DATA LEAKAGE AND SECURITY ON CLOUD COMPUTING

A PROJECT REPORT

Submitted by

DANUSU B	**[711719104019]**
HARI PRASATH S	**[711719104030]**
JEGADEESWARAN R	**[711719104035]**
MOHANKUMAR S	**[711719104305]**

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

KGiSL INSTITUTE OF TECHNOLOGY, SARAVANAMPATTI

ANNA UNIVERSITY :: CHENNAI 600 025

MAY 2023

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"DATA LEAKAGE AND SECURITY ON CLOUD COMPUTING"** is the bonafide work of "**DANUSU B, HARI PRASATH S, JEGADEESWARAN R, MOHANKUMAR S**" who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.THENMOZHI T, M.E., M.B.A., Ph.D.,	**Mr. RAJINIGANTH R,M.E.,**
HEAD OF THE DEPARTMENT,	**SUPERVISOR,**
Computer Science and Engineering,	Assistant Professor,
KGiSL Institute of Technology,	Computer Science and Engineering,
Saravanampatti,	KGiSL Institute of Technology,
Coimbatore-641035.	Saravanampatti,
	Coimbatore-641035.

Submitted for the Anna University Viva-Voice examination held on____________

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

We express our deepest gratitude to our **Chairman and Managing Director Dr. ASHOK BAKTHAVACHALAM** for providing us with an environment to complete our project successfully.

We are grateful to our CEO, Academic Initiatives **MR. ARAVIND KUMAR RAJENDRAN,** our beloved Director-Academics **Dr. P. SHANKAR** and our honorable Secretary **Dr. N. RAJKUMAR**. We are thankful to our beloved Principal **Dr. M. SELVAM, M.E., Ph.D.,** and Vice-Principal **Dr. S. SURESH KUMAR M.E., Ph.D.,** for their support,valuable guidance and blessings.

We would like to thank **Dr. T. THENMOZHI, M.E.,M.B.A., Ph. D,** Head of the Department, Department of Computer Science and Engineering for her unwavering support during the entire course of this project work. We express our sincere thanks to **Mrs.V. KAMALA, M.E.,** our Project Coordinator, Assistant Professor, Department of Computer Science and Engineering who modelled us both technically and morally for achieving greater success in completing this project work.

We express our sincere and heartful thanks to our faculty guide **Mr.R.RAJINIGANTH, M.E.,** Assistant Professor, Department of Computer Science and Engineering for his constant encouragement and support throughout our course, especially for the useful suggestions given during the course of the project period and being instrumental in the completion of our project with his complete guidance.

We also thank all the **Faculty members** of our department and finally, we take this opportunity to extend our deep appreciation to our **Family** and **Friends**, for all they meant to us during the crucial times of the completion of our project.

ABSTRACT

The popularity of cloud computing technology has become the standard for IT deployments in the enterprise, education, and government sectors. However, cloud technologies such as the hypervisor or web dashboard have vulnerabilities that could potentially lead to data exfiltration. The impact of a data breach is enormous. Data breaches at companies like precisely exposed 340 million customer records. The incident also resulted in financial loss to, reputational damage, loss of customer trust, and compliance issues with the firm. Most of the data was processed via a Third-party application (TPA) and users are unaware of the security basics. To detect data leaks, this document proposes a method that includes performing a packet capture on the platform. The proposed model also protects the data before downloading via the TPA, and after successful download of the data the secret key is sent to the original user without being intercepted by the TPA. In some cases, we can also inject "realistic but fake" data records to further improve our chances of detecting leakage and identifying the guilty party.

Keywords: Cloud Computing, Data Leakage, Data Security, CSP, SLA.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

API	Application Programming Interface
CSP	Cloud Services Provider
CSS	Cascading Style Sheet
DBMS	DataBase Management System
HTML	Hypertext Markup Language
JS	JavaScript
PHP	Hypertext PreProcessor
SHA2	Secure Hash Algorithm 2
SLA	Service License Agreement
SQL	Structured Query Language
TPA	Third-Party Application

CHAPTER 1

INTRODUCTION

Data loss refers to data loss that occurs on any device that stores it. This is problem for anyone who uses a computer. Data loss occurs when data is intentionally or unintentionally removed physically or logically from an organization. Data loss has become the largest problem in organizations today, and organizations are responsible for addressing this problem. Data leakage is defined as the accidental or unintentional distribution of private or sensitive data to an unauthorized entity. Data leakage poses a serious issue for companies as the number of incidents and the cost to those experiencing them continue to increase. Data leakage is enhanced by the fact that transmitted data including emails, instant messaging, website forms, and file transfers among others, are largely unregulated and unmonitored on their way to their destinations.

The main scope of this module is providing complete information about the data/content that is accessed by the users within the website. Forms Authentication technique is used to provide security to the website in order to prevent the leakage of the data. Continuous observation is made automatically and the information is send to the administrator so that he can identify whenever the data is leaked.

1.1 PROBLEM DEFINITION:

Current Data Leakage Detection Project propose data allocation strategies that improve the probability of identifying leakages. In some cases, we can also inject "realistic but fake" data records to further improve our chances of detecting leakage and identifying the guilty party.

In the course of doing business, sometimes sensitive data must be handed over to supposedly trusted third parties. Another enterprise may outsource its data processing, so data must be given to various other companies. There always remains a risk of data getting leaked from the agent. Leakage detection is handled by watermarking, e.g., a unique code is embedded in each distributed copy. If that copy is later discovered in the hands of an unauthorized party, the leaker can be identified. But again, it requires code modification. Watermarks can sometimes be destroyed if the data recipient is malicious.

1.2 OBJECTIVE OF THIS PROJECT:

The goal of Data leakage detection is to detect when the distributor's sensitive data has been leaked by agents, and if possible to identify the agent that leaked the data. Perturbation is a very useful technique where the data is modified and made "less sensitive" before being handed to agents. we develop unobtrusive techniques for detecting leakage of a set of objects or records. In this section we develop a model for assessing the "guilt" of agents.

We also present algorithms for distributing objects to agents, in a way that improves our chances of identifying a leaker. Finally, we also consider the option of adding "fake" objects to the distributed set. Such objects do not correspond to real entities but appear realistic to the agents. In a sense, the fake objects acts as a type of watermark for the entire set, without modifying any individual members. If it turns out an agent was given one or more fake objects that were leaked, then the distributor can be more confident that agent was guilty.

1.3 THE TYPE OF EMPLOYEES THAT MAY LEAK DATA

- The security illiterate
  - Majority of employees with little or no knowledge of security
  - Corporate risk because of accidental breaches
- The gadget nerds
  - Introduce a variety of devices to their work PCs
  - Download software

- The unlawful residents
    - Use the company IT resources in ways they shouldn't
    - i.e., by storing music, movies, or playing games
- The malicious/disgruntled employees
    - Typically, minority of employees
    - Gain access to areas of the IT system to which they shouldn’t
    - Send corporate data (e.g., customer lists, R&D, etc.) to third parties.

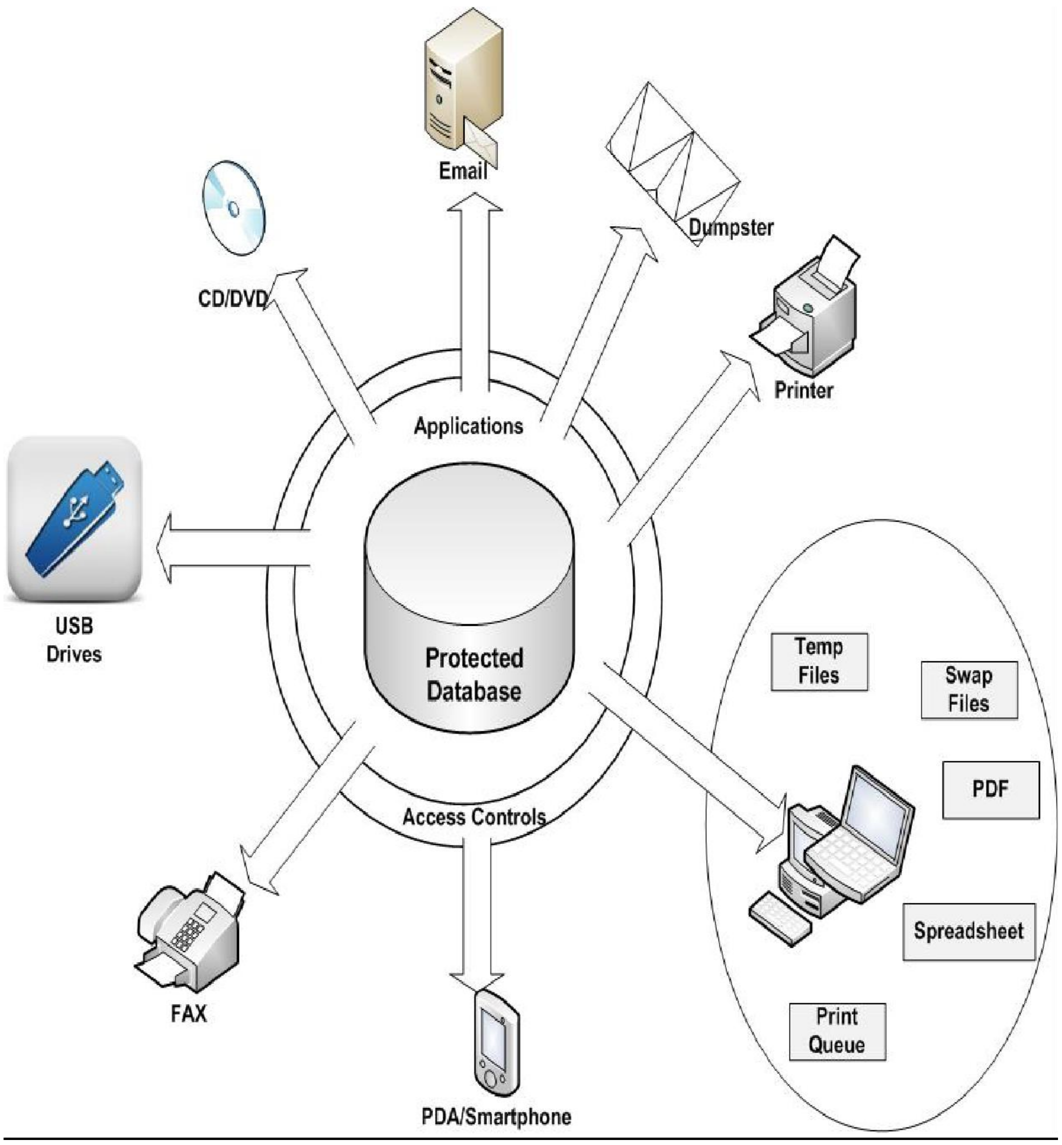


Fig 1.3 An Example of Data Leakage

CHAPTER 2

LITERATURE REVIEW

Literature survey is the most important step in software development process. Before developing the tool, it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, ten next steps are to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system, the above considerations are taken into account for developing the proposed system.

The issue of data security is basically stored in the cloud data store, which is a distributed storage system, and the traditional method uses RSA to validate erasure-encoded data to support duplicate equivalence vectors, erasure correction code for file distribution. It is encryption based and focuses on a scheme where ensures the confidentiality of customer data through the localization of data errors and the security and consolidation of storage.

The Uses browser historical data to determine where the attack occurred. Data is collected from browser history and stored in a database as evidence. Proposed methods and forensic tools help analyses data from law enforcement agencies.

A strong watermarking method can be very useful in some cases, but again requires minor modifications to the original data. Also, the watermark can be destroyed if the recipient of the data is malicious. Distributors may add fake objects to the data being distributed to increase the effectiveness of detecting guilty agents.

Data breaches are a huge problem facing the industry and various organizations. They discussed that distributors create and add fake objects to the data they distribute to agents. Distributors may add fake objects to the data being distributed to improve the effectiveness of detecting guilty agents. Leak detection is handled by algorithms and the integrity of users of these systems is at stake.

