Oblivious Text Retrieval from Untrusted Database Model

A Dissertation

submitted in fulfillment for the award of the degree

Master of Technology,
Computer Engineering

Submitted by

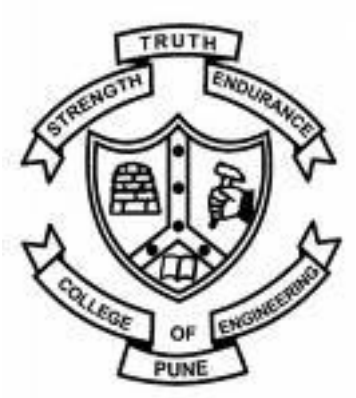
Harshal D. Badgujar

MIS No: 121122002

Under the guidance of

Prof. Sunil B. Mane

College of Engineering, Pune



DEPARTMENT OF COMPUTER ENGINEERING AND INFORMATION TECHNOLOGY,
COLLEGE OF ENGINEERING, PUNE-5

June, 2013

DEPARTMENT OF COMPUTER ENGINEERING
AND
INFORMATION TECHNOLOGY,
COLLEGE OF ENGINEERING, PUNE

CERTIFICATE

This is to certify that the dissertation entitled

"Oblivious Text Retrieval from Untrusted Database Model"

has been successfully completed

by

Mr. Harshal D. Badgujar

MIS No. 121122002

as a fulfillment of End Semester evaluation of

Master of Technology.

SIGNATURE
Prof Sunil B. Mane
Project Guide
Dept. of Computer Engineering
and Information Technology,
College of Engineering Pune,
Shivajinagar, Pune - 5.

SIGNATURE
Dr. J. V. Aghav
Head of Dept.
Dept. of Computer Engineering
and Information Technology,
College of Engineering Pune,
Shivajinagar, Pune - 5.

Dedicated to

My Father

Mr. Diwakar P. Badgujar,

My Mother

Sangita D. Badgujar

for their continuous love and support that they have given me!

And also to

All my family members

for continuously loving me in my success and in my failures.

Acknowledgements

I express my sincere gratitude towards my guide Prof. S.B.Mane for his constant help, encouragement and inspiration till date the project work. Without his invaluable guidance, this work would never have been a Reached to this level. I would also like to thank all the faculty members and staff of Computer and IT department for providing us ample facility and flexibility. Last, but not the least, I would like to thank my classmates for their valuable Suggestions and helpful discussions. I am thankful to them for their unconditional support and help in the last 3 months till this end-term evaluation.

Harshal D. Badgujar

College of Engineering, Pune

Abstract

Today, there is a huge development in Information Technology field. Many organizations are providing various types of services to millions of users. To provide different kinds of services, they need to handle large amount of different type of data. Nowadays, many organizations prefer to adopt new technology as outsourcing their data on database servers to store sensitive, important and huge amount of data.

However, there are issues related to backup services such as privacy and ownership, data security, service provider's reliability, controls and uploading of sensitive and important data etc. The private data stored on untrusted server should be secure and should be accessible to authorized users only. The data should be easily retrievable, whenever the user requires it.

Here, we are proposing a solution to retrieve sensitive data from untrusted database servers easily and securely. Also, the server will not be able to deduct what user is querying for and user also will not be able to get more data than what he wants. In case of data outsourcing, data encryption is a good approach enabling data owner to retain its control over outsourced data. We here are focusing on enabling encrypted outsourced data search with privacy assurance.

Keywords - Information Technology, Untrusted Service Providers, Oblivious Data Retrieval.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the recent past, a developing nation tended to be seen as a country that lacked access to modern technology. Today, increasing globalization means that rapid communication; market forces and lower import restrictions can help make a new technology available anywhere that it might be useful. New technologies are transforming social and economic prospects globally. The biggest impact has come from information and communication technologies (ICTs)--the computers, mobile telephones and satellite communications at the core of the modern information society.

With the advancement of technology, there is need to handle, process & manage huge amount of information & data such as text files, various types of data, audio, video files etc.. Vast quantities of data are made available to different people at an ever-accelerating rate, and sophisticated and innovative approaches to data discovery, data mining and analysis are being developed to extract the full scientific content of this data tsunami.

We have investigated two aspects of data processing and management that have been raised nowadays.

- What is the cheapest and most efficient way of processing data to create new products, given that data are distributed across the world and that processing them often requires access to high-end compute and storage resources? Should data be uploaded to the software, or should software be built close to the data?
- How can we capture the processing histories of these third-party products and capture their processing histories and provenances?

Outsourcing the data on database server for storage is now receiving a lot of attention nowadays. Cloud computing can also provide database as a service (DaaS) for efficient data storage and processing.

1.1 Database Outsourcing

Today, many organizations are expanding. They need to handle large amount of data daily. There are many non-IT organizations, for which it is difficult to handle and maintain this data. Massive amount of different types of data need to be uploaded, stored, managed, processed and retrieved, whenever needed. The main problems in handling huge data on local databases are viz., a) huge storage space is needed b) security to data must be provided so that it cannot be hacked by outside user, c) non-IT organizations do not have proper setup and soft wares and cannot waste manpower to handle and process the data, d) proper encryption techniques should be applied so that it is not accessible to outside users, e) non-IT organizations cannot afford maintenance of large amount of data by themselves, f) it should be easily retrievable by users.

1.1.1 What is Database Outsourcing

Due to the complexity of managing data on its own, in modern enterprise networks, individual users and organizations are using the services of external servers to store their data. Many outside servers provide database as a service. This is called as data outsourcing. Nowadays, outsourcing data to database servers for uploading and retrieval is gaining a lot of importance. This approach is proving to be more efficient than storing data on local database. Many organizations are now gaining benefits from outsourcing their data on database servers for storage.

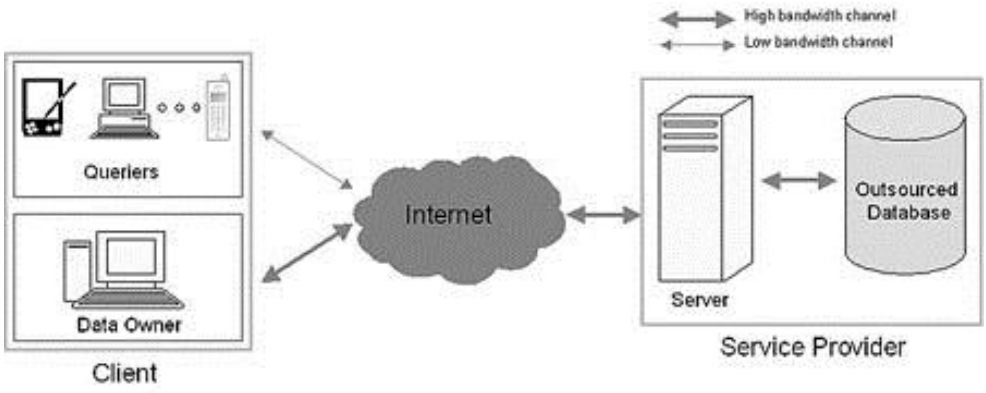


Figure 1 Outsourced Database Model

1.1.2 Increase in Outsourcing Data

As the new technology developed & emerged, no. of users using technology increased rapidly. Database as a service (Daas) is now receiving a lot of attention as a new service. There are different reasons why the need of outsourcing the data in today's environment aroused and is becoming popular. :

- Increasing no. of technology users.
- Increase in amount of data to be processed.
- Overhead in storing, managing & processing huge amount of data by the user himself.
- Advantages the data outsourcing is implementing such as:

    - Ease of functionality by outsourcing data.
    - Efficient storage & maintenance facilities.
    - Secure from unauthorized access.
    - Huge amount of space is saved.
    - Cost reduction in data maintenance.
    - Reduction in liability.

1.1.3 Issues in existing outsourcing database servers

There still are some issues with outsourcing the data, due to which many organizations still hesitate to outsource their data. [2] The main issue is the untrustworthiness of the server. Organizations do not fully trust the server with their data. As there can be many clients with one service provider, one's secret data can be viewed by another user. Also, server can make inference about organization's data after repeated queries. The main problem of this untrustworthiness on servers is lack of proper encryption techniques, which can guarantee about privacy & security of data. Absence of multilayer encryption in existing systems creates suspicions in users' minds.

