java.io.FileOutputStream,java.i
o.FileInputStream,java.security.DigestInputStream,java.math.BigInteger,java.security.
MessageDigest,java.io.BufferedInputStream"%> <%@page
import="java.security.Key,java.security.KeyPair,java.security.KeyPairGenerator,javax.c
rypto.Cipher"%>

<%@pageimport="java.util.*,java.text.SimpleDateFormat,java.util.Date,java.io.FileInpu
tStream,java.io.FileOutputStream,java.io.PrintStream"%> <%

String name = request.getParameter("userid"); String pass =
request.getParameter("pass");

try {

String sql = "SELECT * FROM cloud where name='" + name+ "' and pass='" + pass + "'
";

Statement stmt = connection.createStatement(); ResultSets = stmt.executeQuery(sql);
if (rs.next()==true)
```

```

{
response.sendRedirect("C_Main.jsp");
}
else
{
response.sendRedirect("wronglogin.html");
}
}
catch (Exception e)
{
out.print(e);
e.printStackTrace();
}
%>

```

ResgisterAuthentication.jsp Code

```

<title>Registration authen</title>

<%@page
import="com.oreilly.servlet.*,java.sql.*,java.lang.*,java.text.SimpleDateFormat,java.util
.*,java.io.*,javax.servlet.*,javax.servlet.http.*"
%> <%@ page import="java.sql.*"%> <%@
include file="connect.jsp" %>

<%@ page import="java.util.Date" %>

<%
ArrayList list = new ArrayList();
ServletContext context = getServletContext();
String dirName =context.getRealPath("Gallery\\");
String paramname=null;
String file=null;

```

```

String a=null,b=null,c=null,d=null,image=null;
String ee[]=null;
String checkBok=" ";
int ff=0;
String bin = "";
String uname=null;
String pass=null;
String email=null;
String mno=null;
String addr=null;
String dob=null;
String location=null;
String status="Waiting";
String gender=null;
String pincode=null;
FileInputStream fs=null;
File file1 = null;
try {
    MultipartRequest multi = new MultipartRequest(request, dirName, 10 * 1024 * 1024); //
    10MB

    Enumeration params = multi.getParameterNames(); while (params.hasMoreElements())

    {
        paramname = (String) params.nextElement();
        if(paramname.equalsIgnoreCase("userid"))
        {uname=multi.getParameter(paramname);

        }

        if(paramname.equalsIgnoreCase("pass"))

```

```
{  
pass=multi.getParameter(paramname);  
}  
if(paramname.equalsIgnoreCase("email"))  
{  
email=multi.getParameter(paramname);  
}  
if(paramname.equalsIgnoreCase("mobile"))  
{  
mno=multi.getParameter(paramname);  
}  
if(paramname.equalsIgnoreCase("address"))  
{  
addr=multi.getParameter(paramname);  
}  
if(paramname.equalsIgnoreCase("dob"))  
{  
dob=multi.getParameter(paramname);  
}  
if(paramname.equalsIgnoreCase("gender"))  
{  
gender=multi.getParameter(paramname);  
}  
if(paramname.equalsIgnoreCase("pincode"))  
{  
pincode=multi.getParameter(paramname);  
}  
if(paramname.equalsIgnoreCase("location"))
```

```

{
location=multi.getParameter(paramname);
}
if(paramname.equalsIgnoreCase("pic"))
{
image=multi.getParameter(paramname);
}
}
int f = 0;
Enumeration files = multi.getFileNames();
while (files.hasMoreElements())
{
paramname = (String) files.nextElement();
if(paramname != null)
{
f = 1;
image = multi.getFilesystemName(paramname); String fPath =
context.getRealPath("Gallery\\"+image);

file1 = new File(fPath);

fs = new FileInputStream(file1);
list.add(fs);