CHAPTER 3

SYSTEM ANALYSIS

System analysis is a problem-solving technique that decomposes a system into component pieces of purpose of studying how well those component parts work and interact to accomplish their purpose the following chapter provides the detail description of the existing system. It also provides an overview of the proposed system and feasibility of the Image resolution.

3.1 EXISTING SYSTEM:

Traditionally, leakage detection is handled by watermarking, e.g., a unique code is embedded in each distributed copy. If that copy is later discovered in the hands of an unauthorized party, the leaker can be identified. Watermarks can be very useful in some cases, but again, involve some modification of the original data. Furthermore, watermarks can sometimes be destroyed if the data recipient is malicious. E.g. A hospital may give patient records to researchers who will devise new treatments. Similarly, a company may have partnerships with other companies that require sharing customer data. Another enterprise may outsource its data processing, so data must be given to various other companies.

3.1.1 Fake Objects Method

Each In some applications, fake objects can cause less problems than real objects. For example, suppose the distributed data object is a medical record and the agent is a hospital. In this case, even a small change in the actual patient record may be undesirable. However, adding some fake medical records may be acceptable as there are no patients matching that record and therefore no one will treat based on the fake records. In this case, Company A sells the mailing list to Company B at for a one-time use (eg sending an ad).

3.1.2 Adversary Model

This model catches all kinds of threats to data integrity, this cloud data is not specified on the cloud client side, but in the cloud, service provides a domain address.

3.1.3 Drawbacks using existing system:

3.1.3.1 Insider attack: The cloud service provider is not trusted.

3.1.3.2 External Attacks: These attacks originate from third-party and non-domain-controlled cloud service providers.

3.2 PROPOSED SYSTEM:

The aim of this is to detect when the distributor's sensitive data have been leaked by agents, and if possible to identify the agent that leaked the data. The proposed method proposes the addition of a "realistic but false record" that increases the probability of data distribution strategy and leak detection and the encryption algorithm improves data security. It can also keep your data safe and detect leaks during transmission. To address data breaches, implementations of various data dissemination strategies have been submitted that may increase the likelihood that a data breach will be detected by a distributor.

3.2.1 Advantages over the existing system:

3.2.1.1 Client & Distributor Software (Data Detection System) installed at the server along with its database.

3.2.1.2 Database backup

3.2.1.3 Agent machine installed with the software and connected to the database at the server.

3.2.2 Architecture Diagram:

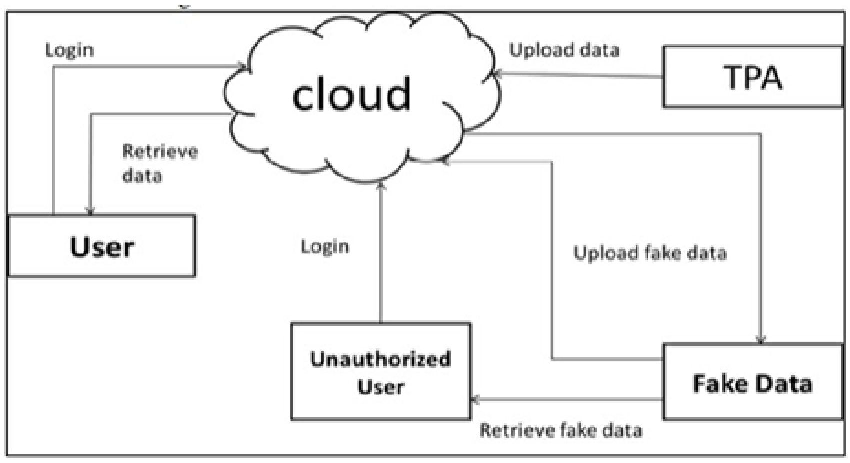


Fig 3.2.2 Proposed Architecture

Data leak detection, the proposed system shown in Figure 3.2.2 is to detect when the confidential data of the distributor is lost by the agent, and possibly identify the agent with the data leak.

Perturbation is a very useful technique to make data modified and "less sensitive" before being passed to agent. To increase the security of user data, the model has an encrypted algorithm that ensures secure data transfer between the user and the TPA. It includes the cloud as well as algorithms that distribute objects between agents in a way that improves the likelihood of detecting leaks. When adding misinformation objects to a distributed set, misinformation is provided if the data is provided to another third party.

3.2.3 Proposed Methodology:

SHA Algorithm: Secure SHA algorithm provides a hash function when downloading data by referring to a third-party agent and provides to the real cloud. This security feature has been added to improve the security of sensitive data transmission over the Internet. The SHA algorithm is used because it uses message digests to ensure data integrity and confidentiality. This approach uses Secure Hash Algorithm 2 (SHA2), a subset of the Keccak cryptography primitive family, to ensure data security. The SHA2 structure uses an internal block size of 1024, works with And, Xor, Rot, Add(mod 264), OR, Shr, and has 256-128 security bits.

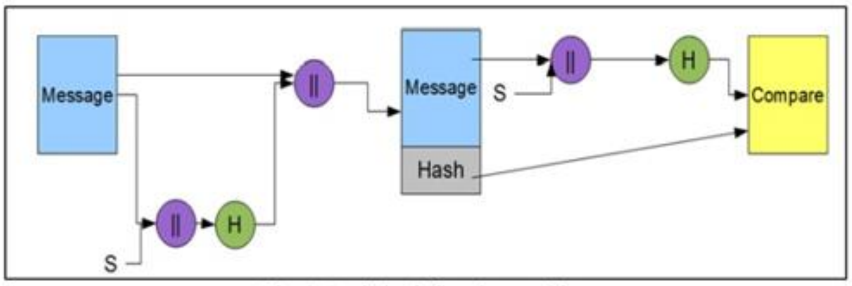


Fig 3.2.3 Proposed Methodology

Basic Operations: Boolean operations AND, XOR and OR. Bitwise complement, denoted by ̄. Integer addition modulo 232, denoted by A + B.Each of them operates on 32-bit words. For the last operation, binary words are interpreted as integers written in base 2. ShR(A, n) denotes the right shift of n bits of the binary word A.A||B denotes the concatenation of the binary words A and B.

CHAPTER 4

SYSTEM SPECIFICATIONS AND IMPLEMENTATION

A software requirements specification (SRS) is a description of a software system to be developed. It lays out functional and non-functional requirements, and may include a set of use cases that describe user interactions that the software must provide. Software requirements specification permits a rigorous assessment of requirements before design can begin and reduces later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules.

4.1 FUNCTIONAL REQUIREMENTS:

4.1.1 Login

The system should maintain a database to store all the login credentials.

4.1.2 File encryption

SHA-256 is that it doesn't have any known vulnerabilities that make it insecure and it has not been “broken” unlike some other popular hashing algorithms.

4.1.3 File sharing

Secure file sharing is the act of sharing files so that they are protected from data leak threats and only accessed by user granted access key by the sender.

4.1.4 Administrative system

- Information management: The administrator should be able to add, update and delete user data related activities.
- Login management: The administrator should be able to view and maintain login credentials.

4.2 NON-FUNCTIONAL REQUIREMENTS

4.2.1 User Interface

- The system shall maintain an easy to use interface across all functionality and for all users
- The client's user interface should be compatible with all commonly used browsers, such as Internet explorer, Firefox, Google chrome and Safari and also on all mobile devices.

4.2.2 Scalability

- The system shall be able to scale based on the number of users accounts using the system.

4.2.3 Security

- The administrative system should be protected from unauthorized access. Firebase provides the authentication which prevents the unauthorized access.
- The database should protect from attacks and unauthorized access. Firebase Database rules are strong enough to protect against attacks.
- The interface should be protected from attacks.The backend Code is in the minified format. So, Interface protects database from attacks.
- All passwords should be stored as a secure hash of the administrator password.

4.2.4 Third party interactions

- The system should be able to interact with the MySQL, to download and upload data. All browsers can able to interact with the MySQL.
- The system should be able to interact with the Google search server,

which is used for the customized search on the admissions website. When deployed, systems can be submitted to the Google server for crawling to be appearing in Google search.

4.2.5 Portability

- The system should run on all operating Systems. Since this is a multi- platform application, it will run on all Operating System.
- The system should run on a variety of hardware. The application supports all kinds of hardware including mobile devices.

4.2.6 Maintainability

- The system should be easy to maintain. API's we are using in this app can be maintained easily.
- There should be a clear separation between the interface and the business logic code. Business logic code is abstracted from the user.

4.2.7 Exception handling

Exceptions should be reported effectively to the user if they occur. Most likely, exception will occur on data upload and retrieval. If any exception occurs, User will be notified about the exception.

4.2.8 Ethics

The system shall not store or process any information about its users. The data we are collecting from the user won't be used for unethical purpose, data will be kept confidential.

4.3 HARDWARE REQUIREMENTS

- Processor : Dual core processor
- RAM: 2 GB
- Hard Disk : 500 GB
- Monitor : 16’ color Monitor
- Key board : Standard 110 keys
- Pointing Device : Mouse

4.4 SOFTWARE REQUIREMENTS

- Programming Language: PHP
- Operating System: Windows 10
- Front End : HTML, CSS, JavaScript , bootstrap5.
- Back End : MySQL
- Web Browser : Mozilla Firefox, Google Chrome, Opera

CHAPTER 5

SOFTWARE DESCRIPTION

A software requirements specification (SRS) is a description of a software system to be developed. It lays out functional and non-functional requirements, and may include a set of use cases that describe user interactions that the software must provide. Software requirements specification establishes the basis for an agreement between users and banking bot on what the software product is to do as well as what it is not expected to do. Software requirements specification permits a rigorous assessment of requirements before design can begin and reduces later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules.

5.1 FRONT END

The front end is designed using Bootstrap framework. Bootstrap is the platform made by developers for the Responsive from-End Design. Bootstrap is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites. Bootstrap is completely free to download and use.

5.1.1 HTML

HTML stands for Hyper Text Programming Language which was created by Berners-Lee. HTML first standard specification was published in 1995. It is one of the globally accepted programming languages for the development of interactive websites and webpages. Nowadays, HTML is used with other languages like Javascript, CSS which gives more look and feel to a website by providing different font color, font size, and alignment to the whole content.

5.1.1.1 Features

- Simple: It is very simple and easy to use, compare to other scripting language it is very simple and easy, this is widely used all over the world.

- Interpreted: It is an interpreted language, i.e. there is no need for compilation.
- Faster: It is faster than other scripting language e.g. asp, jsp and php.
- Open Source: Open source means you no need to pay for use html, you can free download and use.
- Platform Independent: HTML code will be run on every platform, Linux, Unix, Mac OS X, Windows.

5.1.1.2 Advantages

- Almost all the browsers around the globe are supported by HTML. So there is no need to worry about the website written in HTML for the browser support as the website would easily show up in all the browsers if the program keeps in mind to optimize the website for the different browsers.
- HTML is very easy to edit as there is no need to have a special interface or platform to edit it. It is written in simple Notepad and hence can be simply edited in any text editor like Notepad, Notepad++, etc.
- HTML can be easily integrated with multiple languages and does not create any issues in it. For example in Javascript, Php, node.js, CSS and many more, we write the code of these languages between the HTML and it mixes with them very easily.
- HTML is lightweight language. It has a high signal to noise ratio as compared to other forms of communication. It is also faster to download HTML code, which means it is highly compressive also.
- For the programmer to be either frontend or backend developer, one must have knowledge of HTML as it is the basic language and all the other languages integrate with it while coding like JavaScript, JSP, Php, etc. Similarly, XML syntax is just like HTML and XML which is used widely these days for data storage. If one has good knowledge of HTML, it is easy working with XML too for him.

- HTML is a user-friendly programming language. One does not need to have any prior knowledge of any language. Understanding of simple English is sufficient to work with it.

5.1.2 Cloud Computing

- Cloud computing is on-demand access, via the internet, to computing resources—applications, servers (physical servers and virtual servers), data storage, development tools, networking capabilities, and more—hosted at a remote data center managed by a cloud services provider (or CSP).

5.1.2.1 Features

- Self-service On-Demand.
- Resources Pooling.
- Easy Maintenance.
- Economical.
- Rapid Elasticity and Scalability.