User privacy & data secrecy are main issues, due to which organizations do not trust the database servers easily. Resources in database service provider can be shared among a large number of users. So the service provider must ensure the security of data stored in the system. The company must have confidence in the server if they want to store their private data on that server. Governance and security are crucial to success of data outsourcing. The traditional security mechanism cannot protect the outsourced data entirely. The main security problems include data security, user privacy protection.

The Various issues that can be and should be handled are:

- Security of Data
- Privacy & Ownership
- Service Provider's Reliability
- Revocation of Rights
- Controls & Uploading of Sensitive data
- Ease of Accessibility & Retrieval

1.2 Data Retrieval

As the data produced by individuals and enterprises that need to be stored and utilized (e.g. emails, personal health records, photo albums, tax documents, financial transactions, etc.) is rapidly increasing, data owners are motivated to outsource their local complex data management systems onto the outside server for its great flexibility and economic savings. To protect data privacy and combat unsolicited accesses, sensitive data may have to be encrypted by data owners before outsourcing.

1.2.1 Oblivious Data Retrieval

As we have seen earlier, due to the complexity of managing huge data, organizations are using the services of external servers to store data. The text data of organizations should be stored on server in encrypted form to ensure security. Due to untrustworthiness of server, the data should be protected from server & other users also. Oblivious retrieval means the data should be uploaded & retrieved obliviously. The problem is to query and retrieve data while simultaneously satisfying the criteria of both the user and the database privacy. Oblivious transfer protocols are capable of ensuring the privacy of both the user and the server privacy at the same time. In this, the database server does not know the exact user query or what user is querying for, and the user does not learn of any other data than his own from the database.

Here, we are proposing idea for oblivious retrieval of text data from untrusted database server. Our system implements oblivious transfer of text data from database servers. The idea is to upload encrypted text data on server so that data privacy is ensured; and to retrieve that data obliviously so that server will not be able understand user queries.

Chapter 2

Literature Survey

2.1 Related Search

This chapter will cover the prior work and related papers in this field which are helpful in defining basic idea of our work. Much research had been done on data security, text-based search & retrieval, but none has combined both the uploading & retrieval of data with ensuring security of user & server both at the same time.

2.1.1 Research on Cloud Computing Security problem and Strategy [2]

The majority of cloud computing infrastructure currently consists of reliable services delivered through data center that are built on servers with different levels of virtualization technologies. The services are accessible anywhere in the world, with The Cloud appearing as a single point of access for all the computing needs of consumers. The cloud computing changed the style of software. The data can be stored in the cloud system and the user can use the data at any time and in anywhere. The data is often stored in the private or personal system such as PC. The cloud computing can guarantee the data security and the user do not protect the data by himself again. So the cloud computing must ensure the security of data stored in the cloud system. Many companies provide the cloud computing platform such as Google, IBM, Microsoft, Amazon, VMware and EMC. As the cloud computing system has more data which may be the private data of user, the data must not be destroyed or grabbed. Because the data in the cloud system may be important for the user, the hacker may pay more attention to get the data. The system must be protected more carefully than the traditional system. The company uses the cloud system and stores the data in it. The data can be seen by other people who are not person

of company. The company must have confidence in the cloud computing if they want to store the private data in the cloud system.

2.1.2 Keyword Search Protocol in the Public Database Model [7]

Keyword search is a fundamental database operation. Typically, a keyword search protocol consists of two participants: a server, who holds a database comprised of a set of payloads (database records) and their associated keywords, and a client, who sends queries consisting of keywords and receives the payloads associated with these keywords. We remark that databases associated with keywords, can be public, private or hybrid. As a result, the solutions to keyword search protocols for each type are different.

In the public-database setting, the data-item is public but a client wishes to retrieve some data-item or search data item, without revealing to the server which item it is. In the public-database model, a server does not require to commit to a database to which a client's search is effectively applied, thus the concept of keyword search is quite different from that of a zero-knowledge database [9], where a sever is required to commit to its input and the goal of zero-knowledge databases is to provide privacy for the server but not for the client.

2.1.3 An Oblivious Data Retrieval Protocol [8]

Many organizations are using database as service of external storage servers such as outsourced databases to manage their information. The external server provides support for querying and retrieval of information. Sensitive information such as maps, blueprints, machine part designs, scanned documents, photos, bio-metrics etc. are stored on database servers. This application scenario brings out several important concerns such as protecting confidentiality of the data and protecting the privacy of users among others. These requirements are essential for preventing the leakage of critical information such as data collaboration among users and profiling the access patterns of individual users.

Thus, it is necessary to explore the development of secure data retrieval protocols that satisfy these important requirements.

The basic operation of oblivious retrieval protocols is to allow the user to query the database for a set of values. The database server includes all the data values in the response but the user can only retrieve those values which he queried for.

2.1.4 Security analysis for privacy preserving search of multimedia [9]

With the increasing popularity of digital multimedia and the advent of the computing paradigm, a fast growing amount of private and sensitive multimedia data are being stored and managed over the database servers. To provide enhanced security and privacy protection beyond traditional access control techniques, privacy preserving multimedia retrieval techniques have been proposed recently to allow content-based multimedia retrieval directly over encrypted databases and achieve accurate retrieval comparable to conventional retrieval schemes.

2.2 Existing protocols for data retrieval

Oblivious Transfer protocol (OT) is first introduced and is well-studied as a secure multi-party protocol [5]. OT is applied for keyword search in [11]. Content based Retrieval techniques are found in [1], while rank-based search over large document collection is studied in Confidentiality-preserving rank-ordered search [12]. Text search of encrypted data from database server is introduced in Practical techniques for searches on encrypted data [13].

Previously, data stored on database server was used to be retrieved using Text-Based Retrieval or Content-based Retrieval.

2.2.1 Problems with Existing Solutions

There are many inadequacies present in existing solutions. The main concern is about secrecy of user & security of user's data.

- In Keyword search protocol [11], only one keyword is searched in the database and that particular word is retrieved only. It does not support full text file retrieval from the server.
- In Rank-ordered search [12], server cannot know about user's data or query, but security of server is questionable.
- Practical techniques for searches on encrypted data [13] also does not provide security to both the user & the server.
- Also, time overhead for data uploading & retrieval from server is one of the main issues to be looked at. System should not incorporate uploading and retrieval overhead.

The main issue with existing retrieval systems is that they provide algorithms and security for retrieval of data from untrusted servers only. While our system provides facility for secure uploading of text data on untrusted server, as well as "oblivious" retrieval of that text data from the database server.

2.3 Oblivious Retrieval Technique

We consider the novel problem of oblivious retrieval of text data from an outsourced database server. Different organizations outsource their text data (e.g. emails, health records, client information, scanned documents etc.) on database server for storage. The problem is to i)upload this data on the server in encrypted form to ensure security of data & ii) to query and retrieve data while simultaneously satisfying the criteria of both the user and the database privacy. Here, we describe the first such attempt to develop an oblivious retrieval protocol that can achieve the goals of user and database privacy, which allows the user to query for the data of interest. In this phase, the database server does not know the exact user query and the user does not learn of any other data in the database.

Also, we give protocol, where user does not rely on server for security of his data; data security is ensured by user himself while uploading.

Chapter 3
Motivation and Background

With the advancement in technology, no. of users using technology increased rapidly. Even non-IT organizations like Medical, Small schools are using different services of service providers. Earlier, their data was used to be maintained in traditional format such as paper work. But today, as IT is including everything and everyone in it, the amount of data that need to be stored and maintained increased rapidly. It becomes difficult to handle and store this huge data on local database storage. Therefore, organizations started to outsource their data on untrusted database server.

These servers are untrustworthy, so there is still hesitation for outsourcing data on servers. So, I considered about this problem and thought of providing security and privacy to user as well as database server with efficiency. The existing systems of untrusted servers have lots of problems. Most of the systems implemented in this area provide facility for only secure retrieval of data from outsourced untrusted server. Here, I am considering providing security at the time upload and retrieval also, while achieving obliviousness.

The Oblivious Retrieval has many applications. The system can be used in providing Database as a Service (DaaS). Therefore, I choose the problem of Oblivious Text Retrieval from Untrusted Server.

3.1 Problem Definition

Here, we propose the retrieval protocol for text data obliviously from database server.

> ***"To query and retrieve uploaded text data, while simultaneously satisfying the criteria of both the user and the database privacy."***

Two phases in the system:

i) To upload text data on the database server for storage in encrypted form.
ii) To retrieve that data "Obliviously" from server, thus providing security to both the client & server.