String ss=fPath;
FileInputStreamfis = new FileInputStream(ss); StringBuffer sb1=new StringBuffer();
int i = 0;

while ((i = fis.read()) != -1) {

if (i != -1) {

//System.out.println(i);

```

```

String hex = Integer.toHexString(i);
// session.put("hex",hex);
sb1.append(hex);
// sb1.append(",");
String binFragment = "";
int iHex;
for(int i1= 0; i1 <hex.length(); i1++){ iHex =
Integer.parseInt(""+hex.charAt(i1),16); binFragment =
Integer.toBinaryString(iHex);

while(binFragment.length() < 4)

{

binFragment = "0" + binFragment;
}
bin += binFragment;
//System.out.print(bin);
}
}
}
}
}

FileInputStream fs1 = null;
//name=dirName+"\\Gallery\\"+image;
int lyke=0;
//String as="0";
//image = image.replace(".", "_b.");
String query1="select * from downer where name='"+uname+"'";
Statement st1=connection.createStatement();

```

```

ResultSet rs1=st1.executeQuery(query1);
if ( rs1.next() )
{
out.println("Data Owner Name Already Exits"); %>

<p><a href="DO_Login.jsp">Back</a>
<a href="index.html">Home</a></p>

<%
}
else
{
PreparedStatementps=connection.prepareStatement("insert into
downer(name,pass,email,mobile,addr,dob,gender,pin,location,imageess,status)
values(?,?,?,?,?,?,?,?,?,?,?)");
ps.setString(1,uname);
ps. setString(2, pass);
ps.setString(3,email);
ps.setString(4,mno);
ps.setString(5,addr);
ps.setString(6,dob);
ps.setString(7,gender);
ps.setString(8,pincode);
ps.setString(9,location);
ps.setString(11,"Waiting");
ps.setBinaryStream(10,(InputStream)fs,(int)(file1.length()));
//
                                ps.setString(11,sk);
                                if(f == 0)
                                ps.setObject(10,null);

```

```

else if(f == 13)
{
    fs1 =
    (FileInputStream)list.get(0)
    ;
    ps.setBinaryStream(10,fs1
    ,fs1.available());
}
int x=ps.executeUpdate();
if(x>0){
    out.println("Registered
    Sucessfully");
    %>

    <p><a
    href="DO_Login.jsp">Back
    </a> <a
    href="index.html">Home</
    a></p>

    <%
    }
    }
    }
    catch (Exception e)
    {
        out.println(e.getMessage()
        );
        e.printStackTrace();
    }
    %>

```

```

fs1 = (FileInputStream)list.get(0);
ps.setBinaryStream(10,fs1,fs1.available());
}
int x=ps.executeUpdate();
if(x>0){
out.println("Registered Sucessfully");
%>
<p><a href="DO_Login.jsp">Back</a> <a
href="index.html">Home</a></p>

<%
}
}
}
catch (Exception e)
{
out.println(e.getMessage());
e.printStackTrace();
}
%>

```

Login.jsp 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>Cloud Login</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}

-->

</style>

</head>

<body>

<div class="main">

<div class="header">

<div class="header_resize">

<div class="logo">

<h1><a href="index.html" class="style1">CLASS: Cloud Log Assuring Soundness and
Secrecy Scheme for Cloud Forensics</a></h1> </div>

<div class="menu_nav">

<ul>

<li><a href="index.html"><span>Home </span></a></li>

<li><a href="DO_Login.jsp"><span>Data Owner </span></a></li>
<li><a href="DU_Login.jsp">Data User </a><a href="about.html"></a></li>
<li class="active"><a href="C_Login.jsp"><span>Cloud</span></a></li>
<li><a href=""></a></li>

</ul>

```



```

<input type="password" id="pass" name="pass" class="box" />
<p>&nbsp;</p>
<p>
<input name="imageField" type="submit" class="LOGIN" id="imageField" value="Login"
/>
<input type="reset" name="imageField" id="imageField" class="RESET" />
</p>
<p>&nbsp;</p>
</form>
<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="index.html">Home</a></li>

<li><a href="C_Login.jsp"><span>Cloud</span></a></li>

<li><a href="DU_Login.jsp">Data User </a><a href="about.html"></a></li>
<li><a href="DO_Login.jsp"><span>Data Owner </span></a></li> </ul>

</div>
<div class="gadget">
<h2 class="star"><br />
</h2>