5.1.2.2 Advantages

- More flexibility and reliability
- Increased performance and efficiency
- Helps to lower IT costs
- Security.
- Flexibility.
- Mobility.

5.1.3 PHP

- PHP stands for Hypertext Preprocessor. It is a server scripting language, and a powerful tool for making dynamic and interactive Web pages.
- PHP is a widely-used, free, and efficient alternative to competitors such as Microsoft's ASP.

5.1.3.1 Advantages

- It is an open-source programming and scripting language.
- It is simple to learn and execute for beginners and pro developers.

5.2 BACK END

The back end is designed using MYSQL, whose primary function is to store data securely and retrieve it later, as requested by other software applications.

5.2.1 MYSQL

- MYSQL encompasses a wide variety of different database technologies that were developed in response to the demands presented in building Modernapplications:Developers are working with applications that create massive volumes of new, rapidly changing data types structured, semi-structured, unstructured and polymorphic data.
- Organizations are now turning to scale-out architectures using open software technologies, commodity servers and cloud computing instead of large monolithic servers and storage infrastructure.
- Relational databases were not designed to cope with the scale and agility challenges that face modern applications, nor were they built to take advantage of the commodity storage and processing power available today.

5.2.2 Features of MYSQL

MultiModel - Where relational databases require data to be put into tables and columns to be accessed and analyzed, the various data model capabilities of MYSQL databases make them extremely flexible when it comes to handling data. They can ingest structured, semi-structured, and unstructured data with equal ease, whereas relational databases are extremely rigid, handling primarily structured data. Different data models handle specific application requirements. Developers and architects choose a MYSQL database to more easily handle different agile application development requirements. Popular data models include graph, document, wide-column, and key-value.

5.2.2.1 Easily Scalable

It's not that relational databases can't scale, it's that they can't scale EASILY or CHEAPLY, and that's because they're built with a traditional master-slave architecture, which means scaling UP via bigger and bigger hardware servers as opposed to OUT or worse via sharding. Instead, look for a MYSQL database with a masterless, peer-to-peer architecture with all nodes being the same. This allows easy scaling to adapt to the data volume and complexity of cloud applications. This scalabilty also improves performance, allowing for continuous availability and very high read/write speeds.

5.2.2.2 Flexible

Where relational databases require data to be put into tables and columns to be accesses and analyzed, the multi-model capabilities of MYSQL databases make them extremely flexible when it comes to handling data. They can easily process structured, semi-structured, and unstructured data, while relational databases, as stated previously, are designed to handle primarily structured data.

5.2.2.3 Distributed

Relational databases, in contrast, use a centralized application that is location-dependent (e.g. single location), especially for write operations. A key advantage of using a distributed database with a masterless architecture is that you can maintain continuous availability because data is distributed with multiple copies where it needs to be. Updatable views.

5.2.2.4 Zero Downtime

The final but certainly no less important key feature to seek in a MYSQL database is zero downtime. This is made possible by a masterless architecture, which allows for multiple copies of data to be maintained across different nodes.

5.2.3 Advantages of MySQL

- Data Security
- On-Demand Scalability
- High Performance
- Round-the-Clock Uptime
- Comprehensive Transactional Support
- Complete Workflow Control

CHAPTER 6

PROJECT DESCRIPTION

In this project we are finding out the data is leaked or not. The agent will give the information to broadcast the data via a server to other agents. We will check whether the authorized user leaked the data to another agent.

Admin is the main authority who can do addition, deletion, and modification if required.

In this project authorized agent give the request for data request may be explicit or sample, according to request agent get the data. If agent leaked the data to another agent, then in our side we are checking whether the data matches our data. After that we will find out the agent who leaked the data.

The database connectivity is planned using the “MYSQL Connection” methodology. The standards of security and data protective mechanism have been given a big choice for proper usage.

6.1 OVERVIEW OF THE PROJECT

This project is aimed to detect when the distributor’s sensitive data have been leaked by agents, and if possible to identify the agent that leaked the data. Perturbation is a very useful technique where the data is modified and made “less sensitive” before being handed to agents. we develop unobtrusive techniques for detecting leakage of a set of objects or records. In this section we develop a model for assessing the “guilt” of agents.

6.2 MODULE DESCRIPTION

The implementation details of each the modules can be explained as follows:

6.2.1 sign-in and sign-up

The sign-in and sign-up module makes the way for adding the user's account. It ask the user to type the email id and password or mobile number login, user name along with the user details. All the process are been secured.

6.2.2 Shared Files

A data file is a computer file which stores data to be used by a system, including sensitive data. A data file usually does not contain instructions or code to be executed. It Stores in encrypted type on cloud.

6.2.3 Key Request

A key that is used for granting access the data via the SHA2 protocol (as opposed to a key, which does not grant access to anything but serves to authenticate a host). Both authorized keys and identity keys are user keys. A user key is the equivalent of an access token.

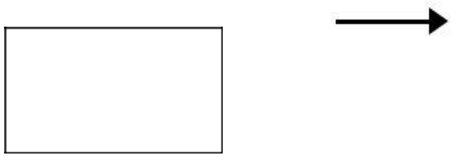
6.2.4 Data Leak analysis

It is identifies an organization's data leaks – the accidental public exposure of sensitive data due to software misconfigurations and poor network security. Data leaks quickly become data breaches when cybercriminals identify and exploit this exposed data.

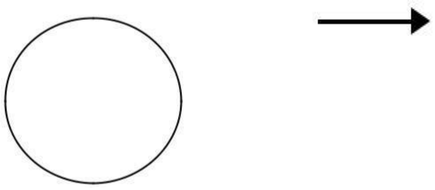
6.3 DATA FLOW DIAGRAM

Data flow diagram is used to describe how the information is processed and stored and identifies how the information flows through the processes. Data flow diagram illustrates how the data is processed by a system in terms of inputs and outputs. The data flow diagram also depicts the flow of the process and it has various levels. The initial level is context level which describes the entire system functionality and the next level describes each and every sub module in the main system as a separate process or describes all the process involved in the system separately.

Data flow diagram are made up of number of symbols,

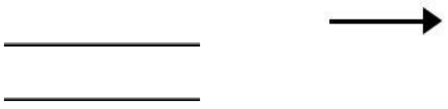


Square representing external entities, which are sources or destinations of data.



Circle representing processes, which take data as input, do something to it and output it.

Arrows representing the data flows, which can either, be electronic data or physical items.



Parallel lines representing data stores, including electronic stores such as databases or XML files and physical stores

7.2.2 DFD Level 0:

The output of the system is reports. The users store and retrieve from the database.

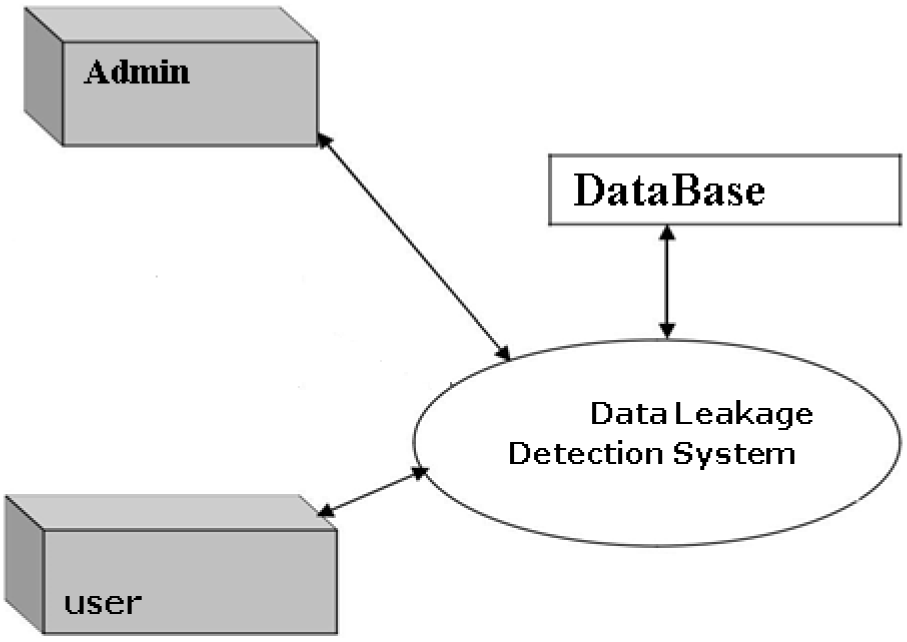


Fig.6.3.1 DFD Level 0

6 .3.2. DFD Level 1:

In the DFD level 1, all the transactions are been explained clearly. The user requests for an action, it performs the user's desired action and updates the changes in the server.

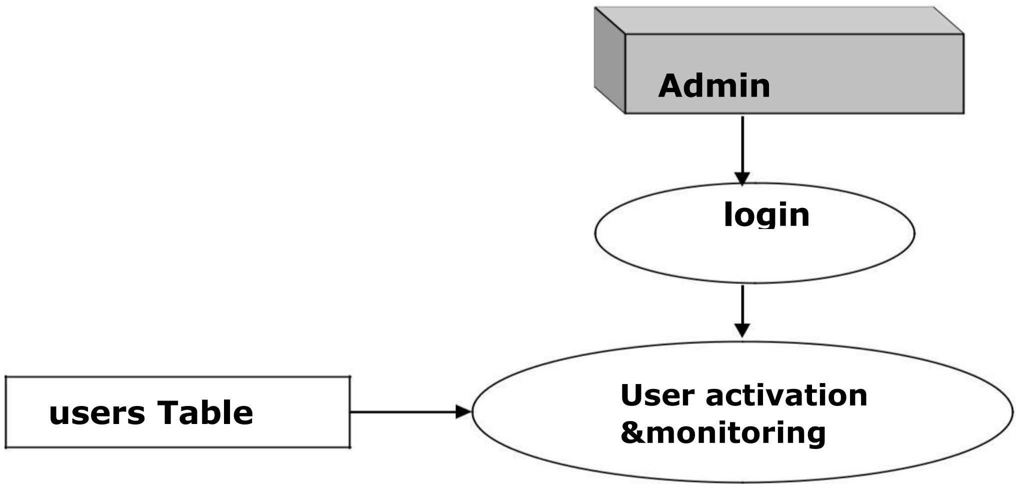


Fig.6.3.2 DFD Level 1 for admin

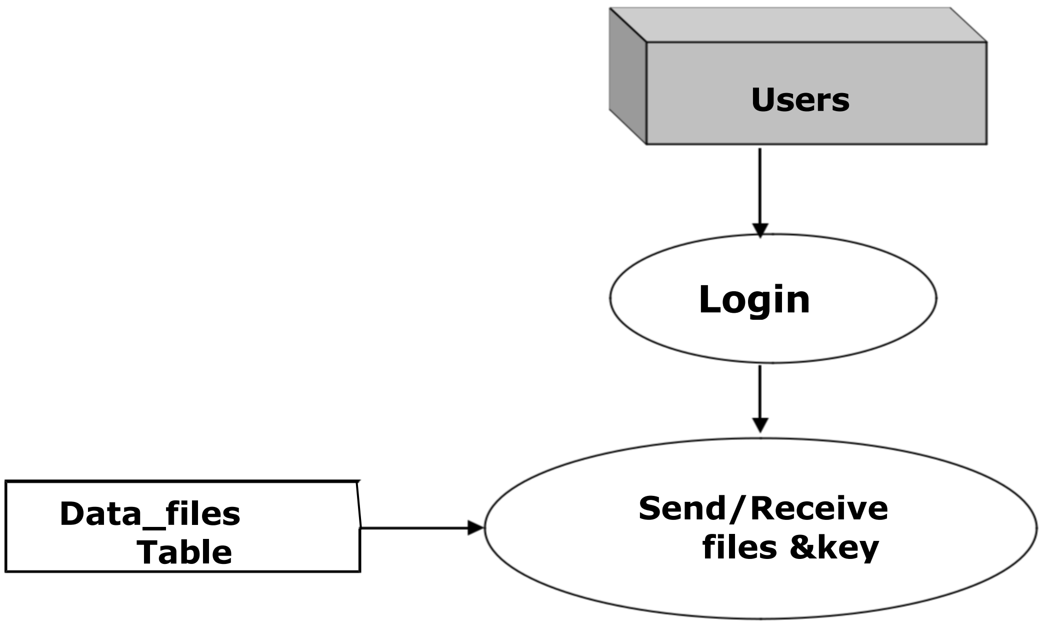


Fig.6.3.3 DFD Level 1 for users

6 .3.3. DFD Level 2:

A DFD Level 2 uses to represent the functions performed by the system.