Here, the client should be able to retrieve data from server. User desires to retrieve some data-item or search data-item, while not revealing to the server that item's identity. Also, the user should not be able to get additional information than what he needs.

3.2 Scope of Research

Scope of this research is not limited to one particular domain. It spans over multiple domains. Domains in which extensive research can be done are Private Information Retrieval, Content Based Retrieval and Oblivious transfer i.e. security. Research papers in this area suggest keyword-based search and secure retrieval over Cloud platform as well as secured transfer techniques of the information but not a single paper talks about fast as well as secure retrieval of data from untrusted servers using oblivious technique.

This system can be evolved in many ways. :

- This work is fundamentally different from private information retrieval where the privacy of only one of the parties is ensured.
- Also, Multiparty Communication can be done, where multiple parties can ask for the data without revealing their identity [5].
- Data Retrieval techniques over multiple files can also be parallelized using Map-Reduce or CUDA [10].

3.3 Objectives

- To Study Existing Retrieval Techniques for Text Data
- To propose an efficient algorithm for retrieving text data securely and privately from untrusted database server
- To Implement and evaluate proposed solution
- To compare proposed solution with existing solutions

3.4 Some Basic Entities

Client: The clients are actually the data users, who read, download and access the data as per their privileges. The client may be a database authorized employee in the organization.

Untrusted Server: In the scenario of database outsourcing the external database server is not trusted so that we have to keep our data on service provider in such a way that, the private information should not get revealed to the service provider itself.

Data Owner: The organization is the database owner. The data owner stores its data to the database server. For this purpose organization has to take care of its data. So, the organization has to consider all the security issues, in order not to reveal the information to the untrusted server.

3.5 Symmetric Encryption

In the proposed system, the aim is to provide security to both the user and server at the same time (obliviousness). To achieve this, there is a need to provide proper encryption to the data at user and server side. There are various encryption techniques present today, which can be used in the system. We are using symmetric encryption technique to

achieve privacy and security. As the data is uploaded by user using his own key and also can be queried with the same key, server does not need to read or see contents of its database. It just stores data as it comes from client. User can use same key while uploading data and to decrypt that data after retrieval. Therefore, we have used a symmetric encryption technique with single key to provide encryption to the system.

There are many symmetric key encryption techniques present today, such as DES, Triple DES, Blowfish, AES, IDEA etc.. There are also some stream cipher algorithms such as RC4. Out of these, we have chosen AES due to following reasons.

- There have been attacks detected in DES and Triple DES, while no known attacks are possible on AES today.
- Stream ciphers encrypt the digits (typically bytes) of a message one at a time, while AES uses Block Ciphers and takes a number of bits and encrypts them as a single unit, padding the plaintext so that it is a multiple of the block size, which provides better security.
- IDEA is just a minor revision and expansion of DES and the very simple key schedule makes IDEA weaker, producing weaker encryption.
- AES is more effective and commonly used nowadays.

Based on the above facts and study, we are using symmetric Advanced Encryption Standard (AES) for encryption & security purpose. Here is some basic information about it. :

3.5.1 Advanced Encryption Standard (AES)

AES relies on a design principle called as substitution-permutation network, and is quick in both software and hardware. In contrast to DES, AES doesn't use a Feistel network. AES is a variant of Rijndael that contains a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits. In contrast, the Rijndael specification as such is fixed with block and key sizes that will be any multiple of thirty two bits, each with a minimum of 128 and a most of 256 bits.

AES operates on a 4×4 column-major order matrix of bytes, termed the state, though some versions of Rijndael have a bigger block size and have extra columns within the state. Most AES calculations are done in a special finite field. The key size used for an AES cipher specifies the count of repetitions of transformation rounds that convert the input, referred to as the plaintext, into the final output, referred to as the cipher text.

The numbers of cycles of repetition are as follows:

- 10 cycles of repetition for 128-bit keys.
- 12 cycles of repetition for 192-bit keys.
- 14 cycles of repetition for 256-bit keys.

Each round consists of many process steps, every containing 5 similar however completely different stages, as well as one that depends on the encryption key itself. A collection of reverse rounds are applied to transform cipher text back to the first plaintext using a similar encryption key.

High-level description of the algorithm

KeyExpansion—round keys are derived from the cipher key using Rijndael's key schedule.

1. InitialRound
    1. AddRoundKey—each byte of the state is combined with the round key using bitwise xor.
2. Rounds
    1. SubBytes—a non-linear substitution step where each byte is replaced with another according to a lookup table.
    2. ShiftRows—a transposition step where each row of the state is shifted cyclically a certain number of steps.
    3. MixColumns—a mixing operation which operates on the columns of the state, combining the four bytes in each column.
    4. AddRoundKey

3. Final Round (no MixColumns)
    1. SubBytes
    2. ShiftRows
    3. AddRoundKey

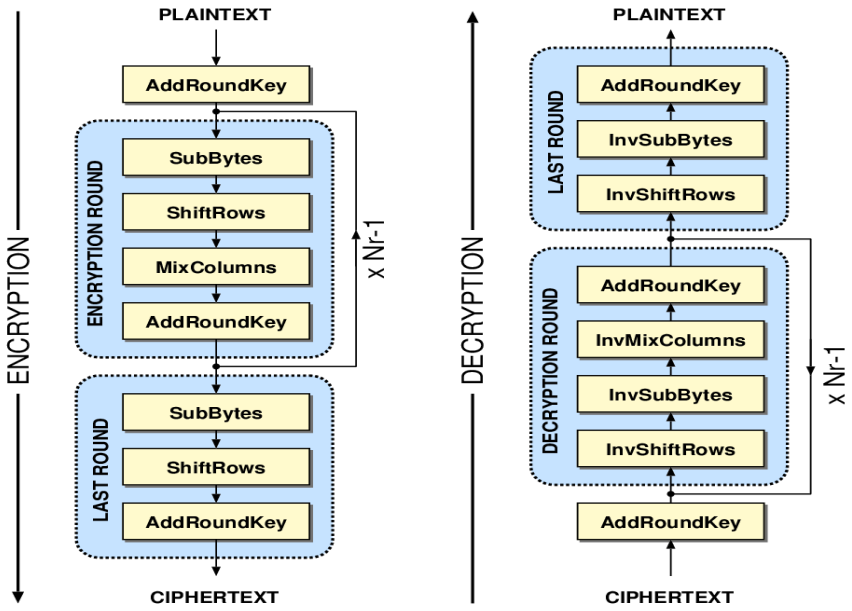


Figure 2 Advanced Encryption Standard (AES)

3.5.2 Advantages of AES

- AES doesn't use a Feistel network and is fast in both software and hardware.
- Advanced Encryption Standard not only assures security but also improves the performance in a variety of settings such as smart cards, hardware implementations etc.
- AES is federal information processing standard and there are currently no known non-brute-force direct attacks against AES.
- AES is strong enough to be certified for use by the US government for top secret information.

3.6 System Requirement Specifications

Hardware:

- Minimum one computer system with Intel's or AMD's processor and 2 GB of RAM is required.
- Machines having Intel's core series processor (i3, i5, i7) of high frequency with 4 GB RAM will definitely perform better than machines having older processor version working on less frequency and less RAM.

Software:

- Operating System: Any open source Linux operating system
- IDE: Eclipse environment
- Coding Language: Java SE Development kit (JDK)
- Database: Any database such as MySQL
- Open Source Web Server Apache Tomcat

Chapter 4

System Design

Here, we consider the problem of uploading & retrieval of text data from untrusted database server. The database server provides support for querying and retrieval of information. User first uploads his data on the server in encrypted form of his own to ensure security to data. Then user can send queries to retrieve that data from server without revealing to server what he is querying. Our idea brings out several important concerns such as protecting confidentiality of the text and protecting the privacy of users among others.

4.1 Basic description of the scheme

The basic outline and design of the system is as explained below. The detailed implementation will be discussed in the next chapter. Here, we give the overall working of the system and the work flow of uploading and retrieval.