```

```
</div>
</div>
<div class="clr"></div>
</div>
</div>
<div class="fbg">
</div>
<div align=center></div>
</body>
</html>
```

SECURING THE LOGS IN FORENSIC DEPARTMENT BY USING CLOUD COMPUTING

M. ANILKUMAR REDDY, M. VIJAY SAI, A. VEERAMUTHU

[1][2] Department of Computer Science, Satyabhama Institute of Science and Technology, Chennai.

[3] Department of Information Technology, Satyabhama Institute of Science and Technology, Chennai.

ABSTRACT: - *Customer movement logs can be a critical wellspring of data in cloud criminological evaluations from this time forward, guaranteeing the dependability and uprightness of such logs is desperate. Generally existing responses for secure logging are planned for standard systems instead of the unpredictability of a cloud circumstance. Right now, propose the Cloud Log Assuring Soundness and Secrecy process as an elective arrangement for the confirming of logs in a cloud circumstance. In logs are encoded using the individual customer's open key with the objective that solitary the customer can unscramble the substance. In order to balance unapproved change of the log, we produce proof of past log using Rabin's one of a kind finger impression and Bloom channel. Such a procedure decreases affirmation time out and out. Disclosures from our examinations passing on CLASS in OpenStack show the utility of CLASS in a certified world setting.*

I INTRODUCTION

CC (Cloud computing) is one of the main innovations lately. Numerous organizations captivated about the distributed computing as a result of these primary attributes like ease in utilizing, freedom of area, execution and dependability and so forth. Due to these advantages, all organizations no compelling reason to squander cash on equipment and they can arrangement their business effectively, Cloud administrations give three significant

conveyance models those are PaaS i.e. Platform as an assistance, IaaS i.e. Infrastructure as a Service and SaaS i.e. Software as a help.

We can say in ordinary use, the expression "the cloud" is basically a figure of speech for the Internet. Advertisers have made advanced the expression "in the cloud" to says programming, stages and framework that are sold "as a help" to the end clients, for example remotely through the Internet. Ordinarily, the seller has genuine energy consuming servers which have items and administrations from a remote area, so end-clients don't need to make a fuss over server upkeep and so on; they can just sign on to the system without introducing anything they can get to administrations.

The significant models of distributed computing administration are referred to as stage as an assistance, Software as a Service and Infrastructure as a service. Google, Amazon, IBM, Oracle Cloud, Rackspace, Salesforce, Zoho and Microsoft Azure are some notable cloud sellers. These cloud administrations might be offered in an open, private, cross breed, arrange. These models offers various types of assistance, Infrastructure as a help gives server-facilitating administration right now need to pay and no compelling reason to stress over upkeep of equipment and different things identified with administration. Distinctive kind of conveyance model gives various advantages. IaaS just likes server-facilitating administrations; however, end users needn't bother with pay sum for equipment and upkeep them any longer. They advantage

much from the nature of being versatile and flexibility. PaaS resembles administration facilitating; however, end-users don't have to stress over the servers or not ready to reaction to enormous number of solicitations. They advantage much from the achievement and unwavering quality.

SaaS looks the Representational State Transfer (REST) without a doubt, and makes end-users profit by execution, multi-occupancy engineering and a wide range of highlights. The entirety of the three models share a shortcoming from the attributes of the distributed computing. As end-users put their sensible procedurals on the cloud, that implies they don't claim the control of the equipment especially for PaaS and SaaS. These are not well disposed to advanced legal. Since Conventional computerized scientific is exceptionally relying upon the media seized from the wrongdoing scene. Now, it ought to be change or improve for distributed computing to be more amiable with advanced criminology.