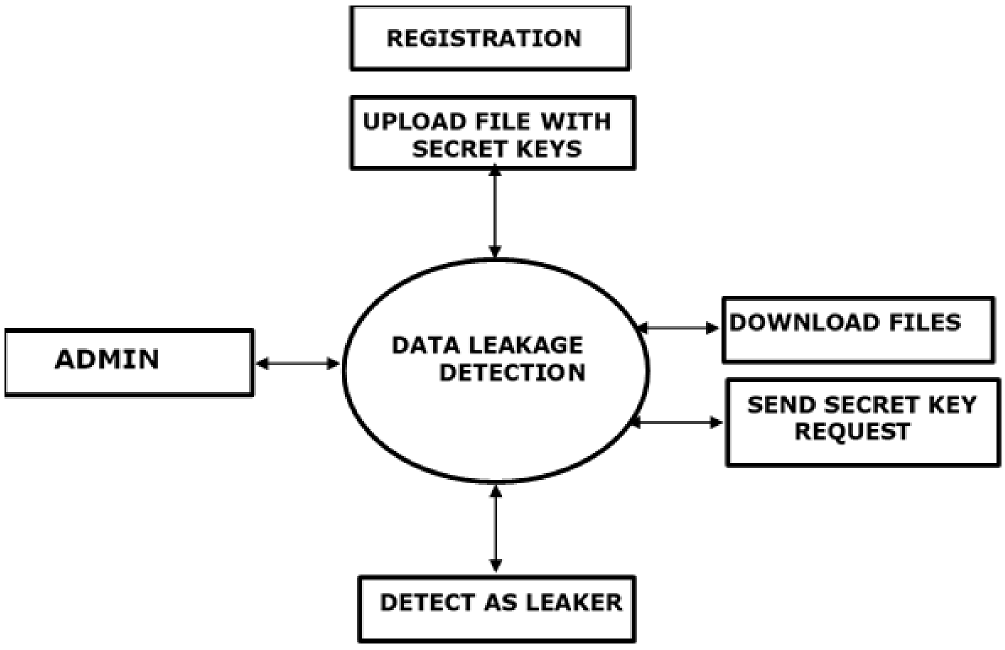


Fig.6.3.4 DFD Level 2

6.4 ER DIAGRAM

An entity–relationship model (ER model) describes inter-related things of interest in a specific domain of knowledge. An ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types. In software engineering an ER model is commonly formed to represent things that a business needs to remember in order to perform business processes. Consequently, the ER model becomes an abstract data model that defines a data or information structure that can be implemented in a database, typically a relational database.

Entity–relationship modeling was developed for database design by Peter Chen and published in a 1976 paper. Some ER modelers show super and subtype entities connected by generalization-specialization relationships, and an ER model can be used also in the specification of domain-specific ontology.

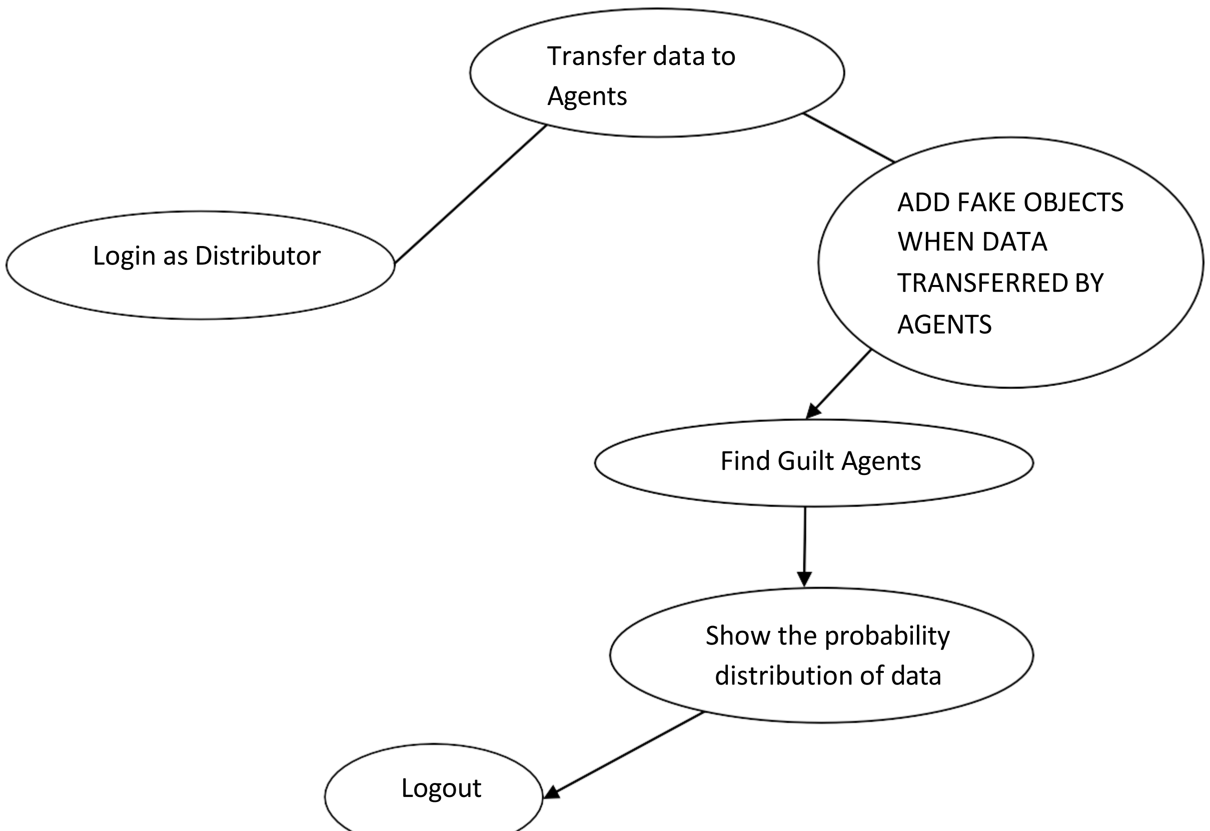

Transfer data to Agents
Login as Distributor
ADD FAKE OBJECTS WHEN DATA TRANSFERRED BY AGENTS
Find Guilt Agents
Show the probability distribution of data
Logout

Fig.6.4 ER Diagram for Data Leakage Detection

6.5 DATABASE DESIGN:

Database design is the process of producing a detailed data model of database. This data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

The term database design can be used to describe many different parts of the design of an overall database system. Principally, and most correctly, it can be thought of as the logical design of the base data structures used to store the data. In the relational model these are the tables and views. In an object database the entities and relationships map directly to object classes and named relationships. However, the term database design could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application within the database management system (DBMS).

Id,Username,Email,Password,User_type,Gender,Mobile,Admin_active, Blocked, Profile

S.NO	FIELD NAME	DATATYPE
1	Id	Int
2	Username	Varchar
3	Email	Varchar
4	Password	Varchar
5	User_type	Varchar
6	Gender	Enum

7	Mobile	Varchar
8	Admin_active	Enum
9	Blocked	Enum
10	Profile	Varchar

TABLE 6.5 User’s Detail

6.6 INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. It also provides record viewing facilities.

6.7 OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

CHAPTER 7

SYSTEM TESTING

System Testing is a level of the software testing where complete and integrated software is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements. By definition of ISTQB system testing is the process of testing an integrated system to verify that it meets specified.

7.1 TESTING METHODS

Software Testing Type is a classification of different testing activities into categories, each having, a defined test objective, test strategy, and test deliverables. The goal of having a testing type is to validate the Application under Test for the defined Test Objective. For instance, the goal of Accessibility testing is to validate the AUT to be accessible by disabled people. So, if your Software solution must be disabled friendly, you check it against Accessibility Test Cases.

7.2 TYPES OF TESTING

7.2.1 Unit Testing

In computer programming, unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use.

In this system, every units of code is been tested and the correctness of every module is been ensured.

Login page, Signup page, Key and the File sharing are the units of this project and all the modules in each units are tested and debugged carefully.

Sl.NO	Input	Expected Output	Actual Output	Status
01	Welcome Page	A page with Admin login and User login	A page with Admin login and User login	Pass
02	AdminLogin	Login page with user name and password arrives	Login page with user name and password arrives	Pass
03	Admin Function	The 4 sub models of Admin login	The 4 sub models of Admin login	Pass
04	Sign up new user	A sign up page where the user has to enter personal details	A sign up page where the user has to enter personal details	Pass
05	Distributor Server	Before sending data from the distributor this server has to be turned on	Before sending data from the distributor this server has to be turned on	Pass
06	Distribute data via the distributor to user	A page with a set of hosts arrive and can send data to various Uses	A page with a set of hosts arrive and can send data to various Uses	Pass
07	Return to Welcome Page	A page with Admin login and User login	A page with Admin login and User login	Pass
08	Login as user	A page where the Users enters the Users details	A page where the Users enters the Users details	Pass

Table 7.2.1 Testcases

7.2.2 Integration Testing

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

In this system, the units are been tested as a whole and the testing was successful.

Login page, Signup page, Key and the File sharing are the core modules of this project. Each modules are combined together and tested to remove all the bugs.

7.2.3 Stress Testing

Stress testing a Non-Functional testing technique that is performed as part of performance testing. During stress testing, the system is monitored after subjecting the system to overload to ensure that the system can sustain the stress.

Reasons can include:

- to determine breaking points or safe usage limits
- to confirm mathematical model is accurate enough in predicting breaking points or safe usage limits
- to confirm intended specifications are being met
- to determine modes of failure (how exactly a system fails)
- to test stable operation of a part or system outside standard usage

The recovery of the system from such phase (after stress) is very critical as it is highly likely to happen in production environment.

7.2.4 White Box Testing

White Box Testing is the testing of a software solution's internal coding and infrastructure. It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability. White box testing is also known as Clear Box testing, Open Box testing, Structural testing, Transparent Box testing, Code-Based testing, and Glass Box are testing. It is one of two parts of the "box testing" approach of software testing. Its counter-part, black box testing, involves testing from an external or end-user type perspective. On the other hand, White box testing is based on the inner workings of an application and revolves around internal testing.

The term "white box" was used because of the see-through box concept. The clear box or white box name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "black box testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested.

7.2.5 Black Box Testing

Black box testing is a software testing techniques in which functionality of the software under test (SUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications. We have tested this web application with external users and we got the report from them. The Report says, the application works fine without any bug.

7.2.5.1 Methods of Black Box Testing

There are many types of Black Box Testing but following are the prominent ones -

- Functional testing - This black box testing type is related to functional requirements of a system; it is done by software testers.
- Non-functional testing - This type of black box testing is not related to testing of a specific functionality, but non-functional requirements such as performance, scalability, usability.
- Regression testing - Regression testing is done after code fixes, upgrades or any other system maintenance to check the new code has not affected the existing code.

7.3 TESTING STRATEGY

Test Strategy is also known as test approach defines how testing would be carried out. Test approach has two techniques:

- Proactive - An approach in which the test design process is initiated as early as possible in order to find and fix the defects before the build is created.
- Reactive - An approach in which the testing is not started until after design and coding are completed.

Test strategy calls for implementing two entirely different methodologies for testing this project. The chat bot includes a fair amount of manual UI-based testing.