4.1.1 System Architecture

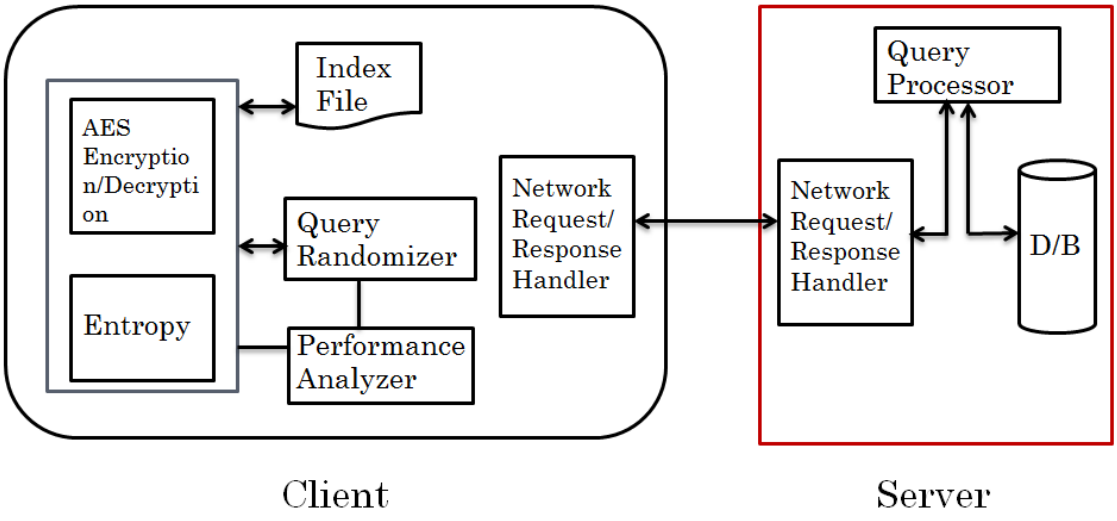


Figure 3 System Architecture

4.1.2 Architecture Explanation

As shown in the architecture fig, there are two components of system trusted client and untrusted server. The client contains various components. The working of each block of the architecture is as explained below. :

Encryption/Decryption module:

Encryption/Decryption Module is responsible for:

- Encrypting plaintext file using AES algorithm
- Store key and plaintext file name and encrypted file name mapping into the index file
- Decryption of query response

Entropy Module:

Entropy Box in the diagram is responsible for calculating the entropy of encrypted text. Firstly, file is encrypted using AES 'word-by-word' and then to provide better security to user data, its entropy is calculated using Base-64 encoding. This entropy is sent to the server to store.

Index File:

At user side, one index file is maintained, which contains original file name, encrypted file name along with the encryption key, by which that file is encrypted. It stores mapping between plaintext file and encrypted file as well as encryption keys. This index is helpful to decrypt the file after retrieval from server.

Query Randomizer:

Query randomizer is responsible for addition random queries in addition to original query. When user wants to retrieve some files containing some word, it sends that word's entropy to server. To confuse server about what user is querying and to create randomness, Query Randomizer is used to protect user query analysis by server.

Query Processor:

It receives queries from Network Request/Response Module and processes those queries. It gets results from D/B and forwards the responses to Network Request/Response Module.

Network Request/Response Handler:

Network Request/Response Handler at client side is responsible for sending network request to server and process response from server.

Network Request/Response Module at server side is responsible for receiving multiple requests from client and forwards them to Query processor Module. It also receives responses from Query processor module and forwards them to the Client.

Performance Analyzer:

User also contains performance analyzer module to perform analysis time taken during query execution. It gives performance overhead of various operations.

It is responsible for measuring encryption time, decryption time, network overheard and server processing time. Details of these times are given in the later chapter.

4.1.3 Work Flow

The following are the steps showing working flow of our proposed system. :

- Firstly, if client wishes to store data on database server, his unique id is created automatically.
- Client encrypts his data i.e. text files using AES algorithm.
- Then that cipher text is converted into Base-64 format & this entropy is stored on server.
- When client wants to retrieve file containing particular word, he encrypts it using AES & then convert it into Base-64 format.

- This entropy is sent to the server. Server receives the query from particular client, so it searches that encrypted word in all the database entries of that client.
- As all the data is stored on server in encrypted format, server tries to match the query within all the files. If the match is found, the results are sent back to the user.
- The names of all the files, in which that word is found, are given back to user.
- From that, user can select which file or all files to download from server and requests accordingly.
- After receiving full file, it is decrypted at client side using same decryption key & client thus retrieves the files of his choice.

Chapter 5
Implementation of System

We'll now see the detailed implementation of the system. We implemented our approach using Java SE development Kit and Eclipse IDE. We evaluated the prototype of our approach on a computer with an Intel Core i5 2.58GHz CPU and 4GB RAM, running Ubuntu 12.04. We provided a GUI to user at client side.

Each client registers with server and get a unique ID. This ID used to retrieve files only from client area at server side. If client wants to upload a file on server, we asked client to provide an encryption key, which we used to encrypt using AES algorithm. To achieve this we used "Cipher", "SecretKeySpec" classes from "Crypto" module in Java. In addition, we calculated entropy of encrypted text using Base64 encoding and sent entropy to server to store. When we want to download a file from server, we retrieve entropy from server, and then we calculate Base64 decoding to get encrypted data, which we decrypt, using original key and AES decryption technique to get original plaintext.

5.1 Detailed Implementation

There are two phases in our project. : A) Uploading Phase and B) Retrieval Phase. In both the phases, security of user data is ensured.

5.1.1 Uploading Phase

As discussed earlier, this phase is related to uploading of text data on the database. The working of upload process is as follows. :

1. User selects one text file from his local database storage i.e. Computer/Hard Disk browsing through it as shown in fig..
2. One unique id for client is created on the database.
3. User enters encryption key for that file. Then file will be encrypted word-by-word.
4. Suppose, file contains n no. of words. This n no. of words will be encrypted separately using the encryption key provided by user.
5. For encryption purpose, AES algorithm is used. Every word in the text file will be encrypted using AES. If a word is too small, padding is added to that word during encryption.
6. On user side, one index is maintained, which contains the original file name, encrypted file name and the encryption key of that file, by which it is encrypted.
7. After AES encoding is done, the entropy of the word is calculated. To calculate entropy, we are using Base-64 encoding format. The word is converted into Base-64 encoding format.
8. Repeat this procedure until the whole file is converted into Base-64 format.
9. Upload this encrypted-entropy-converted file on the server by clicking Submit Button, which stores it on the MySQL in table format. The whole file will be uploaded on MySQL using clob feature of MySQL.
10. User does not see any encryption or implementation details while uploading his data. Unique file id is created for each file based on the encryption key provided.
11. User only sees his files stored on database in encrypted format. Any other user or server can only see data in encrypted form on database.
12. Thus, server cannot know anything about user's data and user cannot infer anything about other user's data, thus Obliviousness is achieved.

5.1.2 Algorithm for Upload Phase

Algorithm for uploading a file

1. **Input**: *file* - a file for encryption , *Key* - a key of encryption.
2. **Output**: *encfile* - encrypted file
3. **function** UPLOAD::AESENCRYPT(token, *Key*)
4. encData = Encrypt token using AES algorithm and *Key*
5. encEntropy = Calculate BASE64 encoding on encData
6. return encEntropy
7. **end function**
8. **for all** strLines € the *file* do > read line by line from *file*
9. *token* = new StringTokenizer(strLines) > tokenize each word in strLine
10. **while** token has more elements do
11. encLine += Upload::AESencrypt (token, *Key*)
12. **end while**
13. **end for**
14. Send *encLine* to server to store.

5.1.3 Retrieval Phase

The second phase of the system is the retrieval phase. The Retrieval Phase consists of searching of a word in all the files and retrieval of necessary files out of those, in which that word is found. We'll see now the detailed procedure of text search and retrieval from the untrusted server.

1. As shown in fig., one text box is provided to user to search the word he requires.

2. User types the word he wants to search in text box and then clicks the search button. After the button is pressed, internal processing for searching of word begins.
3. The purpose of word-by-word encryption is that while searching & retrieval phase, if user wants some file containing particular word, he can directly encrypt that word and send to server; server then can match that word with every word in the file.
4. The word is converted into encrypted format using AES. To retrieve all the files containing that word, the server has to search that word in all the files.
5. As every file is encrypted with different key, that word will be stored in different formats in all the files. Therefore, to search in all the files we have to send the word in different encrypted formats, so that server can match those different formatted words in every file.
6. So, the word will be encrypted using each key of each file. Suppose, there are n files stored of the user on database, each file in different encrypted format.
7. So, the word will be encrypted n times using encryption key of each file. After that, their entropy is calculated using Base-64 encoding format.
8. Then n queries will be sent to server, each query containing the encrypted word along with the file, in which to search it.
9. Server receives n queries, and as the data is stored on server in entropy format, each query will be processed and word will be matched in each file stored on the server.
10. As server receives the query from particular user having unique id, and as destination file in which to search the word is given with each query, the word will be searched in every file of that user's database entries only and not in the other database.
11. If server finds entropies in any files matching with queried entropy, the names of those files are returned to the server.
12. Out of n files, suppose word is found in m files. User then can choose which or all files to download out of those m files.