Because of the inborn idea of cloud advances, customary computerized scientific techniques and instruments should be refreshed to hold a similar convenience and relevance in a cloud domain [14]. In contrast to an ordinary customer gadget, cloud virtual machines be upheld equipment may be found automatically and consequently would not be truly available (for example out of the jurisdictional domain) to an examiner. What's more, VMs can be circulated over different physical gadgets in a grouped domain or they can exist inside a pool of VMs on the equivalent physical segments. Along these lines, holding onto the machine for criminological examination isn't suitable in many examinations.

Moreover, information living in virtual machine might be unpredictable and could lost once the force is off or the virtual machine ends.

Subsequently, cloud service provider assumes the vital job in assortment off suggestive information. For instance, the cloud service provider composes movement log for every client. Subsequently, forestalling adjustment of the logs, keeping up a legitimate chain of guardianship and guaranteeing information security is essential [15]. These explorations considered as "movement log information" recorded PC occasion that relates to particular client. This information should be kept up privately to preserver client protection and to encourage potential analytical exercises.

In 2016, Zawoad et al. suggested protected logging administration known as "SecLaaS" [16] that is intended to gather information from at least one log sources, parse the information and afterward store the parsed information in determined capacity so as to relieve the hazard related with information unpredictability. Preceding the putting away of information, it scrambles the log and produces a log chain to accomplish secrecy and trustworthiness individually.

SecLaaS scrambles the log(s) utilizing the exploring organization's open key and stores the encoded logs in a cloud. This guarantees security and classification of cloud client, except if the specific client is dependent upon an examination (for example by means of a court request). To encourage log uprightness, SecLaaS produces evidence of past log (PPL) with the log chain and distributes it openly after each predefined age. A assurance model was additionally recommended that saves the proof of past log in different mists to limit the danger a malevolent cloud substance adjusting the log. Notwithstanding, in SecLaaS, it is hard to guarantee or check the CSP is composing right data to log, or any data appropriate to examination isn't overlooked or adjusted.

In particular, SecLaaS doesn't give the client the capacity to confirm the precision of log. At the end of the day, SecLaaS has confinements in tending to responsibility and straightforwardness upheld, particularly from the point of view of the client. Broadening SecLaaS, we suggest a safe cloud authentication plan, Cloud Log Assuring Soundness and Secrecy scheme (CLASS), intended a guarantee CSP responsibility (for example composing the right data to the log) and save the client's security for example our commitment right now. In particular, we incorporate the capacity to the client for checking the precision of the log. To do that, the log would be encoded utilizing the client's open key (instead of the organization's open key).

To abstain from acquainting pointless postponements with the measurable examination, during client enrollment with the cloud administration, both the CSP and the client will on the whole pick an open key pair alluded to as substance covering key for client. Comparing (Data disguising) secret key would be imparted to different CSP utilizing Shamir [17] or Blakley [18] mystery transferring plans. This will permit the secret key to recovered at whatever point vital. We likewise exhibit how we could use Rabin unique finger impression [19] blossom channel in proof of past log age set up log accuracy. We at that point actualize CLASS in OpenStack and assess its exhibition.

II RELATED WORK

Right now globalization, concentrating is the way to progress for the little organizations or those equitable on their start, for the most part for those data innovation (IT) industry. At this circumstance, redistributing is an ideal path for dynamic achievement, as they would more be able to focus on the center business. From the start level, they can re-appropriate non-basic

programming or modules to different organizations to decrease the hazard and HR of the organization. The other one is to be redistributed ought to be the administrations or the equipment for figuring and capacity.

What's more, as I would see it, this is only the first driver to the famous of the distributed computing. Distributed computing is virtual figuring condition which can give, stage and programming support as remote administrations. Furthermore, those administrations are gotten to by Internet everywhere throughout the world by buyers. Distributed computing isn't a glossary in data innovation as it won't supplant the IT organizations totally. It is progressively similar to an alias the new sort of business activity mode named "pay-scrutinize". Be that as it may, it includes with numerous new advances, for instance virtual registering, circulated figuring, equal registering, enormous information handling and so forth. The CSP i.e. Cloud Service Provider keeps up huge scope PC frameworks in bunches, and for the most part has huge capacity limit on Data Center.