CHAPTER 8

SYSTEM IMPLEMENTATION

The System Implementation is the process that actually yields the lowest-level system elements in the system hierarchy (system breakdown structure). The system elements are made, bought, or reused. Production involves the hardware fabrication processes of forming, removing, joining, and finishing; or the software realization processes of coding and testing; or the operational procedures development processes for operators' roles. If implementation involves a production process, a manufacturing system which uses the established technical and management processes may be required.

The purpose of the implementation process is to design and create (or fabricate) a system element conforming to that element's design properties and/or requirements. The element is constructed employing appropriate technologies and industry practices. This process bridges the system definition processes and the integration process.

System Implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and in giving confidence on the new system for the user that it will work efficiently and effectively. The existing system was long time process.

The proposed system was developed using Visual Studio .php. The existing system caused more paperwork and consuming time but the system developed now has a very good user-friendly tool, which has a menu-based interface, graphical interface for the end user. After coding and testing, the project is to be hosting on the website is necessary. The file is to be created and loaded in the website. Hosting the developed code in system in the form of executable file is implementation.

8.1 IMPLEMENTATION METHODS

8.1.1 DATA ALLOCATION

The main focus of this paper is the data allocation problem: How can the distributor “intelligently” give data to agents in order to improve the chances of detecting a guilty agent? As illustrated in Fig. 1, there are four instances of this problem we address, depending on the type of data requests made by agents and whether “fake objects” are allowed.

8.1.2 OPTIMIZATION

The distributor’s data allocation to agents has one constraint and one objective. The distributor’s constraint is to satisfy agents’ requests, by providing them with the number of objects they request or with all available objects that satisfy their conditions. His objective is to be able to detect an agent who leaks any portion of his data. We consider the constraint as strict. The distributor may not deny serving an agent request as and may not provide agents with different perturbed versions of the same objects. We consider fake object distribution as the only possible constraint relaxation.

CHAPTER 9
CONCLUSION & FUTURE ENHANCEMENTS

9.1 CONCLUSION

In a perfect world, there would be no need to hand over sensitive data to agents that may unknowingly or maliciously leak it. Thus the project can be extended to allow for the dynamic creation of forged records at the request of the agent. This shows that the method can effectively detect data leaks in cloud platforms. When comparing to related operations, the other methods focus more on data leaks occurring at the application level. Going forward, we would like to expand our investigation by discovering potential data breaches in other cloud components such as block storage, API requests, and telemetry.

9.2 FUTURE ENHANCEMENT

There is scope for future development of this project. The world of computer fields is not static; it is always subject to be dynamic. The technology which is famous today becomes outdated the very next day. To keep abstract of technical improvements, the system may be further refined. So, it is not concluded. Yet it will improve with further enhancements. It can be improved to include new features. Our application is no different from this. The future enhancements that can be made to Data Leakage Detection are:

- Providing support for other file formats.
- Creation of a web based UI for execution of the application.
- Improving the detection process based on user requirements.
- Provision of quality or accuracy variance parameter for the user to set.

CHAPTER 10

APPENDIX

10.1 SOURCE CODE

Home.php

```
<?php require_once("../server/connect.php"); ?>

<?php include_once("../session.php"); ?>

<?php include_once("../sanitize.php"); ?>

<!DOCTYPE html>

<html lang="en">

<head>

 <meta charset="UTF-8">

 <meta   name="viewport"   content="width=device-width,
initial-scale=1.0">

 <title>Team 7_Data leakage and security</title>

 <?php include_once("bootstrap.php"); ?>

 <style>

 ded, to get the result that you can see in the preview
selection



body{
        margin-top:20px;

        background: #f6f9fc;

    }

    .account-block {

        padding: 0;

        background-image:
url(https://bootdey.com/img/Content/bg1.jpg);

        background-repeat: no-repeat;

        background-size: cover;

        height: 100%;

        position: relative;

    }

    .account-block .overlay {

        -webkit-box-flex: 1;

        -ms-flex: 1;

        flex: 1;

        position: absolute;

        top: 0;

        bottom: 0;

        left: 0;

        right: 0;
        background-color: rgba(0, 0, 0, 0.4);
    }
    .account-block .account-testimonial {
        text-align: center;
        color: #fff;
        position: absolute;
        margin: 0 auto;
        padding: 0 1.75rem;
        bottom: 3rem;
        left: 0;
        right: 0;
    }

    .text-theme {
        color: #5369f8 !important;
    }

    .btn-theme {
        background-color: #5369f8;
        border-color: #5369f8;
```

```
        color: #fff;

    }

    </style>

    </head>

    <body class="login_background">

    <?php include_once("menubar.php"); ?>



                              <?php
include_once("success_message.php"); ?>


    <?php

    if(isset($_POST['login'])){

         $email=sanitize($_POST['email']);

         $password=sanitize($_POST['password']);

         $result=mysqli_query($conn, "SELECT * FROM users
WHERE email='$email' AND password='$password'");

         if(mysqli_num_rows($result)>0){

              $rows=mysqli_fetch_array($result);

              if($rows['admin_active'] == "1"){

                   $_SESSION['user_id']=$rows['id'];
```

```
                    $_SESSION['username']=$rows['username'];
                    $_SESSION['profile']=$rows['profile'];

$_SESSION['user_type']=$rows['user_type'];
                    $_SESSION['is_login']="loginned";
                    $url="dashboard.php";
                    $_SESSION['success_message']="You    are
successfully loginned.";
                    header("Location:$url");
                    exit();
               }
               else{
                    echo    "<div    class='alert    alert-
warning'>Failed  to  login:  Your  account  not  approved  by
admin.</div>";
               }
          }
          else{
               ?>
     <div class="alert alert-danger">
     Invalid email id or password
     </div>
     <?php
          }
      }
       ?>
       <div id="main-wrapper" class="container">
          <div class="row justify-content-center">
              <div class="col-xl-10">
                  <div class="card border-0">
                      <div class="card-body p-0">
                          <div class="row no-gutters">
                              <div class="col-lg-6">
                                  <div class="p-5">
                                      <div class="mb-5">
                                          <h3 class="h4 font-
weight-bold text-theme">Login</h3>
                                      </div>

                                      <h6    class="h5    mb-
0">Welcome back!</h6>
                                      <p    class="text-muted
```

```
mt-2 mb-5">Enter your email address and password to access
your account.</p>
                                  <form
action="<?=$_SERVER['PHP_SELF']?>" method="POST">
                              <div class="form-group">
                                              <label
for="email">Email address</label>
                                              <input
type="email" class="form-control" name="email">
                                          </div>
                                          <div   class="form-
group mb-5">
                                              <label
for="password">Password</label>
                                              <input
type="password" class="form-control" name="password">
                                          </div>


<input type="submit" name="login" value="Login" class="btn
btn-theme">                    &nbsp; &nbsp; &nbsp; &nbsp
<a    href="#l"    class="forgot-link    float-right    text-
primary">Forgot password?</a>
                                     </form>
                                 </div>
```

```
                                        </div>


                                        <div class="col-lg-6 d-none d-
lg-inline-block">
                                            <div   class="account-block
rounded-right">
                                                <div     class="overlay
rounded-right"></div>
                                                <div    class="account-
testimonial">
                                                    <h4    class="text-
white mb-4">Data Leakage and Security</h4>  <p class="lead
text-white">Data leakage is the big challenge in front of
the industries & different institutes. Though there are
number of systems designed for the data security by using
different encryption algorithms, there is a big issue of the
integrity of the users of those systems. It is very hard for
any system administrator to trace out the data leaker among
the system users. It creates a lot many ethical issues in
the working environment of the office.</p>
                                                    <p>Admin/User
login page</p>
                                                </div>
                                            </div>
```

```
                              </div>
                          </div>
                      </div>
                      <!-- end card-body -->
                  </div>
                  <!-- end card -->
                  <p class="text-muted text-center mt-3 mb-0">Don't have an account? <a href="register.php" class="text-primary ml-1">register</a></p>
                  <!-- end row -->
              </div>
              <!-- end col -->
          </div>
          <!-- Row -->
      </div>
      </body>
      </html>
```

Dashboard.php

```
<?php require_once("../server/connect.php"); ?>
<?php include_once("../session.php"); ?>
<?php require_once("hasAccessUser.php"); ?>
<?php include_once("library.php"); ?>
```

```
  <!DOCTYPE html>
  <html lang="en">
  <head>
      <meta charset="UTF-8">
      <meta name="viewport" content="width=device-
width, initial-scale=1.0">
      <title>T7_Data-leakage-detection-
system</title>
      <?php include_once("bootstrap.php"); ?>
  </head>
  <body class="dashboard_background">
      <?php include_once("menubar.php"); ?>
      <div class="container-fluid">
          <div class="row my-3">
              <div class="col-sm-3">
  <div class="list-group">
  <a href="user_profile.php" class="list-group-
item"><i class="bi bi-person-circle"></i>
Profile</a>
  <a href="send_files_to_users.php" class="list-
group-item"><i class="bi bi-send-fill"></i> Send
Files</a>
  <a href="list_of_key_requests.php" class="list-
group-item"><i class="bi bi-key-fill"></i> Key
Requests</a>
  <a href="list_of_files_send_by_me.php"
class="list-group-item"><i class="bi bi-send-check-
```

```
fill"></i> Files sent by me</a>
  <a href="list_of_files_send_by_other_users.php"
class="list-group-item"><i class="bi bi-
people"></i> Files sent by another</a>
  <?php
  if(isset($_SESSION['user_type'])){
      if($_SESSION['user_type']=='admin'){
  ?>
  <a href="users.php" class="list-group-item"><i
class="bi bi-info-circle-fill"></i> User
registeration request</a>
  <a href="leaker_user_list.php" class="list-group-
item"><i class="bi bi-paint-bucket"></i> Leaker</a>
  <?php
      }
      elseif($_SESSION['user_type']=='user'){
  ?>
  <?php
      }
  }
   ?>
  </div>
              </div>
              <!-- col -->
              <div class="col-sm-9">
                  <div class="row">
```

```
                      <div class="col-sm-12">
                          <?php
include_once('success_message.php');?>
                      </div>
                  </div>
                  <!-- ROW -->

                  <?php if($_SESSION['user_type']
== "admin"){ ?>
                  <div class="row" >
                      <div class="col-sm-8"></div>
                      <div class="col-sm-4
overflow-auto" style="height:400px;">
                          <h4 class="text-white bg-
primary p-2 rounded">Leaked Messages</h4>
                          <?php
                          $sql="SELECT * FROM
leaked_messages ORDER BY id DESC";
                          $result=mysqli_query($con
n, $sql);
                          ?>
                          <div class="list-group">
                          <?php
while($rows=mysqli_fetch_array($result)){?>
                          <div class="list-group-
item">
```

```
                              <?=getfiledetail($row
s['file_id'])?>
                                  <span
class='text-primary'>
                                       <small>
                                          <?=ucfirs
t(getusername($rows['user_id']))?>, tries to
download file
                                       </small>
                                  </span>
                                      <a
                      href="delete_leaked_message.p
hp?id=<?=$rows['id']?>"
                      class="btn btn-danger btn-sm
mb-2">Delete</a>
                          </div>
                          <?php } ?>
                          </div>
                      </div>
                  </div>
                  <?php } ?>

              </div>
              <!-- col -->
          </div>
          <!-- row -->
```

```
      </div>
      <!-- container -->
  </body>
  </html>
```

sharefiles.php

```
       <?php require_once("../server/connect.php");
?>
  <?php include_once("../session.php"); ?>
  <?php include_once("library.php");?>
  <!DOCTYPE html>
  <html lang="en">
  <head>
      <meta charset="UTF-8">
      <meta name="viewport" content="width=device-
width, initial-scale=1.0">
      <title>Document</title>
      <?php include_once("bootstrap.php"); ?>
  </head>
  <body class="dashboard_background">
      <?php include_once("menubar.php"); ?>
  <div class="container">
      <div class="row">
          <div class="col-sm-12">


  <table class='table table-bordered my-5 bg-
white'>
```

```
      <thead>
       <tr>
          <th>Sr No</th>
          <th>Subject</th>
          <th>File name</th>
          <th>file size</th>
          <th>User(To)</th>
       </tr>
      </thead>
  <tbody>
  <?php
      $self=$_SESSION['user_id'];
      $sql="SELECT * FROM data_files WHERE
sender_id='$self'";
      $result=mysqli_query($conn,$sql);
      if($result){
         if(mysqli_num_rows($result)>0){
              $n=0;
              while($rows=mysqli_fetch_array($resul
t)){
                  $n++;
                  $subject=$rows['subject'];
                  $filename=$rows['file_name'];
                  $filesize=$rows['file_size'];
                  $userid=$rows['receiver_id'];
                  $user=getusername($userid);
?>

<tr>
  <td><?=$n?></td>
  <td><?=$subject?></td>
  <td><?=$filename?></td>
  <td><?=$filesize?></td>
  <td><?=ucfirst($user)?></td>
</tr>

<?php } ?>

</tbody>
</table>

<?php }else{ ?>

<div class='alert alert-danger'>
  <strong>Failed:</strong>
  No users found.
</div>

 <?php    } } else{ ?>

<div class='alert alert-danger'>
```

```
    <strong>Failed:</strong>
  </div>

  <?php } ?>
          </div>
          <!-- col -->
      </div>
      <!-- row -->
  </div>
  <!-- container -->
  </body>
  </html>
```