13. Using Download option shown above in fig. , user can download his required file.
14. Download file: When user selects his choice of file from list of files and click on Download button, that file will be retrieved from the database.
15. It will be decoded by Base-64 format and then decrypted using AES decryption key of that file. The encryption key of each file is maintained on user side in an index file. After the decryption, the contents of that file will be displayed to the user and thus, user retrieves his data.

5.1.4 Algorithm for Retrieval Phase

Algorithm for Searching and Retrieving a file

1. **Input**: *SearchWord* - a word to search for, *IndexFile* - It contains a list of files uploaded by user on the server and keys used for encryption.
2. **Output**: *FileList*- a list of files contains *SearchWord*
3. **function** UPLOAD::AESENCRYPT(*SearchWord, Key*)
4. encData = Encrypt *SearchWord* using AES algorithm and *Key*
5. encEntropy = Calculate BASE64 encoding on encData
6. return encEntropy
7. **end function**
8. **for all** strLines € the *IndexFile* do > read line by line from *Indexile*
9. *Key* = getKeyFrom(strLines)
10. *FileName* = GetFileNameFrom(strLines)
11. encWord = Upload::AESencrypt (*SearchWord, Key*)
12. Send *encWord* to Server for search

13. If (*ServerResons* is TRUE) FileList += FileName
14. **end for**
15. SDisplay *FileList* to User. > *FileList* contains names of files in which the entropy of *SearchWord* is found.

5.2 User Interaction with the Server

When user first comes in contact with the system, he interacts with Graphical User Interface (GUI) first. Graphical user interface (GUI) is the most significant part of any application. The better the design of GUI as well as better its simplicity, the more user-friendly the application is. When any user will come across this application, he or she is going to interact with interface firstly.

In our system, complete application including GUI is created in eclipse [19]. Eclipse is an open source integrated development environment (IDE), using which helps applications can be developed easily. GUI is developed using Java Swing Applet. A Java applet is a small application written in Java, and delivered to users in the form of bytecode. The user launches the Java applet from a web page. The Java applet then executes within a Java Virtual Machine (JVM) in a process separate from the web browser itself, yet it can appear in a frame of the web page, in a new application window, or in Sun's AppletViewer, a stand-alone tool for testing applets.

5.2.1 Upload File on Server

As the title suggests, this application provides oblivious uploading and retrieval of text files on untrusted server. So, first, files are uploaded on server.

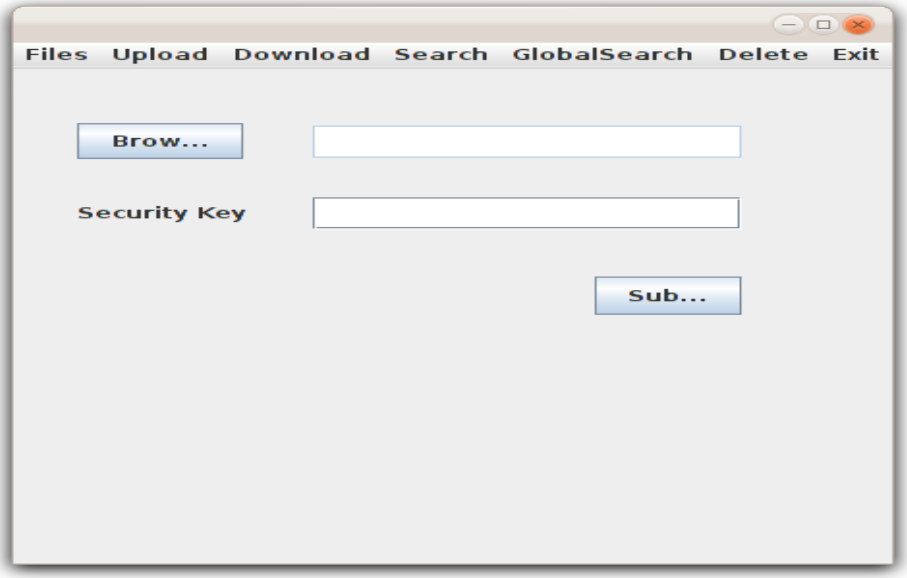


Figure 4 File Upload

To protect security of user data, file is uploaded in encrypted format, with the encryption key of user's choice. User can browse through his local data storage and can upload text file.

While uploading, user has a choice to give his own security key to encrypt file. File is encrypted using AES and then converted into Base-64 format. This version of file is stored on server. User does not see any encryption or entropy calculation on his side. That is calculated internally

5.2.2 Display screen to user

This window provides the user the facility to see his uploaded files. User sees the files uploaded along with their uploaded date.

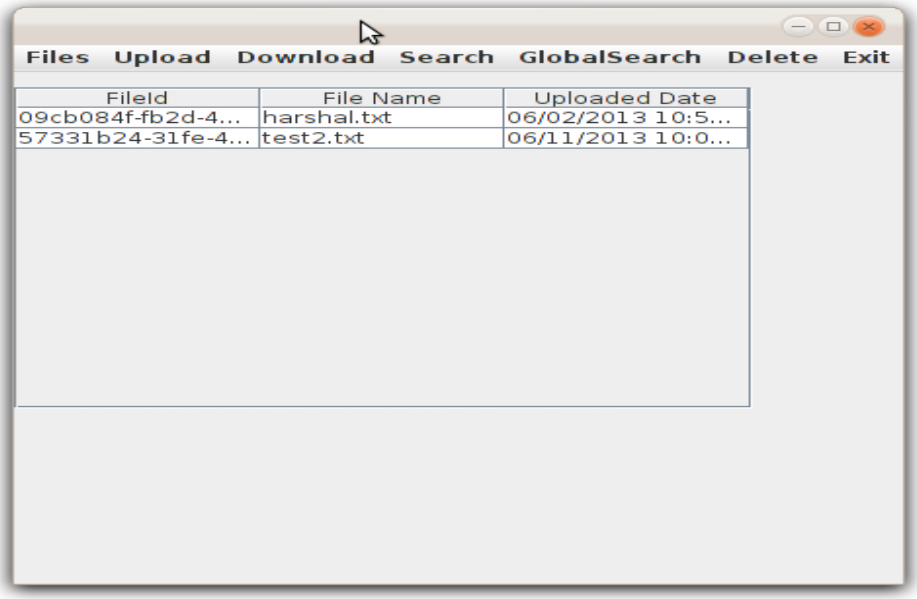


Figure 5 Display screen for user

5.2.3 Search a word in all the files

After uploading is done, our aim is to retrieve files containing particular word. This window provides the option to search & retrieve those files. User types his choice of word in text box. After clicking search, that word is encrypted using AES and converted into Base-64 entropy format and sent to the server. The entire user's data on server is stored in encrypted format only. Therefore, server matches the encrypted word within his database and if the matches are found, the list of file names, in which that word is found, is given back to user. Along with that, the network time i.e. the time difference of request send time and response received time, and the processing time i.e. time to identify network delay, are also given.