The CSP give successful framework to meeting the clients' prerequisites on programming, administration, registering and capacity in an adaptable mode. So we can believe that the client organizations redistribute a few sorts of prerequisite to those CSPs. Anwar *et al.* [20] endeavored address a portion off these difficulties by distinguishing chance of Syslog or grunt log to aid the location of cloud assaults.

The creators led cloud scientific examinations dependent on logs produced on Eucalyptus, an open-source distributed computing programming. In particular, they created own dataset by recreating a DDoS assault on Eucalyptus and recognized the assaulting machine IP address breaking down the log. Security, get to control, and check of log were not used. *Patrascu et al.* [21] proposed a

criminological module be kept up as a major aspect of the cloud's controller module, which intended to speak with an alternate heap of cloud resources (for example virtual document framework, virtual memory, organize stack, and framework call interface) so as to gather and store logs. Likewise, security of the gathered log was not considered. In a previously undermined framework (for example after an enemy has acquired access to cloud server mystery key), is past the point where it is possible to cryptographically guarantee that the unbound log information has not been changed.

Bellare et al. [22] presented the idea of forward uprightness or advance mystery as a framework property to alleviate an aggressor's capacity to defile a logging framework without recognition. Tainting incorporates addition of bogus logs, adjustment or cancellation of previous logs, and reorder of logs. Honesty is built up utilizing a cryptographically solid single direction hash work (for example HMAC) and a mystery key that fills in as an underlying purpose of a chain of a same arbitrary capacity (PRF). As it were, each progressive log section has a related hash key that is reliant upon the past passage. **Schneir et al. [23]** proposed a log the board plot dependent on forward uprightness and gave a few real-world model applications.

Right now, labels are related with each and every log passage, specifically: one for half confided in log aggregator and another is for confided in checker. Utilizing labels, they will had the option for guarantee secure stream uprightness as opposed to sending security. Additionally, a label tag can be sanction by any log preceding it by a specific age; consequently, the least label of an age could 'affirm' whole chain of log sections to that age. It will be causes extra calculation costs during the check stage. The plans talked about so far are intended for a regular

framework, as opposed to a cloud domain. What's more, none of the plans thinks about a noxious lumberjack.

Beam et al. [26] introduced a structure, where lumberjack (or) log aggregator send progression of logs to cloud server by means of confirmed medium. At that point, cloud would found way to look after secrecy, trustworthiness, undeniable nature, and accessibility of secure logs. To keep truncation blunder from the two finishes, they utilized unique kind of logs on beginning and consummation purposes of every square of log. The shield logs from securing ruptures or protection infringement, they may encode log sections with a chain of successively created keys and save respectability they to utilize other arrangement keys produced similarly.

Be that as it may, there is no alternative for open evidence because of the utilization of symmetric key encryption to ensure classification and security. Confirmation of log brings about a security infringement to those leading check. **Zawood et al. [16]** proposed the novel plan "SecLaaS" [16], that mulled over a large number of the CSP's characteristic shortcomings, for example, unpredictability, blending of cloud information, and potential vindictive lumberjacks. They had given total arrangement of security logging for every cloud exercise of VMs (or customers). We can assess SecLaaS, recognizing various shortcomings. We propose an elective methodology that steadily expands upon SecLaaS trying to react to these recognized shortcomings.

III EXISTING SYSTEM

SecLaaS scrambles the log(s) using the looking at office's open key and recorded the encoded logs in a cloud. This establish security, mystery of cloud customer, aside from if the particular customer is at risk to an assessment (for instance through a court demand). To empower

log uprightness, SecLaaS makes affirmation of proof of past log (PPL) with log chain and it disperses transparently after every pretend age. A hope model similarly suggests that stores the PPL in various fogs can restrain the hazard of noxious cloud component changing log. In every case, in SecLaaS, is difficult to ensure and watch that CSP is creating right data to the log, or that any information applicable to assessment isn't disposed of or changed. Specifically, SecLaaS doesn't enable the customer to check the exactness of log.