Share Key.php

```
  <?php
  require_once("../session.php");
  require_once("../server/connect.php");
  $id=$_GET['id'];
  $asker=$_SESSION['user_id'];

  function getfiledetail($fileid){
  global $conn;
    $data="";
  $sql="SELECT * FROM data_files WHERE id='$fileid'
LIMIT 1";
  $result=mysqli_query($conn,$sql);
  $row=mysqli_fetch_array($result);
  return $row;
  }

  $file = getfiledetail($id);
  $secret_key = $file['secret_key'];
  $request_to_user = $file['sender_id'];
  $sql="INSERT INTO key_requests(request_by_user,
request_to_user, file, secret_key,
status)VALUES('$asker', '$request_to_user', '$id',
'$secret_key', 'pending')";
  mysqli_query($conn, $sql);
  echo "Request successfully sent.";
  ?>
```

deleteleakmessage.php

```
  <?php
  require_once("../server/connect.php");
  $id=$_GET['id'];
  $sql="DELETE FROM leaked_messages WHERE id='$id'
LIMIT 1";
  $result=mysqli_query($conn, $sql);
  header("Location:dashboard.php");
  exit();
   ?>
```

library.php

```
  <?php
  function get_attempt($id){
    global $conn;
    $user_id=$_SESSION['user_id'];
    $sql="SELECT * FROM attempts WHERE
file_id='$id' AND user_id='$user_id'";
    $result=mysqli_query($conn, $sql);
    $record = mysqli_fetch_array($result);
    return $record['attempt'];
  }
  function mark_attempt($id){
    global $conn;
    $user_id=$_SESSION['user_id'];
    $sql="SELECT * FROM attempts WHERE
file_id='$id' AND user_id='$user_id'";
    $result=mysqli_query($conn, $sql);
    $record = mysqli_fetch_array($result);
    $attempt = $record['attempt']-1;
    mysqli_query($conn, "UPDATE attempts SET
attempt='$attempt' WHERE file_id='$id' AND
user_id='$user_id'");
    if($attempt==0){
      mysqli_query($conn, "UPDATE users SET
blocked='1' WHERE id='$user_id' LIMIT 1");
    }
  }
  function check_attempts($id){
    global $conn;
    $user_id=$_SESSION['user_id'];
    $sql="SELECT * FROM attempts WHERE
file_id='$id' AND user_id='$user_id'";
    $result=mysqli_query($conn, $sql);
    $record = mysqli_num_rows($result);
    if($record > 0){
    }
    else{
    mysqli_query($conn, "INSERT INTO
attempts(user_id,
file_id,attempt)VALUES('$user_id', '$id','2')");
    }
  }

  function getusername($userid){
      global $conn;
    $sql="SELECT * FROM users WHERE id='$userid'
LIMIT 1";
    $result=mysqli_query($conn,$sql);
    $rows=mysqli_fetch_array($result);
    return $rows['username'];
  }
  function getuser($userid){
      global $conn;
    $sql="SELECT * FROM users WHERE id='$userid'
```

```
LIMIT 1";
    $result=mysqli_query($conn,$sql);
    $row=mysqli_fetch_array($result);
    return $row;
  }
  function getfiledetail($fileid){
  global $conn;
    $data="";
  $sql="SELECT * FROM data_files WHERE id='$fileid'
LIMIT 1";
  $result=mysqli_query($conn,$sql);
  if($result){
     if(mysqli_num_rows($result)>0){
              while($rows=mysqli_fetch_array($resul
t)){
                 $subject=$rows['subject'];
                 $filename=$rows['file_name'];
                 $f=$rows['file_size'];
                 $filesize=round($rows['file_size']
,2)."Mb";

                 if($filesize==0){
                  $filesize=round(($f*1024),3)."Kb"
;
                 }
                 $data.="
```

```
        <strong>Subject:</strong> $subject <br>
        <strong>File Name:</strong> $filename <br>
        <strong>File Size:</strong> $filesize
                 ";
              }
     }
     else{
      $data.="not found";
     }
}
else{
  $error=mysqli_error($conn);
    $data.="
<div class='alert alert-danger'>
  <strong>Failed:</strong>
  dberror:$error
</div>
    ";
  }
  return $data;
}
?>
```

BACKEND

connectdb.php

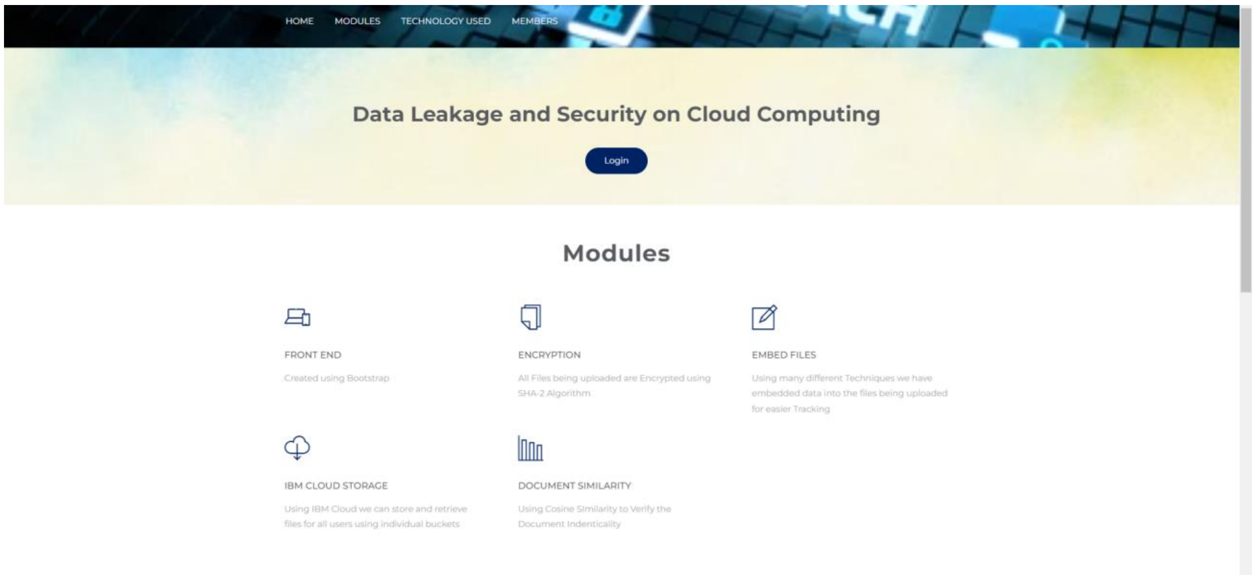
```
<?php
require_once("query.php");
$conn=mysqli_connect("localhost","root",'','data_
leakage');
if($conn){
    mysqli_query($conn, USER_TABLE_QUERY);
    mysqli_query($conn, DATA_FILE_TABLE_QUERY);
    mysqli_query($conn, KEY_REQUEST_TABLE_QUERY);
    mysqli_query($conn, LEAKER_TABLE_QUERY);
    mysqli_query($conn, LEAKED_MESSAGES_QUERY);
    mysqli_query($conn, ATTEMPTS_QUERY);
    if(mysqli_num_rows(mysqli_query($conn,
ADMIN_EXIST_QUERY)) == 0){
        mysqli_query($conn, CREATE_ADMIN_QUERY);
    }
}
else{
    die('Please make sure you have created a
database with name "Data Leakage" ');
}
?>
```

query.php

```
<?php
define('USER_TABLE_QUERY',"CREATE TABLE IF NOT
EXISTS users(
id INT(10) AUTO_INCREMENT PRIMARY KEY,
username VARCHAR(255) NOT NULL,
email VARCHAR(255) NOT NULL,
password VARCHAR(255) NOT NULL,
user_type ENUM('user','admin') DEFAULT 'user',
gender ENUM('male', 'female') DEFAULT 'male',
mobile VARCHAR(20) DEFAULT NULL,
admin_active VARCHAR(10) DEFAULT '0',
blocked ENUM('0', '1') DEFAULT '0',
profile VARCHAR(255) DEFAULT 'user_profile.jpg'
)");

define('DATA_FILE_TABLE_QUERY',"CREATE TABLE IF
NOT EXISTS data_files(
id INT(10) AUTO_INCREMENT PRIMARY KEY,
subject VARCHAR(255) NOT NULL,
file_name VARCHAR(255) NOT NULL,
file_size VARCHAR(255) NOT NULL,
sender_id VARCHAR(255) NOT NULL,
receiver_id VARCHAR(10) NOT NULL,
  secret_key VARCHAR(255) NOT NULL
  )");

  define('KEY_REQUEST_TABLE_QUERY',"CREATE TABLE IF
NOT EXISTS key_requests(
  id INT(10) AUTO_INCREMENT PRIMARY KEY,
  request_by_user VARCHAR(255) NOT NULL,
  request_to_user VARCHAR(255) NOT NULL,
  file VARCHAR(255) NOT NULL,
  secret_key VARCHAR(255) NOT NULL,
  status ENUM('pending','rejected','shared')
DEFAULT 'pending'
  )");

  define('LEAKER_TABLE_QUERY',"CREATE TABLE IF NOT
EXISTS leakers(
  id INT(10) AUTO_INCREMENT PRIMARY KEY,
  user_id VARCHAR(255) NOT NULL,
  subject VARCHAR(255) NOT NULL,
  file_id VARCHAR(255) NOT NULL,
  secret_key VARCHAR(255) NOT NULL
  )");

  define('ADMIN_EXIST_QUERY',"SELECT * FROM users
WHERE user_type='admin' LIMIT 1");
```

```
  define('CREATE_ADMIN_QUERY',"INSERT INTO
users(username, email, password, user_type,
admin_active)VALUES('admin', 'admin@gmail.com',
'12345', 'admin', '1')");

  define('LEAKED_MESSAGES_QUERY',"CREATE TABLE IF
NOT EXISTS leaked_messages(
      id INT(10) AUTO_INCREMENT PRIMARY KEY,
      user_id VARCHAR(255) NOT NULL,
      file_id VARCHAR(255) NOT NULL,
      created_at DATETIME
  )");
  define('ATTEMPTS_QUERY',"CREATE TABLE IF NOT
EXISTS attempts(
      id INT(10) AUTO_INCREMENT PRIMARY KEY,
      user_id VARCHAR(255) NOT NULL,
      file_id VARCHAR(255) NOT NULL,
      attempt VARCHAR(10) DEFAULT '2'
  )");
  ?>
```

10.2 SCREEN SHOTS



Data Leakage and Security on Cloud Computing

Modules

FRONT END

Created using Bootstrap

ENCRYPTION

All Files being uploaded are Encrypted using SHA-2 Algorithm

EMBED FILES

Using many different Techniques we have embedded data into the files being uploaded for easier Tracking