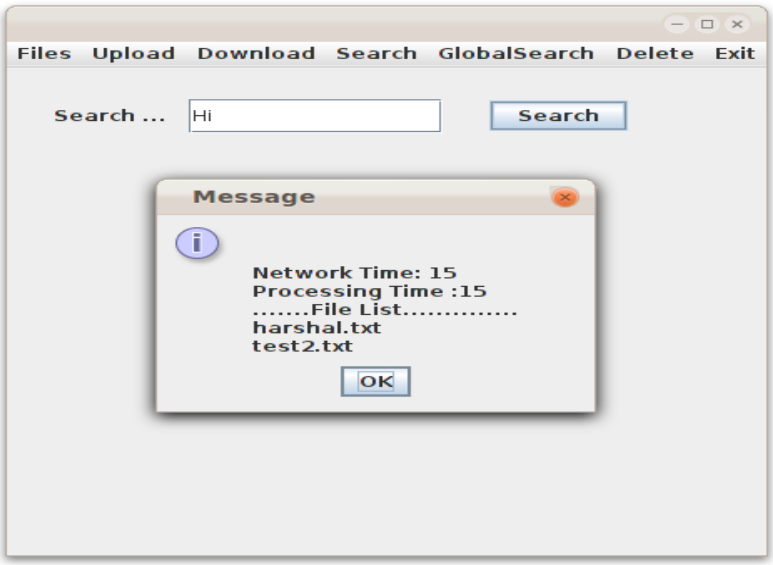


Figure 6 Search a word in all the files

5.2.4 Retrieval of text file

After a word is found in some files, if user wishes to retrieve the files containing that word, this facility is provided in 'Download File' option. If word is found in multiple files, user can select which file to download out of those files. When user presses the download button, contents of that file are displayed to the user. Thus, user gets his file/s containing particular word. User does not see any background retrieval or decryption process on his screen. Along with Downloaded file, we are giving Decryption time i.e. time required to convert entropy to encrypted text and time required to convert encrypted text to plaintext by AES algorithm, Network time and Processing time.

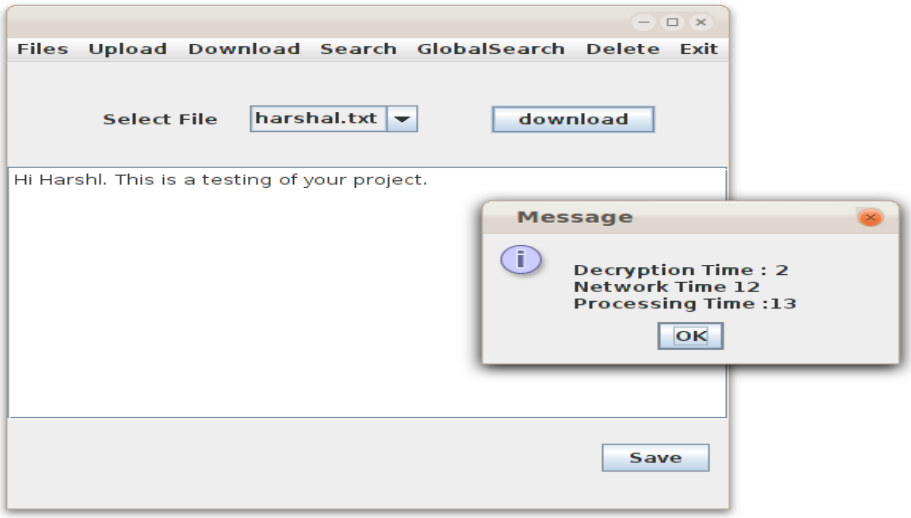


Figure 7 Download file from server

5.2.5 Delete the file from server

If a user or organization wants to delete its some data from server, this can be done through delete option.

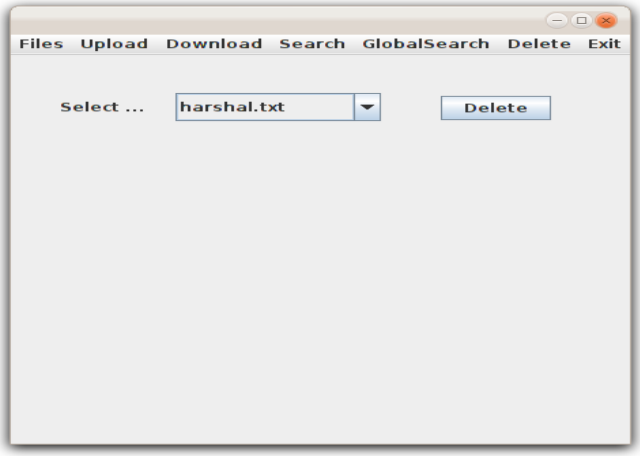


Figure 8 Delete file from server

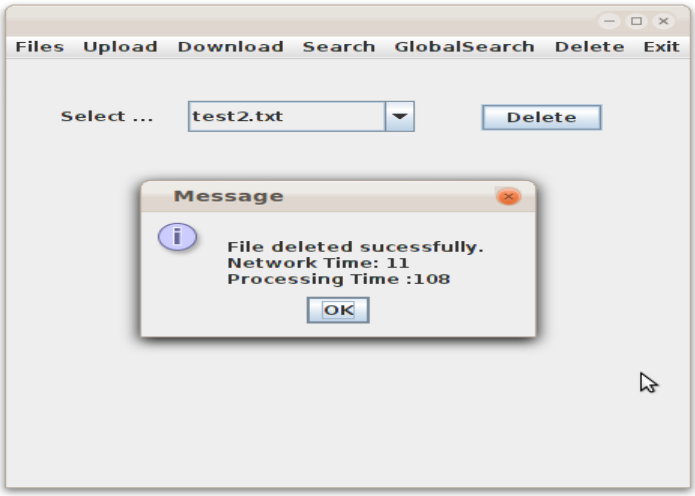


Figure 9 User screen after file deletion

After deletion of file named ‘harshal.txt’ the window is displayed to user along with list of his remaining files stored on the server. The network time and processing time are also given to user.

Chapter 6

Evaluation of System

The issues that were addressed at the start of design of this system were Obliviousness and Ease of Accessibility and Retrieval. Both the issues are being satisfied in this system. Obliviousness is achieved using AES + Base-64 encoding as well as creating unique id to each user.

Obliviousness:

A) Server Security :

When user connects with the server, his unique id is created automatically. When user sends his queries, they are searched in the database entries having that user id only. We are sending destination file along with each encrypted query we are sending to server. So, the searching is done in user's database only and not in any other files. So, user cannot access or retrieve other than his own data. Thus, data on server is secured from unauthorized access. Hence, server privacy is ensured.

B) User Privacy :

When user uploads his data, it is converted into AES 'word-by-word' and then its entropy is calculated using Base-64. This entropy value is sent to the server. Practical deciphering of AES +Base-64 is nearly impossible today. So, server will not be able to know anything about what user is uploading.

During retrieval, if the word is searched in all the files, the word is encrypted using different encryption keys of all the files. If there are n files of the user, then one word is encrypted in n different ways. And server receives n different queries, each searching for different encrypted word in different files. Therefore, server cannot predict or guess which word user is searching for. Also, as there is huge randomness created in querying phase, server cannot infer about user data or in

which file user is searching. Therefore, security and privacy of user data is ensured in this way.

Thus, our protocol ensures the user privacy and database security at the same time. Hence, Obliviousness is achieved.

Effectiveness:

Our system is effective in many ways.

- Both the client & server privacy is being ensured.
- Use of AES and unique id to each user + Base-64 encoding enhances security of the system and makes it difficult to make attack on system by any intruder.
- Practical deciphering of AES without encryption key is impossible today. So, it provides security better than most of the existing systems.

Performance:

In this system, we have measured the performance of various operations. While measuring the performance overhead we included encryption time, decryption time, network overheard and server processing time.

- Encryption Time: Encryption time is the time required by AES encryption module to perform encryption of plaintext data 'word-by-word' by using a key provided by client plus calculation of entropy.
- Decryption Time: Decryption time is the time required to convert entropy to encrypted text and time required to convert encrypted text to plaintext by AES algorithm.
- Network overhead and server processing time: This is calculated by measuring the time difference of request send time and response received time.
- Server processing time: We also give the server processing time separately from server to client, to identify network delay time.

All these time are in few milliseconds, as shown in snapshots and results. As shown above, the upload, search, download, delete options require time between 10 to few hundred milliseconds. Our system requires only a few milliseconds to encrypt and decrypt an entire file. Thus, it is fast and the same time provides better security to both the user and server.

For each operation i.e. uploading, retrieval, delete a simple user-friendly GUI is provided so that non-IT users can also use the application effectively and easily. Therefore, the other goal of system i.e. Ease of accessibility and fast Retrieval is achieved.

Transparency and Compatibility:

Our system provides great deal of transparency to the user.

- The conversion of text into entropy is transparent to the user. User sees only his plaintext and do not need to involve in encryption or entropy calculation. These procedures are done internally.
- Also, at the time of retrieval, user only has to enter text and his files are retrieved. At one click of user, his files will be deleted from the database. Thus, transparency is achieved along with obliviousness.

The system is highly compatible. We have used web-based https service for client-server communication. Due to this, this system is easy to migrate or deploy on outsourced database servers or even cloud servers. Web-based services are used in Cloud service providers to avail their service. So, this system can be easily deployed and expanded on these servers.