IV PROPOSED SYSTEM

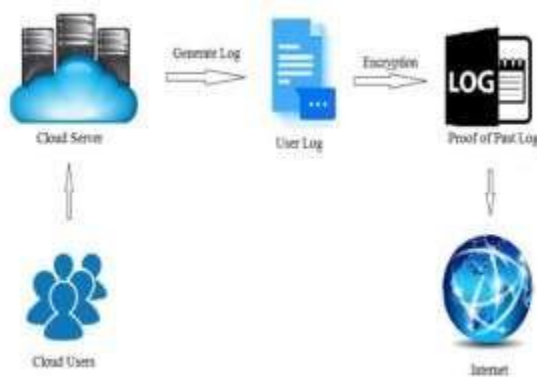


Fig. 1. Over View of the Proposed System

Extend SecLaaS, we suggest an ensured cloud authentication plan, Cloud Log Assuring Soundness and Secrecy, expected to ensure CSP obligation and secure the customer's security. Specifically, we consolidate the limit with respect to the customer to check the correctness of their log. for doing this, the log will be mixed using the customer's open key (slightly than the association's open key). Which is shown in figure.1. To swear off familiarizing trivial deferments with the criminological assessment, during customer selection with the cloud organization, both the CSP and the customer will aggregately pick an open/private key pair suggested as substance masking key (CC-key) for customer. The related secret key will be granted to various cloud service provider puzzle sharing

plans. This would empower the private key to be recuperated at whatever point fundamental. We moreover show how we can utilize Rabin's extraordinary finger impression and bloom direct in PPL age to set up log accuracy. By complete CLASS in in stock and survey the introduction.

V MODULE IMPLEMENTATION

PRESERVATION OF LOG & ITS PROOF

Parser accumulates the log from log source. Right when log report changing (for instance new line join) it causes the parser to verify the change and to start taking care of new log. Recover log from the log source, the passer passes the log and got the significant data. Us goal to keep log substance security given passer that will give system a log notification in the string configuration, paying little psyche to content. The course of action of the log is out of the degree of this work.

GATHERER DESIGN

Grow channel is a proof of past information possess, which is fails to speak to Bloom channel's trademark possible for bogus positives. While utilizing a Bloom channel method, there a tradeoff in between the amount of bogus positive and the size of channel. To moderate this issue, a make secret single course authority could be utilized. Regardless, it requires immense computational overhead. In SecLaaS, they will use their own data structure Bloom Tree that reduces the amount of bogus positive events anyway required an extended number of instance of logs and basic computational resources at affirmation time. This is certified paying little psyche to the amount of areas being checked. Moreover, regardless of all that it remains feeble against bogus positives (however diminished).

CHECK

Only a check method that develops believability will no doubt choose the proximity of log debasement. There will two sorts of affirmations in my strategy. one is affirmation where customer checks if CSP is creating right data sections or not. another is affirmation by social event: customer, authority, law execution master (LEA) or authority court to check PPL to check for log change. In the two cases, the CSP can give little usefulness affirmation programming or the customer/inspector can get from singular programming shipper to check.

MYSTERY KEY SHARING

We suggest, in CLASS, to encode log with customer's private key. In affirmation it may provoke enduring waste of log data (despite of smart substances) the private key of customer's secret key is known exceptionally to customer; we suggest to send particular customer private key according to Shamir or Bleckley puzzle key share methodology between different CSP. This share arrangement required standardization. We may gather sharing fogs for such an explanation when customer purchases in to a cloud organization. The CSP and customer together pick two or three open/secret key that is known as the substance concealing key since it is used to hide customer's log information.

Table. 1. Comparison between SecLaas scheme and our Proposed scheme

Challenge / Threat	SecLaas Scheme	CLASS Scheme
Modification of cloud -log after publishing proof of past log (PPL)	Detected	Detected
Modification of cloud -log before publishing PPL	Remains undetected	User avails a way to detect it beforehand
User's privacy violation either by cloud server or by investigator	Privacy preserved	Privacy preserved
Repudiation of cloud-log by cloud-server	Detected	Detected
Repudiation of cloud-log by cloud-user	Undetected	detected
Log recovery for investigation	Recovered	Recovered even when the log files were encrypted with user's private key.

In this, we confer the details for execution of our proposed system and follow with analysis and performance.

PERFORMANCE ANALYSIS

In this, the performance is done in two different methods. One is log processing time and another one is verification time.

LOG PROCESSING TIME

We take 512KB of the log file with almost 11akh log entries and we have implemented both the tasks. Similarly, we execute files of 1MB ,1.5MB,2MB,2.5MB,3MB, and so on. Our scheme is performing the same number of operations as SecLass does which is shown in Figure2, so there is no difference between CLASS and SecLasss.

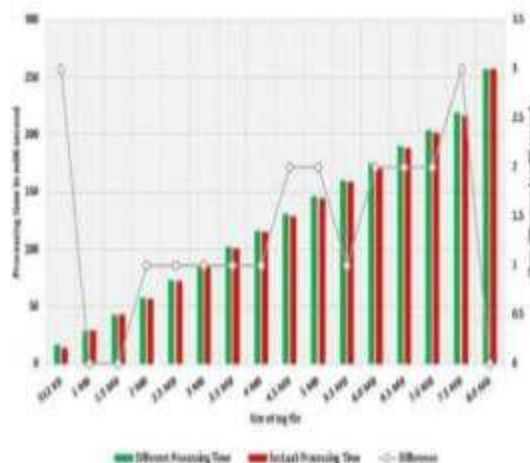


Fig. 2. Log processing Time

VERIFICATION TIME

In the case of testifying proof of log, we consider 5000 to 85000 log entries at each epoch with an increment of 5000 log entries at each step. In CLASS, we concatenate each log entries of an epoch and generate the fingerprint. Later, this fingerprint's membership in a bloom filter is calculated and published: for one epoch one fingerprint and one bloom filter is shown in Figure3, in this case, fingerprint reflects each log entry of an epoch and bloom filter's accuracy improves as only one member resides in a bloom filter.

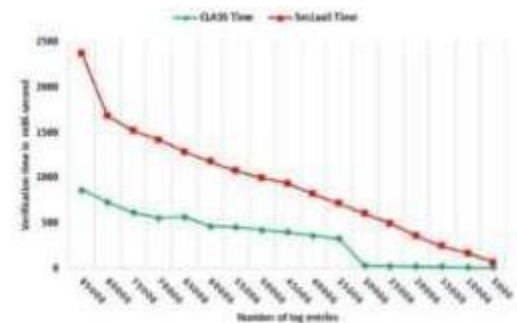


Fig. 3. Verification Time Comparison

VII RESULTS AND DISCUSSIONS

The following are the results of our proposed system:



Fig. 4. Home Page

In Figure 4, home page it contains all the modules of our proposed system like data owner, data user, cloud service provider and auditor.



Fig. 5. Data Owner

In Figure 5, Data owner can upload the data, view the data. He can also view the attacker.



Fig. 6. Cloud Service Provider

Cloud service provider can view who is authorized and unauthorized users. It will generate the PPL and also it accepts the search requests from the data user which is shown in Fig. 6.

VII CONCLUSION

Right now, proposed a secured logging plan for dispersed figuring with the features that energize safeguarding of customers protection and that moderate hurting effects of arrangement between different gatherings. CLASS spares insurance of cloud clients by scrambling cloud logs with an open key of individual customer although in like manner empowering log recuperation if there should be an occurrence of an assessment. What's more, this ensures duty of cloud server by empowering customer to perceive any log change. It has another effect of envisioning client from disavowing areas in their own special log once log has had its PPL developed. Our utilization on OpenStack shows feasibility and reasonableness of the planned plot. Test outcomes shows improvement in capability on account of the features of the CLASS plot, especially under control arrange.

REFERENCE

- [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] Y. Mansouri, A. N. Toosi, and R. Buyya, "Data storage management in cloud environments: Taxonomy, survey, and future directions," *ACM Computing Surveys (CSUR)*, vol. 50, p. 91, 2017.
- [3] M. Tao, J. Zuo, Z. Liu, A. Castiglione, and F. Palmieri, "Multi-layer cloud architectural model and ontology-based security service framework for IoT-based smart homes," *Future Generation Computer Systems*, vol. 78, pp. 1040-1051, 2018.

- [4] Z. Xia, Y. Zhu, X. Sun, Z. Qin, and K. Ren, "Towards privacy-preserving content-based image retrieval in cloud computing," *IEEE Transactions on Cloud Computing*, pp. 276-286, 2018.
- [5] 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.
- [6] 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.
- [7] L. Li, R. Lu, K.-K. R. Choo, A. Datta, and J. Shao, "Privacy-preserving-outsourced association rule mining on vertically partitioned databases," *IEEE Transactions on Information Forensics and Security*, vol. 11, pp. 1847-1861, 2016.
- [8] K.-K. R. Choo, M. Herman, M. Iorga, and B. Martini, "Cloud forensics: State-of-the-art and future directions," *Digital Investigation*, pp. 77-78, 2016.
- [9] C. Esposito, A. Castiglione, F. Pop, and K.-K. R. Choo, "Challenges of Connecting Edge and Cloud Computing: A Security and Forensic Perspective," *IEEE Cloud Computing*, vol. 4, pp. 13-17, 2017.
- [10] Z. Qi, C. Xiang, R. Ma, J. Li, H. Guan, and D. S. Wei, "ForenVisor: A tool for acquiring and preserving reliable data in cloud live forensics," *IEEE Transactions on Cloud Computing*, vol. 5, pp. 443-456, 2017.
- [11] C. Hooper, B. Martini, and K.-K. R. Choo, "Cloud computing and its implications for cybercrime investigations in Australia," *Computer Law & Security Review*, vol. 29, pp. 152-163, 2013.
- [12] O. Osanaiye, K.-K. R. Choo, and M. Dlodlo, "Distributed denial of service (DDoS) resilience in cloud: review and conceptual cloud DDoS mitigation framework," *Journal of Network and Computer Applications*, vol. 67, pp. 147- 165, 2016.
- [13] N. H. Ab Rahman, W. B. Glisson, Y. Yang, and K.-K. R. Choo, "Forensic-by-design framework for cyber-physical cloud systems," *IEEE Cloud Computing*, vol. 3, pp. 50-59, 2016.
- [14] B. Martini and K.-K. R. Choo, "An integrated conceptual digital forensic framework for cloud computing," *Digital Investigation*, vol. 9, pp. 71-80, 2012.
- [15] S. Khan, A. Gani, A. W. A. Wahab, M. A. Bagiwa, M. Shiraz, S. U. Khan, et al., "Cloud log forensics: foundations, state of the art, and future directions," *ACM Computing Surveys (CSUR)*, vol. 49, p. 7, 2016.
- [16] S. Zawoad, A. K. Dutta, and R. Hasan, "Towards building forensics enabled cloud through secure logging-as-a-service," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, pp. 148-162, 2016.
- [17] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, pp. 612-613, 1979.
- [18] G. R. Blakley, "Safeguarding cryptographic keys," *Proc. of the National Computer Conference* 1979, vol. 48, pp. 313-317, 1979.
- [19] M. O. Rabin, "Fingerprinting by random polynomials: Center for Research in Computing

Techn., Aiken Computation Laboratory, Univ., 1981.

[20] F. Anwar and Z. Anwar, "Digital forensics for eucalyptus," in *Frontiers of Information Technology (FIT)*, 2011, 2011, pp. 110-116.

[21] A. Patrascu and V.-V. Patriciu, "Logging system for cloud computing forensic environments," *Journal of Control Engineering and Applied Informatics*, vol. 16, pp. 80-88, 2014.