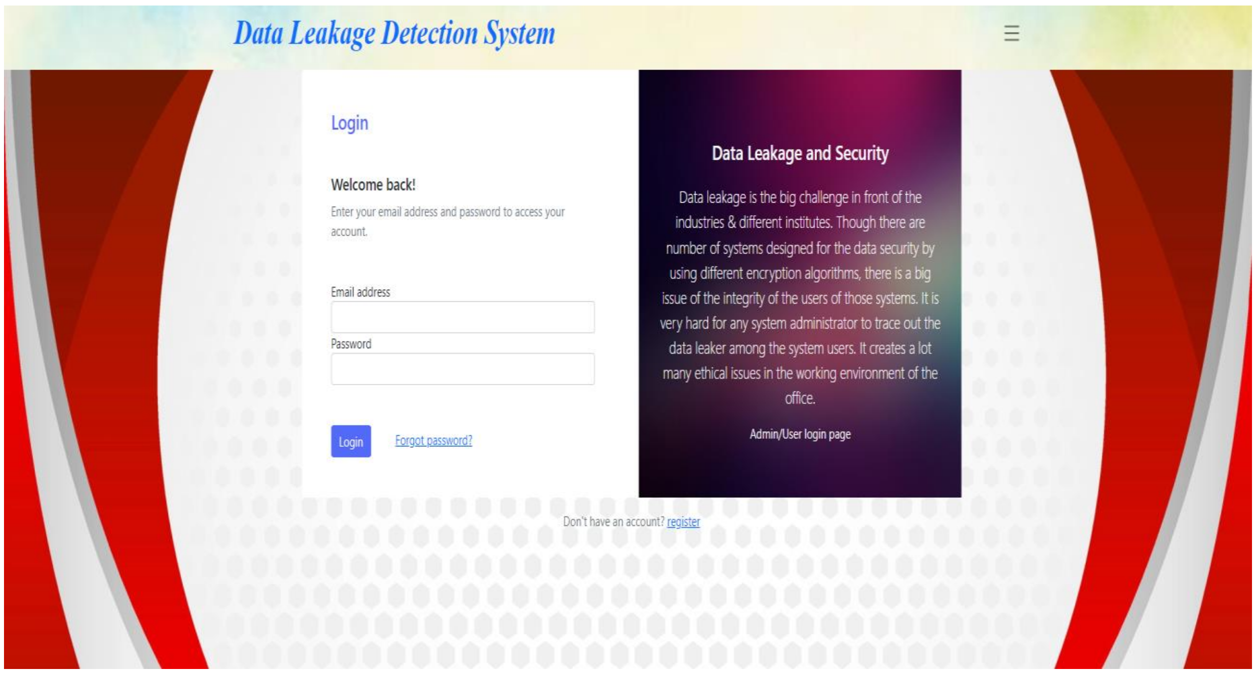
IBM CLOUD STORAGE

Using IBM Cloud we can store and retrieve files for all users using individual buckets

DOCUMENT SIMILARITY

Using Cosine Similarity to Verify the Document Indenticality

Fig 10.2.1 Home Page


Data Leakage Detection System

Login

Welcome back!

Enter your email address and password to access your account.

Email address

Password

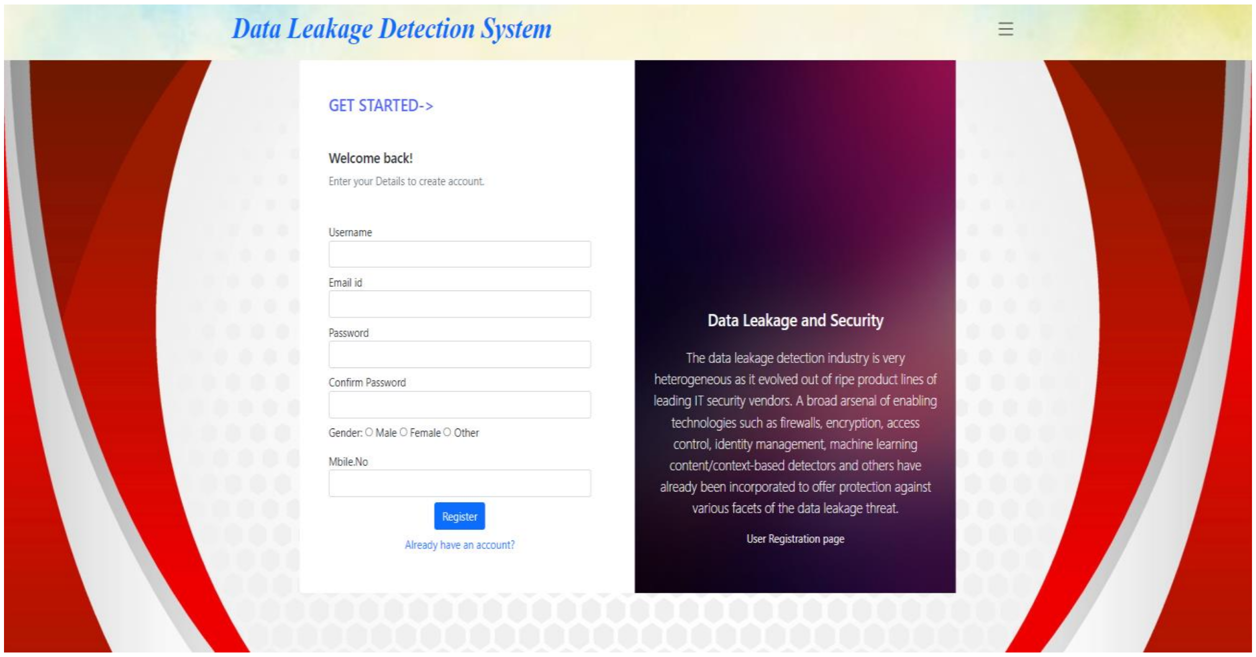
Data Leakage and Security

Data leakage is the big challenge in front of the industries & different institutes. Though there are number of systems designed for the data security by using different encryption algorithms, there is a big issue of the integrity of the users of those systems. It is very hard for any system administrator to trace out the data leaker among the system users. It creates a lot many ethical issues in the working environment of the office.

Admin/User login page

Don't have an account? register

Fig 10.2.2 Login Page



Data Leakage Detection System

GET STARTED->

Welcome back!

Enter your Details to create account.

Username

Email id

Password

Confirm Password

Gender: ○ Male ○ Female ○ Other

Mbile.No

Register

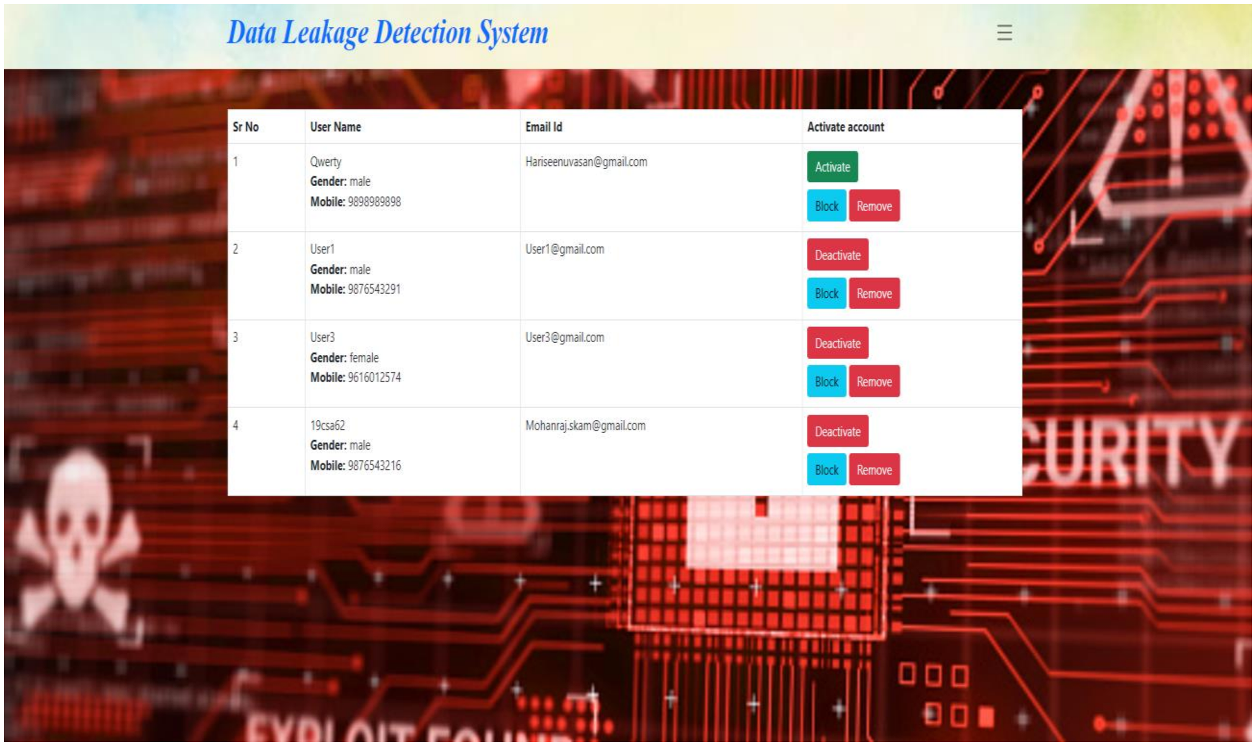
Already have an account?

Data Leakage and Security

The data leakage detection industry is very heterogeneous as it evolved out of ripe product lines of leading IT security vendors. A broad arsenal of enabling technologies such as firewalls, encryption, access control, identity management, machine learning content/context-based detectors and others have already been incorporated to offer protection against various facets of the data leakage threat.

User Registration page

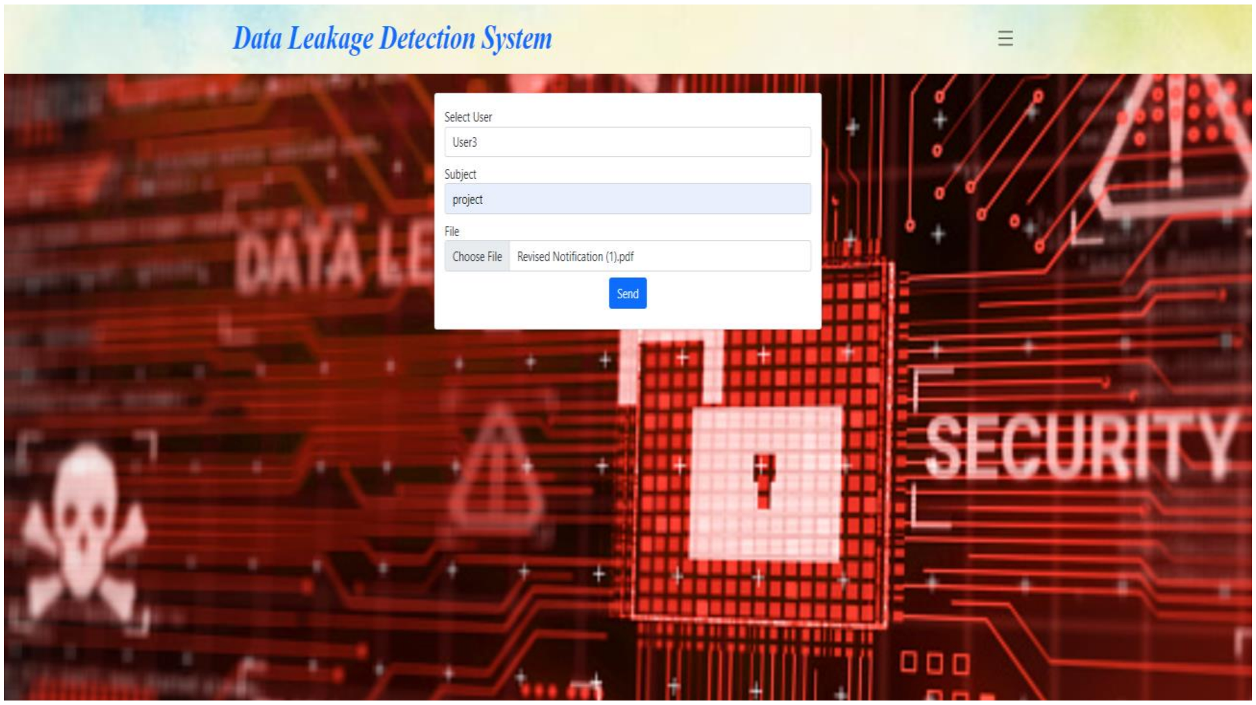
Fig 10.2.3 Register Page



Data Leakage Detection System

Sr No	User Name	Email Id	Activate account
1	Qwerty **Gender:** male **Mobile:** 9898989898	Hariseenuvasan@gmail.com	Activate Block Remove
2	User1 **Gender:** male **Mobile:** 9876543291	User1@gmail.com	Deactivate Block Remove
3	User3 **Gender:** female **Mobile:** 9616012574	User3@gmail.com	Deactivate Block Remove
4	19csa62 **Gender:** male **Mobile:** 9876543216	Mohanraj.skam@gmail.com	Deactivate Block Remove

Fig 10.2.4 Active User Status

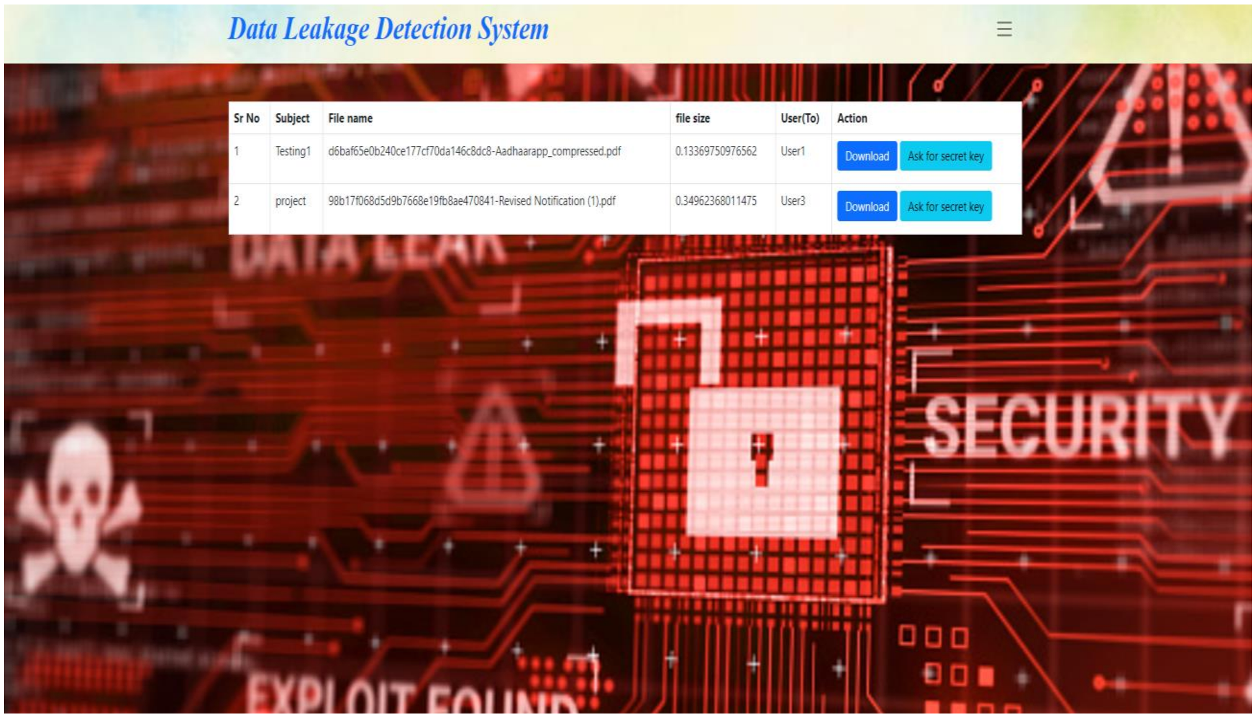


Select User: User3

Subject: project

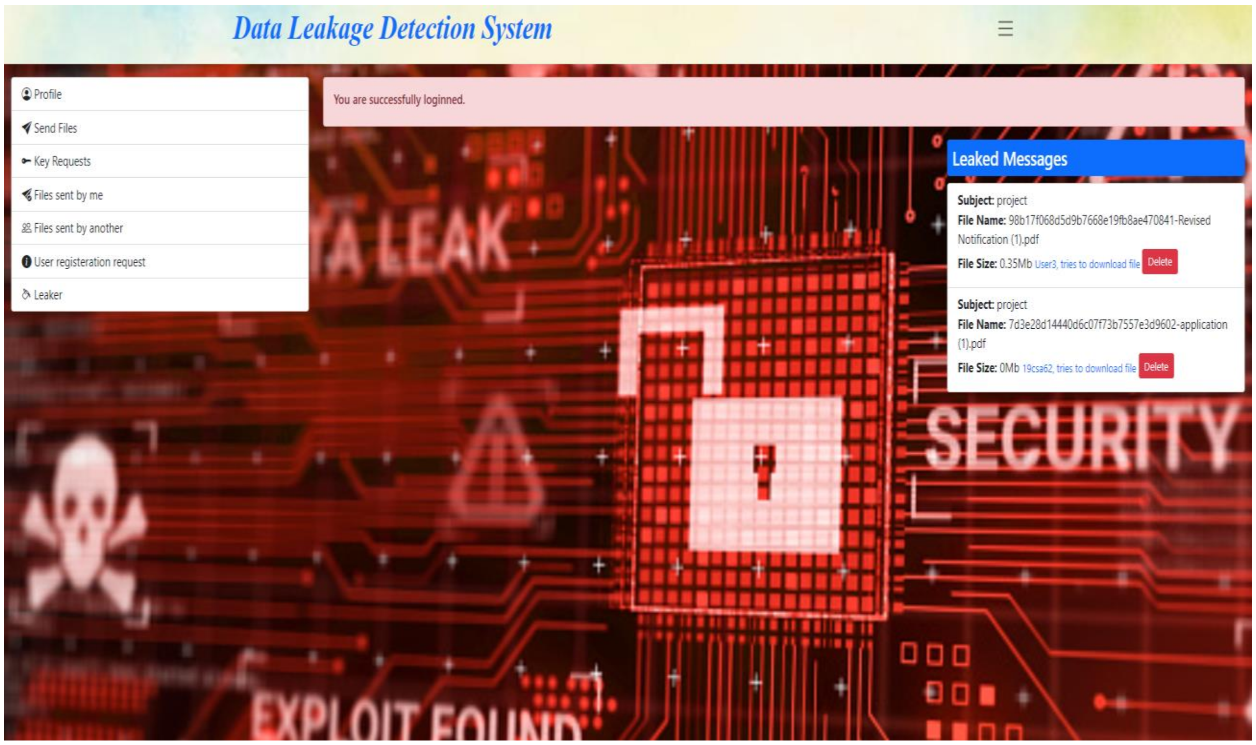
File: Revised Notification (1).pdf

Fig 10.2.5 Share Files



Sr No	Subject	File name	file size	User(To)	Action
1	Testing1	d6baf65e0b240ce177cf70da146c8dc8-Aadhaarapp_compressed.pdf	0.13369750976562	User1	Download Ask for secret key
2	project	98b17f068d5d9b7668e19fb8ae470841-Revised Notification (1).pdf	0.34962368011475	User3	Download Ask for secret key

Fig 10.2.6 Share Secret Key



You are successfully loginned.

Leaked Messages

Subject: project
File Name: 98b17f068d5d9b7668e19fb8ae470841-Revised Notification (1).pdf
File Size: 0.35Mb User3, tries to download file

Subject: project
File Name: 7d3e28d14440d6c07f73b7557e3d9602-application (1).pdf
File Size: 0Mb 19csa62, tries to download file

Fig 10.2.7 Leak Detection

CHAPTER 11
BIBLIOGRAPHY

Good Teachers are worth more than thousand books, we have them in Our Department.

REFERENCE LINKS

[1] Chaudhary, H., Chaudhary, H., & Sharma, A. K. (2022). Optimized Genetic Algorithm and Extended Diffie Hellman as an Effectual Approach for DOS-Attack Detection in Cloud. International Journal of Software Engineering and Computer Systems, 8(1), 69-78.

[2] Gupta, I., & Singh, A. K. (2022). A Holistic View on Data Protection for Sharing, Communicating, and Computing Environments: Taxonomy and Future Directions. arXiv preprint arXiv:2202.11965.

[3] Rishabh Singh, V Gokul Rajan, Galgotias University, U.P. India,May 2022, 'Data Leakage and Security on Cloud Computing', vol. 127, no. 1-2.

[4] Alshammari, S. T., & Alsubhi, K. (2021). Building a reputation attack detector for effective trust evaluation in a cloud services environment. Applied Sciences, 11(18), 8496.

[5] Wang, X., Pan, Z., Zhang, J., & Huang, J. (2021). Detection and elimination of project engineering security risks from the perspective of cloud computing. International Journal of System Assurance Engineering and Management, 1-9.

[6] Sharma, A., Singh, U. K., Upreti, K., & Yadav, D. S. (2021, October). An investigation of security risk & taxonomy of Cloud Computing environment. In 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC) (pp. 1056-1063). IEEE.

[7] Al-Shehari, T., & Alsowail, R. A. (2021). An insider data leakage detection using onehot encoding, synthetic minority oversampling and machine learning techniques. Entropy, 23(10), 1258.

[8] Chandu Vaidya etl. & BE scholars "Data leakage Detection and Dependable Storage Service in cloud Computing" IJSTE volume 2 issues 10 April 2021 ISSN online 2349-784.

[9] Khobragade, P. K., & Malik, L. G., “Data Generation and Analysis for Digital Forensic Application Using Data Mining”. In Communication Systems and Network Technologies (CSNT), 2019 Fourth International Conference on (pp. 458-462). IEEE. April, 2019

[10] M.Sai Charan Reddy, T. Venkata Satya Yaswanth, T. Gopal, L. Raji, Dr. K.Vijaya, “DATA LEAKAGE DETECTION USING CLOUD COMPUTING”, International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056, Volume: 06 Issue: 03, Mar 2019.

[11] Herrera Montano, I., García Aranda, J.J., Ramos Diaz, J. et al. Survey of Techniques on Data Leakage Protection and Methods to address the Insider threat. Cluster Comput 25, 4289–4302 (2022).

[12] Yasmin R., Memarian, M.R., Hosseinzadeh, S., Conti, M., & Leppänen, V. (2018). Investigating the Possibility of Data Leakage in Time of Live VM Migration. In: Dehghantanha A., Conti M., Dargahi T. (eds) Cyber Threat Intelligence. Advances in Information Security, vol 70. Springer, Cham.

[13] Prof. Sushilkumar N. Holambe, Dr.Ulhas B.Shinde, Archana U. Bhosale, “Data Leakage Detection Using Cloud Computing”, International Journal of Scientific & Engineering Research, Volume 6, Issue 4, April 2020, ISSN 2229-5518

[14] Muhammad Azizi Mohd Ariffin, “Data Leakage Detection in Cloud Computing Platform”, August 2019, International Journal of Advanced Trends in Computer Science and Engineering 8(1.3):400-408, DOI:10.30534/ijatcse/2019/7081.3201

[15] R. Naik and M. N. Gaonkar, "Data Leakage Detection in cloud using Watermarking Technique," 2019 International Conference on Computer Communication and Informatics (ICCCI), 2019, pp. 1-6, doi: 10.1109/ICCCI.2019.8821894.

[16] Chhabra, S., & Singh, A. K. (2022). A Comprehensive Vision on Cloud Computing Environment: Emerging Challenges and Future Research Directions. arXiv preprint arXiv:2207.07955.

[17] Singh, P., & Ranga, V. (2021). Attack and intrusion detection in cloud computing using an ensemble learning approach. International Journal of Information Technology, 13(2), 565-571.