Chapter 7

Experimentation and Result Analysis

We have evaluated our system on Intel Core i5 2.58GHz CPU and 4GB RAM, running Ubuntu 12.04. We conducted experiments using a lot of text files and evaluated results from them. And our results show little performance overhead. The Performance Analyzer tool on client side gives the performance time for different operations such as Network time, Query processing time, Encryption/Decryption time.

We have uploaded hundreds of files on database server and calculated the required time for uploading, retrieval, deleting a file and searching a word in all the files. We already have shown the snapshots of various operations performed by user. The results show that few milliseconds are required for these operations, providing performance efficiency.

7.1 Results for Upload

In our system, each file is uploaded with a unique key provided by user. A text file takes only a few milliseconds to upload on database server in encrypted format. As shown in the below figure, each file is uploaded uniquely with the encryption key. Uploading time for file includes time for encryption, time for travel between network and time for processing i.e. uploading a file. These times are shown in figure 10.

A normal text file requires only 2-20 ms for encryption. It requires 10s of milliseconds for Network time. The Processing time is the time between file uploading is started (Submit Button is clicked) and file is successfully uploaded on server. Results show that a few kb file takes only few hundred milliseconds to be uploaded.

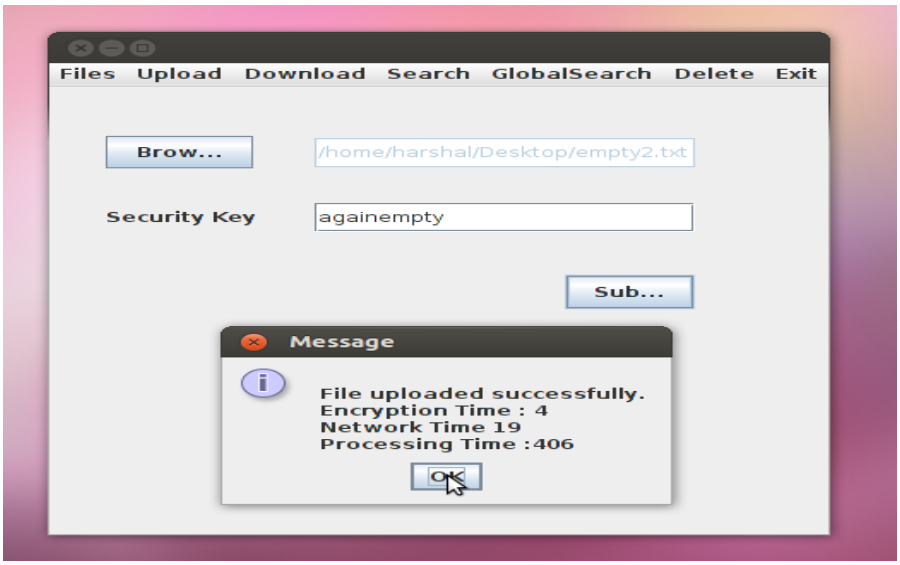


Figure 10 File Upload Time

7.2 Results for File Retrieval

File Retrieval from database server includes downloading of that file and then decryption of that file (Base-64 +AES) and then retrieving contents. File is first downloaded on user side in encrypted format as it was stored on server. Then, its contents are decrypted using the decryption key, which was provided by user and was maintained in index file. Our results show that downloading of file also takes only few milliseconds, which includes File Decryption time, Network time and Processing time.

Our system incurs little overhead and requires 2- 30 milliseconds to retrieve particular text file.

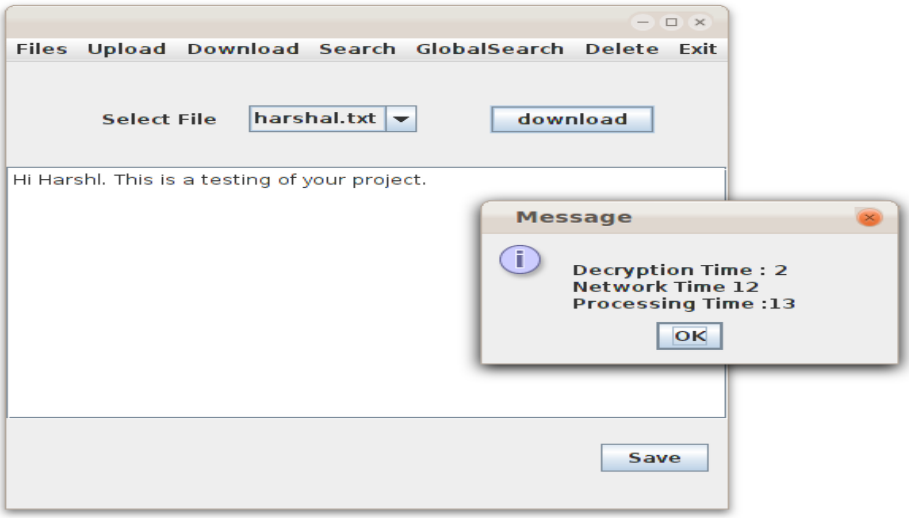


Figure 11 File Retrieval Time

7.3 Results for searching a word in all the files

The main aim of this system is to retrieve the text file containing some particular text or word. If user wants to search all the files containing particular word, server should reply with those required files. To achieve this, there is a necessity to search all the words in all the files, to determine whether that file contains all that word or not. If there are multiple files stored on the server, system should not incur unnecessary overhead in searching all the files.

We have experimented and tested our system by uploading hundreds of files on the server. We first uploaded hundreds of files on the server and then searched for files containing a word. And we calculated time required for each process. The results of the experiments are as shown below. :

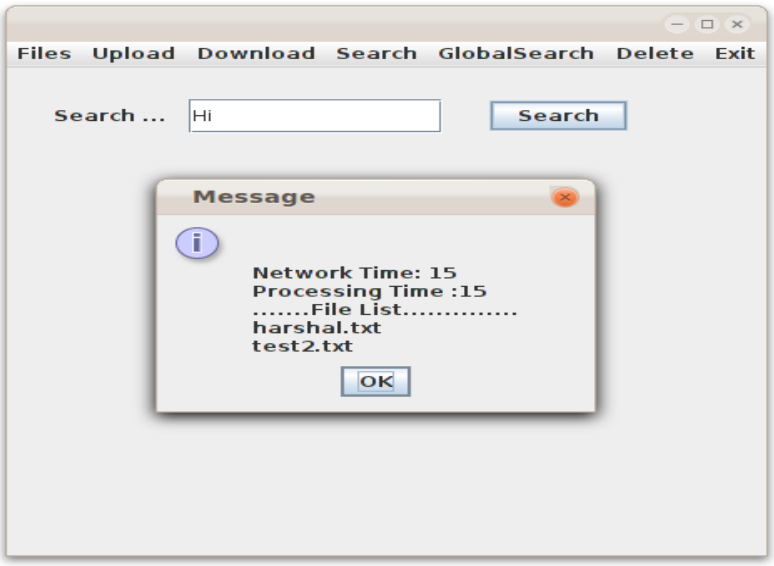


Figure 12 Searching time in all the files

The Network time is same in all the cases despite the no. of files.

The difference is in Processing time, which includes time between request sent and response received.

Table 1 Comparison between No. of Files and Time Required to Search in the Files

No. of Files	Time Required to search all the files (in milliseconds)
10	105 milliseconds.
20	210 ms.
50	450 ms.
100	820 ms.
200	1500 ms.
500	4448 ms.
1000	9077 ms ~= 9 seconds.

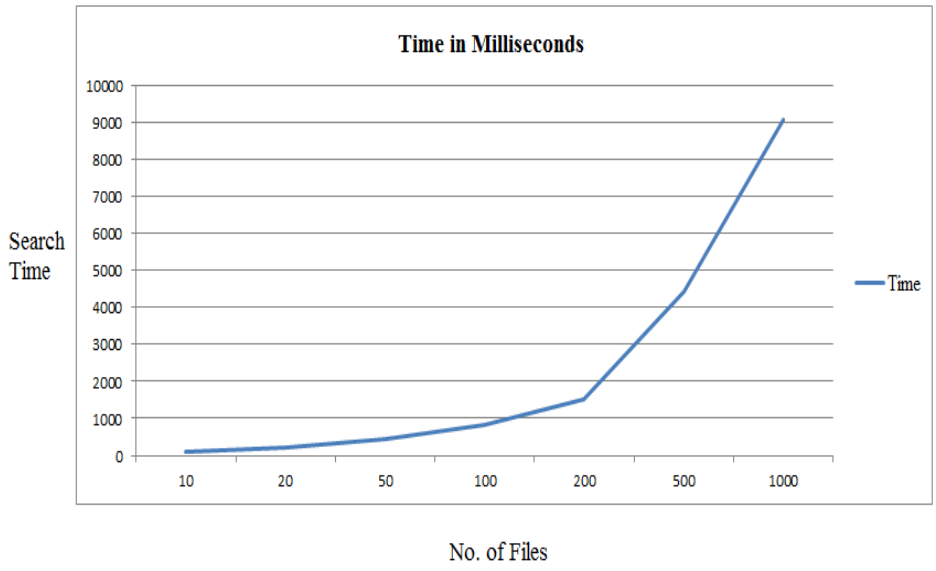


Figure 13 Line Graph for No. of Files

We can conclude that the text search and retrieval from thousand files can be done within 10 seconds. Thus, user can retrieve his required files from thousands of files within a few seconds, and that also with obliviousness.

We have also calculated different times required for various operations performed by user. These operations are given below. :

Table 2 Time Required for Different User Operations

Time	Upload	Download	Search	Delete
Network	19	15	15	14
Processing	220	50	20	80

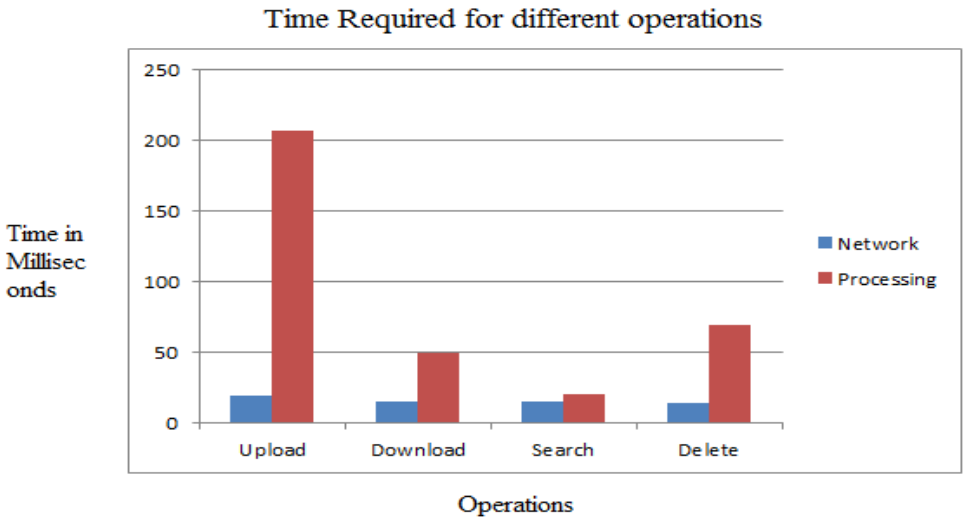


Figure 14 Bar Graph for Various Operations

Thus, at the same time we are providing user with the facility to upload his data, search his required data from untrusted server and provide privacy and security to both the data and user at user and server side. Doing this, we are also providing with retrieval efficiency to user and obliviousness to the system.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

In this work, we addressed the novel problem of oblivious information retrieval. We have experimented and evaluated the proposed system with two aspects in mind. One is to provide security and privacy to user and server at the same time during data retrieval, and other is the fast and efficient retrieval of information from untrusted server.

We have successfully implemented the system for uploading and retrieval of text data from untrusted server. At the same time we have achieved obliviousness at both the sides. Using AES and Base-64 encoding, the better security is provided to the user information. AES, Base-64 and Randomness in the queries with unique client id together provide obliviousness to text retrieval. The system is secure against unauthorized intruder and untrusted server and provides high level of multilayered security to both the user and the server.

Our protocol satisfies the necessities of privacy whereas maintaining affordable performance potency. At the same time, our system incurs little overhead and is efficient and fast in data retrieval, as well as provides user-friendliness and transparency to users. Experimentation shows that little time is required for searching the word in all the files. We have experimented system for many files and have shown text retrieval takes a little time.

Although the system has provided satisfactory results, we consider this as the first step to provide internet/cloud users, an Oblivious Data Retrieval System for their use and it opens avenues for further research and development in this.

8.2 Future Work

The longer term work in this domain lies in extending a similar application to different information varieties like multi-media information. The big size of the information in such situations makes it troublesome to enforce a good way of security measures that are desired by users; as well as it is difficult to retrieve that data quickly and efficiently. Coming up with economical oblivious retrieval protocol for such applications may be a promising new space.

The long-term aim of this system is to deploy this system on Cloud servers & provide fast, efficient & secure Database as a Service (DaaS).

Data Retrieval techniques over multiple files can also be parallelized using Map-Reduce or CUDA [10].

Appendix A

Paper Publication

Paper Published in ICCTCMECAT-2013

Title	Journal	Status
Survey on data retrieval techniques from the public databases	International Journal of Advances in Management, Technology & Engineering Sciences [ISSN 2249-7455]	Published.

Bibliography

[1] Shashank J, Kowshik P, Kannan Srinathan and C.V. Jawahar "*Private Content Based Data Retrieval*".

[2] Wentao Liu, "*Research on Cloud Computing Security Problem and Strategy*", IEEE 2012.

[3] C. Wang, Q. Wang, and K. Ren, "*Towards Secure and Effective Utilization over Encrypted Data*" 2011 31st International Conference on DCS Workshops.

[4] G.Aggarwal, Ashwin T. V. and Sugata Ghosal, "*A Retrieval System With Automatic Query Modification*" IEEE transactions on multimedia, vol. 4, no. 2, june 2002

[5] Y. Yang, H. Lu and J. Weng, "*Multi-User Private Keyword Search*" 2011 Third IEEE International Conference

[6] Michael O. Rabin, "*How to Exchange Secrets with Oblivious Transfer*". Harvard University Technical Report, TR-81,Aiken Computation Lab, Harvard University, 1981.

[7] Huafei Zhu and Feng Bao, "*Oblivious Keyword Search Protocols in the Public Database Model" IEEE Communications Society, publication in the ICC 2007 proceeding*

[8] Pruthvi Reddy Sabbu, Umadevi Ganugulay, Srinathan Kannan and Bruhadeshwar Bezawada, "*An Oblivious Retrieval Protocol*" 2011 Workshops of International Conference on Advanced Information Networking and Applications

[9] Wenjun Lu, Avinash L. Varna, and Min Wu, "*Security analysis for privacy preserving search of multimedia*", IEEE 2010.

[10] Zhenman Fang, Donglei Yang, Weihua Zhang, Haibo Chen and Binyu Zang, "*A Comprehensive Analysis and Parallelization of an Image Retrieval Algorithm"* 2011 IEEE.

[11] Wakaha Ogata, Kaoru Kurosawa, "*Oblivious Keyword Search*".

[12] Swaminathan, Y. Mao, G.-M. Su, H. Gou, A. Varna, S. He, M. Wu, and D. Oard "*Confidentiality-Preserving Rank-Ordered Search*" ACM October 29, 2007.

[13] Dawn Xiaodong Song, David Wagner, Adrian Perrig "*Practical Techniques for Searches on Encrypted Data*" IEEE 2000.

[14] Java platform, enterprise edition. [Online]. Available: http://en.wikipedia.org/wiki/Java_Platform, Enterprise Edition

[15] Peerapon Vateekul and Amon Rungsawang, "*DWORM: A Distributed Text Retrieval Prototype*", Intemational Symposium on Communications and Informatian Technologies ISCIT 2004.

[16] Youssef Gahi1, Mouhcine Guennoun, Zouhair Guennoun1, Khalil El-Khatib, "*Encrypted Processes for Oblivious Data Retrieval*", 6th International Conference on Internet Technology and Secured Transactions December 2011.

[17] P. Suau, F. Escolano, "*Entropy Estimation and Multi-Dimensional Scale Saliency*", International Conference on Pattern Recognition 2010.

[18] Yihun Alemu, Jong-bin Koh, Muhammed Ikram, Dong-Kyoo Kim, "*Image Retrieval in Multimedia Databases: A Survey*", Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing 2009.

[19] Eclipse. [Online]. Available: http://www.eclipse.org/

[20] http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf