**Secure loT data outsourcing with Aggregate Statistics and fine - Grained access control**

A Project Report submitted in partial fulfillment of the degree of the Bachelor of Technology in Computer Science and Engineering

By

M. LOKESH 19C41A05D9

N.KARTHEEK 19C41A05F2

K. SRAVAN 19C41A05F9

CH. GANESH 19C41A05E0

Under the Guidance of

Mr K. CHAITANYA KUMAR, Assistant Professor



**Department of Computer Science and Engineering**

**Jayamukhi institute of technological Sciences**

**Narsampet, warangal-506 332**

**(Affiliated to JNTUH, Accredited by NAAC ‘A’ Grade)**

**2023**

**Jayamukhi institute of technological Sciences**

**Narsampet, warangal – 506 332**

**(Affiliated to JNTUH, Accredited by NAAC ‘A’ Grade)**

**CERTIFICATE**



This is to certify that the Project Report entitled “ **Secure loT data outsourcing with Aggregate Statistics and fine - Grained access control** ” is A bona fide work of the students **M.LOKESH,N.KARTHEEK,K.SRAVAN,CH.GANESH** bearing Roll No.s **19C41A05D9,19C41A05F2,19C41A05F9,19C41A05E0** submitted in partial fulfillment of the requirements for the award of the degree of ***Bachelor of Technology*** in **Computer Science & Engineering** during the academic year **2022-23.**

**Guide Head of the Department**

**Principal**

**ACKNOWLEDGEMENT**

We express our gratitude and sincere thanks to **Prof. V. JANAKI ,** principal, Jayamukhi Institute of Technological Sciences for her constant motivation and support.

We express our gratitude and sincere thanks to **Dr. P. SRINIVAS RAO,** Head of the Department Of CSE, and Jayamukhi Institute of Technological Sciences for his constant motivation and support.

We are grateful to our Project Guide **Mr.K. CHAITANYA KUMAR**, Assistant Professor for his Valuable guidance and giving us the opportunity to do our work in a better manner.

We are also thankful to our **Management** for providing all the facilities in completing the project.

**ABSTRACT**

With the rapid development of the Internet of Things (IoT) and the dramatic increase of IoT devices, the desire to outsource huge amounts of IoT data to the cloud becomes more urgent than ever. In order to ensure the confidentiality, IoT data are usually encrypted before they are outsourced to the cloud, which will inevitably hinder the statistical analysis of them. Homomorphic encryption is an alternative to achieve the computation of encrypted data, but its inefficiency makes it not practical in the IoT environment. Another problem comes with the encryption is how to enable IoT data to be accessed by users who possess a certain set of attributes defined by data owners. In this paper, we propose a novel and practical IoT data outsourcing scheme based on Corrigan-Gibbs et al. ’s computation of aggregate statistics and the ciphertextpolicy attribute-based encryption (CP-ABE). It supports both secure aggregation and fine-grained access control of outsourced IoT data. Users only have to bear a small amount of computation in the process of data upload and recovery. Security analysis demonstrates that our scheme well protects the confidentiality of IoT data. A thorough and detailed performance comparison shows that our scheme enjoys a better performance on both the client side and the fog server side.

**CONTENT**

1. INTRODUCTION
2. LITERATURE SURVEY
3. SYSTEM DESIGN
4. System Architecture
5. Data flow diagram
6. UML diagram
7. USE CASE DIAGRAM
8. Class diagram
9. Sequence diagram
10. Activity diagram
11. INPUT AND OUTPUT DESIGN
12. SYSTEM ANALYSIS
13. PROPOSED SYSTEM
14. PROPOSED SYSTEM
15. SOFTWARE ENVIRONMENT
16. SYSTEM STUDY AND SYSTEM TESTING
17. SYSTEM REQUIREMENT
18. CONCLUSION
19. REFERENCES

I**NTRODUCTION**

INTERNET of Things (IoT) is developing rapidly and the use of IoT devices has dramatically increased in recent years. It was forecasted that the IoT market would grow from more than 15 billion devices in 2015 to more than 75 billion in 2025 [1]. These devices have the potential to improve the living standard of their users significantly through interactions with the physical and digital worlds [2]. For example, users with smart home and wearable devices can obtain seamless and customized services from digital housekeepers, doctors, and fitness instructors [3]. Managing a constant stream of data collected from a variety of devices is a significant burden for IoT users with limited storage and computing resources. The “pay-as-you-go” Cloud Computing

model is an efficient alternative to manage data for customers. Users can outsource a large amount of IoT data to the cloud and recover whenever they need it. However, since IoT embeds different kinds of sensors and other devices into a variety of things in our daily life, IoT data usually involves much private information about users [12]. It might be the heart rate of the user at a certain moment collected from the smart sphygmomanometer, user exercise data collected from the smart watch, and the like. In order to protect the security of the outsourced data, an intuitive way is to encrypt the data before outsourcing it. But there will be some new problems coming with encryption. The first challenge is how to perform aggregate statistical analysis on encrypted data as accurate as possible. For example, we may want to learn about our health condition in a certain period of time or whether our exercise has reached the average level of people in a certain area [10] [11]. Several attempts have been made to solve this problem. Sun et al. [12] use homomorphic encryption to encrypt the IoT data so that the service providers can process the needs of users without acquiring the plaintext data. However, the homomorphism technology is not mature currently. As we all know, its encryption and decryption are quite inefficient, and only a few homomorphism properties are supported by the privacy homomorphism. All these bottlenecks will hinder the widespread use of the Internet of Things. What’s more, the cloud service provider process all the data analysis, which will inevitably cause transmission latency and degraded service when traffic between IoT devices and the cloud becomes extraordinary huge. The most important is that the trusted third party is in charge of homomorphism encryption and decryption, meaning that it can obtain all the plaintext data. Such a strong security assumption is quite problematic. In addition to homomorphic encryption, there is no better solution yet.

The second problem is how to achieve precise access control of encrypted data. The data owner may want to define an access policy and enable users that satisfy the policy to access the corresponding data. The attribute-based encryption (ABE) is a promising approach to realize this. It enables the data owner to define access policy over a universe of attributes that the user needs to possess in order to decrypt the ciphertext and enforce it on the data. ABE has two variants, key-policy attribute-based encryption (KP-ABE) [7] and ciphertext-policy attribute-based encryption (CP-ABE) [8]. The latter turns out to be well suited for access control in IoT due to its expressiveness in describing access policy of ciphertext. Huang et al. [9] achieves secure data access control using exactly the ciphertext-policy attribute-based encryption (CP-ABE) in IoT data outsourcing. But they did not consider aggregation of these encrypted data, thus data analysis is impossible.

Besides, due to the extraordinary huge volume of traffic between IoT devices and the cloud, the centralized cloud computing systems will suffer from unbearable transmission latency and degraded service. Fortunately, fog computing is a promising technology to solve this problem. Fog computing extends the cloud computing paradigm to the edge of the network, and it is characterized by low latency and widespread geographical distribution [5] [6]. Fog computing is

actually a tool for cloud-based services that can be considered as an interface between the users and the cloud. In this paper, we also adopt fog as an auxiliary tool and propose a secure IoT data outsourcing scheme. As far as we know, our scheme is the first to achieve encrypted data aggregation and precise access control simultaneously in IoT data outsourcing. The main contributions of this paper are summarized as follows.

**LITERATURE SURVEY**

**1) PERM: Practical reputation-based blacklisting without TTPS**

**AUTHORS:**  M. H. Au and A. Kapadia

Some users may misbehave under the cover of anonymity by, e.g., defacing webpages on Wikipedia or posting vulgar comments on YouTube. To prevent such abuse, a few anonymous credential schemes have been proposed that revoke access for misbehaving users while maintaining their anonymity such that no trusted third party (TTP) is involved in the revocation process. Recently we proposed BLACR, a TTP-free scheme that supports `reputation-based blacklisting' --- the service provider can score users' anonymous sessions (e.g., good vs. inappropriate comments) and users with insufficient reputation are denied access.

The major drawback of BLACR is the linear computational overhead in the size of the reputation list, which allows it to support reputation for only a few thousand user sessions in practical settings. We propose PERM, a revocation-window-based scheme (misbehaviors must be caught within a window of time), which makes computation independent of the size of the reputation list. PERM thus supports millions of user sessions and makes reputation-based blacklisting practical for large-scale deployments.

**2) BLACR: TTP-free blacklistable anonymous credentials with reputation**

**AUTHORS:** M. H. Au, A. Kapadia, and W. Susilo

Anonymous authentication can give users the license to misbehave since there is no fear of retribution. As a deterrent, or means to revocation, various schemes for accountable anonymity feature some kind of (possibly distributed) trusted third party (TTP) with the power to identify or link misbehaving users. Recently, schemes such as BLAC and PEREA showed how anonymous revocation can be achieved without such TTPs—anonymous users can be revoked if they misbehave, and yet nobody can identify or link such users cryptographically. Despite being the state of the art in anonymous revocation, these schemes allow only a basic form of revocation amounting to ‘revoke anybody with d or more misbehaviors’ or ‘revoke anybody whose combined misbehavior score is too high’ (where misbehaviors are assigned a ‘severity’ score). We present BLACR, which significantly advances anonymous revocation in three ways: 1) It constitutes a first attempt to generalize reputation-based anonymous revocation, where negative or positive scores can be assigned to anonymous sessions across multiple categories. Servers can block users based on policies, which specify a boolean combination of reputations in these categories; 2) We present a weighted extension, which allows the total severity score to ramp up for multiple misbehaviors by the same user; and, 3) We make a significant improvement in authentication times through a technique we call express lane authentication, which makes reputation-based anonymous revocation practical.

**3) Constant-size dynamic *k*-TAA**

**AUTHORS:** M. H. Au, W. Susilo, and Y. Mu

Dynamic k-times anonymous authentication (k-TAA) schemes allow members of a group to be authenticated anonymously by application providers for a bounded number of times, where application providers can independently and dynamically grant or revoke access right to members in their own group. In this paper, we construct a dynamic k-TAA scheme with space and time complexities of O(log(k)) and a variant, in which the authentication protocol only requires constant time and space complexities at the cost of O(k) -sized public key. We also describe some tradeoff issues between different system characteristics. We detail all the zero-knowledge proof-of-knowledge protocols involved and show that our construction is secure in the random oracle model under the q-strong Diffie–Hellman assumption and q-decisional Diffie–Hellman inversion assumption. We provide a proof-of-concept implementation, experiment on its performance, and show that our scheme is practical.

**4) A secure cloud computing based framework for big data information management of smart grid**

**AUTHORS:** J. Baek, Q. H. Vu, J. K. Liu, X. Huang, and Y. Xiang

Smart grid is a technological innovation that improves efficiency, reliability, economics, and sustainability of electricity services. It plays a crucial role in modern energy infrastructure. The main challenges of smart grids, however, are how to manage different types of front-end intelligent devices such as power assets and smart meters efficiently; and how to process a huge amount of data received from these devices. Cloud computing, a technology that provides computational resources on demands, is a good candidate to address these challenges since it has several good properties such as energy saving, cost saving, agility, scalability, and flexibility. In this paper, we propose a secure cloud computing based framework for big data information management in smart grids, which we call “Smart-Frame.” The main idea of our framework is to build a hierarchical structure of cloud computing centers to provide different types of computing services for information management and big data analysis. In addition to this structural framework, we present a security solution based on identity-based encryption, signature and proxy re-encryption to address critical security issues of the proposed framework.

**5) Ciphertext-policy attribute based encryption**

**AUTHORS:** J. Bethencourt, A. Sahai, and B. Waters

In several distributed systems a user should only be able to access data if a user posses a certain set of credentials or attributes. Currently, the only method for enforcing such policies is to employ a trusted server to store the data and mediate access control. However, if any server storing the data is compromised, then the confidentiality of the data will be compromised. In this paper we present a system for realizing complex access control on encrypted data that we call Ciphertext-Policy Attribute-Based Encryption. By using our techniques encrypted data can be kept confidential even if the storage server is untrusted; moreover, our methods are secure against collusion attacks. Previous AttributeBased Encryption systems used attributes to describe the encrypted data and built policies into user’s keys; while in our system attributes are used to describe a user’s credentials, and a party encrypting data determines a policy for who can decrypt. Thus, our methods are conceptually closer to traditional access control methods such as Role-Based Access Control (RBAC). In addition, we provide an implementation of our system and give performance measurement

**6)Theoretical modelling of fog computing: a green computing paradigm to support IoT applications**

**AUTHORS:** S. Sarkar and S. Misra

In this study, the authors focus on theoretical modelling of the *fog computing* architecture and compare its performance with the traditional cloud computing model. Existing research works on fog computing have primarily focused on the principles and concepts of fog computing and its significance in the context of *internet of things* (*IoT*). This work, one of the first attempts in its domain, proposes a mathematical formulation for this new computational paradigm by defining its individual components and presents a comparative study with cloud computing in terms of service latency and energy consumption. From the performance analysis, the work establishes fog computing, in collaboration with the traditional cloud computing platform, as an efficient green computing platform to support the demands of the next generation IoT applications. Results show that for a scenario where 25% of the IoT applications demand real-time, low-latency services, the mean energy expenditure in fog computing is 40.48% less than the conventional cloud computing model.

**7)The Quest for Privacy in the Internet of Things**

**AUTHORS:** P. Porambage, M. Ylianttila, C. Schmitt

The Internet of Things (IoT) is the current evolutionary paradigm of networking and the key driving force toward a smart world. Although privacy in the IoT is highly regarded to ensure the protection of users and personal information from the perspective of individual or cooperative users, it's insufficiently studied. As members of the always-connected paradigm of the massive IoT world, people can scarcely control the disclosure of their personal information. The biggest challenge is to allow users to experience the best utilization of IoT-based products and services with the fewest privacy threats and failures. This article provides a holistic view of the challenges of and issues related to preserving IoT privacy, as well as the existing solutions. Privacy by design (PbD) is identified as the key solution for many IoT privacy issues. The article also discusses hot topics in IoT privacy and future research directions.

**8)Lightweight RFID Protocol for Medical Privacy Protection in IoT**

**AUTHORS:** K. Fan, W. Jiang, H. Li, and Y. Yang

Traditional medical privacy data are at a serious risk of disclosure, and many related cases have occurred over the years. For example, personal medical privacy data can be easily leaked to insurance companies, which not only compromises the privacy of individuals, but also hinders the healthy development of the medical industry. With the continuous improvement of cloud computing and big data technologies, the Internet of Things technology has been rapidly developed. Radio frequency identification (RFID) is one of the core technologies of the Internet of Things. The application of the RFID system to the medical system can effectively solve this problem of medical privacy. RFID tags in the system can collect useful information and conduct data exchange and processing with a back-end server through the reader. The whole process of information interaction is mainly in the form of ciphertext. In the context of the Internet of Things, the paper presents a lightweight RFID medical privacy protection scheme. The scheme ensures security privacy of the collected data via secure authentication. The security analysis and evaluation of the scheme indicate that the protocol can effectively prevent the risk of medical privacy data being easily leaked.

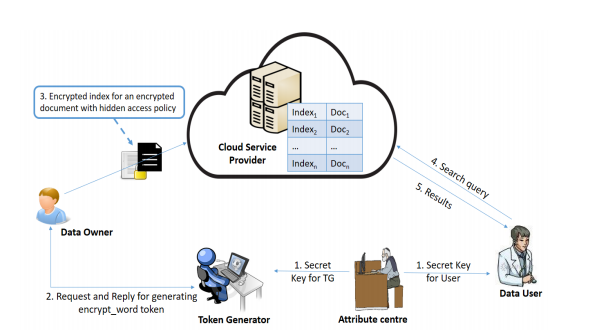
**9)A Medical Healthcare System for Privacy Protection Based on IoT**

**AUTHORS:** T. Gong, H. Huang, P. Li, K. Zhang, and H. Jiang

China's medical healthcare system has witnessed a great evolution from clinical normalization phase to pervasive informatization phase. Unquestionably, emergence of IoT (Internet of things) technology contributes to the development of medical healthcare informatization. Yet, it is widely researched and accepted that privacy protection has become a bottleneck of smart medical healthcare nowadays. Consequently, how to protect the privacy of users' information in a smart healthcare system remains a pending problem. Under this background, this paper analyses the main problems in current smart healthcare system. Furthermore, a lightweight private homomorphism algorithm and an encryption algorithm improved from DES are designed when considering the characteristic of IoT and privacy protection. Finally, based on the above, we designed and completed a prototype system based on both software and hardware.

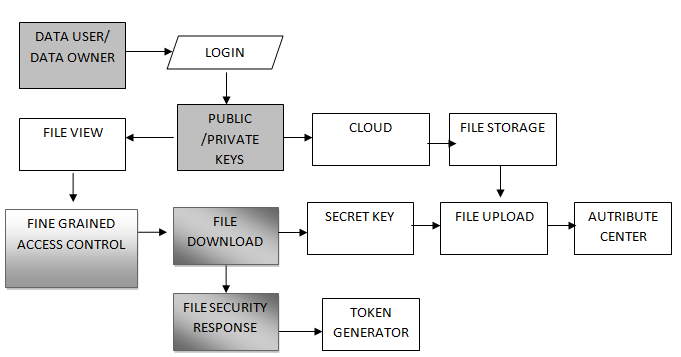
**SYSTEM DESIGN**

**SYSTEM ARCHITECTURE:**



**DATA FLOW DIAGRAM:**

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

****

**UML DIAGRAMS**

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

**USE CASE DIAGRAM:**

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

OWNER

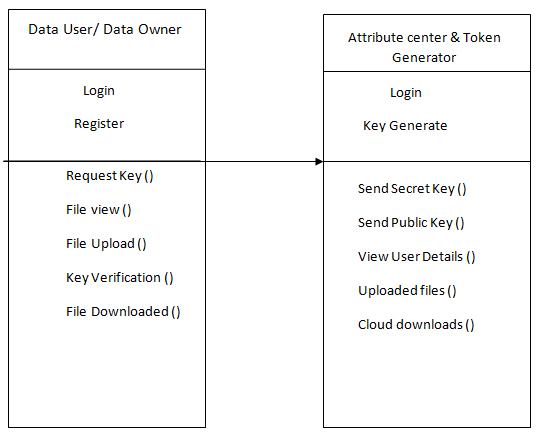
CLOUD

DATA USER

OR

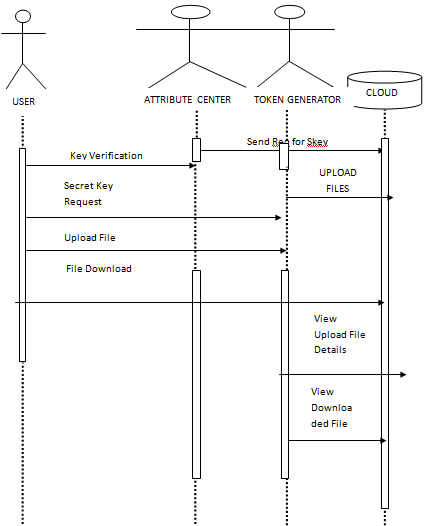
**CLASS DIAGRAM:**

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



**SEQUENCE DIAGRAM:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

****

**ACTIVITY DIAGRAM:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

FILE VIEW

ATTRIBUTE Center Req

REGISTRATION

DATA USER

Start

LOGIN

TOKEN Generator Req

FILE UPLOAD

DATA OWNER

GET SECRET KEY

Accept

NO

KEYWORD SRCH

LOGIN

FILE DOWNLODED

**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.

**OBJECTIVES**

1.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.

2. 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. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3.When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user

will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

**OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. 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.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2.Select methods for presenting information.

3.Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

* Convey information about past activities, current status or projections of the
* Future.
* Signal important events, opportunities, problems, or warnings.
* Trigger an action.
* Confirm an action.

**SYSTEM ANALYSIS**

**EXISTING SYSTEM:**

* A general approach to protect the data confidentiality is to encrypt the data before outsourcing.
* Searchable encryption schemes enable the client to store the encrypted data to the cloud and execute keyword search over ciphertext domain. So far, abundant works have been proposed under different threat models to achieve various search functionality, such as single keyword search, similarity search, multi-keyword boolean search, ranked search, multi-keyword ranked search, etc. Among them, multi-keyword ranked search achieves more and more attention for its practical applicability. Recently, some *dynamic* schemes have been proposed to support inserting and deleting operations on document collection. These are significant works as it is highly possible that the data owners need to update their data on the cloud server.

**DISADVANTAGES OF EXISTING SYSTEM:**

* Huge cost in terms of data usability. For example, the existing techniques on keyword-based information retrieval, which are widely used on the plaintext data, cannot be directly applied on the encrypted data. Downloading all the data from the cloud and decrypt locally is obviously impractical.
* Existing System methods not practical due to their high computational overhead for both the cloud sever and user.

**PROPOSED SYSTEM:**

* Suppose that there is a data user Alice, a data sender Bob, and an honest-but-curious server. When Bob wants to send some data to Alice, Bob should first obtain the pre-tag of the keyword from Alice in a secure way. Such communication channels between the senders and user are private in this phase.
* Then, Bob encrypts the data, generates indexes with pre-tags and sends them together to the server. When Alice wants to search for the required encrypted data, she could generate a trapdoor with an expression and send the trapdoor to the server.
* Once the server receives the search query with the index, it searches on the indexes to obtain matched encrypted data and sends them to Alice.
* All people, including the data user and data senders, could send the encrypted data and indexes to the server, but only the user can generate the encrypted keyword.
* This is a multi-sender/one-user system. Note that Alice does not search with a single keyword. She could search with any n keywords of a keyword set. At the same time, the server cannot learn anything about the keywords because the proposed scheme only sends the structure of the expression tree together with the index to the server.

**ADVANTAGES OF PROPOSED SYSTEM:**

* An innovative encryption scheme that supports the user to customize the relevancy of the keywords and the server to obtain the search results with the relevancy by taking one operation.
* A new definition of the semantic security model that defends against adaptive chosen keyword attacks. Under the defined security model, the proposed scheme can also defend against the offline keyword guessing attack.
* A performance analysis that contains comparisons with other schemes, the implementation of the proposed scheme and the evaluation of its computational overhead and the storage overhead.

**Software Environment**

**Java Technology**

Java technology is both a programming language and a platform.

**The Java Programming Language**

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

* + - Simple
    - Architecture neutral
    - Object oriented
    - Portable
    - Distributed
    - High performance
    - Interpreted
    - Multithreaded
    - Robust
    - Dynamic
    - Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Java byte codes —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.



You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it’s a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.



### The Java Platform

A platform is the hardware or software environment in which a program runs. We’ve already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it’s a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

* The Java Virtual Machine (Java VM)
* The Java Application Programming Interface (Java API)

You’ve already been introduced to the Java VM. It’s the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as packages. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that’s running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.



Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

## What Can Java Technology Do?

The most common types of programs written in the Java programming language are applets and applications. If you’ve surfed the Web, you’re probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a server serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a servlet. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

* **The essentials**: Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
* **Applets**: The set of conventions used by applets.
* **Networking**: URLs, TCP (Transmission Control Protocol), UDP (User Data gram Protocol) sockets, and IP (Internet Protocol) addresses.
* **Internationalization**: Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
* **Security**: Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
* **Software components**: Known as JavaBeansTM, can plug into existing component architectures.
* **Object serialization**: Allows lightweight persistence and communication via Remote Method Invocation (RMI).
* **Java Database Connectivity (JDBCTM)**: Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.



## How Will Java Technology Change My Life?

We can’t promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages. We believe that Java technology will help you do the following:

* **Get started quickly**: Although the Java programming language is a powerful object-oriented language, it’s easy to learn, especially for programmers already familiar with C or C++.
* **Write less code**: Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
* **Write better code**: The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people’s tested code and introduce fewer bugs.
* **Develop programs more quickly**: Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.
* **Avoid platform dependencies with 100% Pure Java**: You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure JavaTM Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.
* **Write once, run anywhere**: Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
* **Distribute software more easily**: You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded “on the fly,” without recompiling the entire program.

### ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a *de facto* standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE. There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources.

From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn’t change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn’t as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

**JDBC**

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of “plug-in” database connectivity modules, or *drivers*. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC’s framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

### JDBC Goals

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

1. **SQL Level API**

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to “generate” JDBC code and to hide many of JDBC’s complexities from the end user.

1. **SQL Conformance**

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

1. **JDBC must be implemental on top of common database interfaces**   
   The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.
2. **Provide a Java interface that is consistent with the rest of the Java system**

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

1. **Keep it simple**

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

1. **Use strong, static typing wherever possible**

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

1. **Keep the common cases simple**

Because more often than not, the usual SQL calls used by the programmer are simple SELECT’s, INSERT’s, DELETE’s and UPDATE’s, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible.

Finally we decided to proceed the implementation using Java Networking.

And for dynamically updating the cache table we go for MS Access database.

Java ha two things: a programming language and a platform.

Java is a high-level programming language that is all of the following

Simple Architecture-neutral

Object-oriented Portable

Distributed High-performance

Interpreted multithreaded

Robust Dynamic

Secure

Java is also unusual in that each Java program is both compiled and interpreted. With a compile you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer.

Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.

**Java Program**

**Compilers**

**Interpreter**

**My Program**

You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it’s a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

Java byte codes help make “write once, run anywhere” possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

## Networking

### TCP/IP stack

The TCP/IP stack is shorter than the OSI one:



TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

**IP datagram’s**

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

**UDP**

UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model - see later.

**TCP**

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

### Internet addresses

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32 bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

### Network address

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16 bit network addressing. Class C uses 24 bit network addressing and class D uses all 32.

**Subnet address**

Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

### Host address

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

### Total address



The 32 bit address is usually written as 4 integers separated by dots.

### Port addresses

A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

**Sockets**

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call socket. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with Read File and Write File functions.

#include <sys/types.h>

#include <sys/socket.h>

int socket(int family, int type, int protocol);

Here "family" will be AF\_INET for IP communications, protocol will be zero, and type will depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

**JFree Chart**

JFreeChart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart's extensive feature set includes:

A consistent and well-documented API, supporting a wide range of chart types;

A flexible design that is easy to extend, and targets both server-side and client-side applications;

Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

JFreeChart is "open source" or, more specifically, [free software](http://www.gnu.org/philosophy/free-sw.html). It is distributed under the terms of the [GNU Lesser General Public Licence](http://www.gnu.org/licenses/lgpl.html) (LGPL), which permits use in proprietary applications.

## 1. Map Visualizations

Charts showing values that relate to geographical areas. Some examples include: (a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world. The tasks in this project include:

Sourcing freely redistributable vector outlines for the countries of the world, states/provinces in particular countries (USA in particular, but also other areas);

Creating an appropriate dataset interface (plus default implementation), a rendered, and integrating this with the existing XYPlot class in JFreeChart;

Testing, documenting, testing some more, documenting some more.

## 2. Time Series Chart Interactivity

Implement a new (to JFreeChart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data, with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

***3*. *Dashboards***

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of JFreeChart chart types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

1. **Property Editors**

The property editor mechanism in JFreeChart only handles a small subset of the properties that can be set for charts. Extend (or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

**J2ME (Java 2 Micro edition):-**

Sun Microsystems defines J2ME as "a highly optimized Java run-time environment targeting a wide range of consumer products, including pagers, cellular phones, screen-phones, digital set-top boxes and car navigation systems." Announced in June 1999 at the JavaOne Developer Conference, J2ME brings the cross-platform functionality of the Java language to smaller devices, allowing mobile wireless devices to share applications. With J2ME, Sun has adapted the Java platform for consumer products that incorporate or are based on small computing devices.

**1. General J2ME architecture**



J2ME uses configurations and profiles to customize the Java Runtime Environment (JRE). As a complete JRE, J2ME is comprised of a configuration, which determines the JVM used, and a profile, which defines the application by adding domain-specific classes. The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. We'll discuss configurations in detail in the The profile defines the application; specifically, it adds domain-specific classes to the J2ME configuration to define certain uses for devices. We'll cover profiles in depth in the The following graphic depicts the relationship between the different virtual machines, configurations, and profiles. It also draws a parallel with the J2SE API and its Java virtual machine. While the J2SE virtual machine is generally referred to as a JVM, the J2ME virtual machines, KVM and CVM, are subsets of JVM. Both KVM and CVM can be thought of as a kind of Java virtual machine -- it's just that they are shrunken versions of the J2SE JVM and are specific to J2ME.

**2.Developing J2ME applications**

Introduction In this section, we will go over some considerations you need to keep in mind when developing applications for smaller devices. We'll take a look at the way the compiler is invoked when using J2SE to compile J2ME applications. Finally, we'll explore packaging and deployment and the role preverification plays in this process.

**3.Design considerations for small devices**

Developing applications for small devices requires you to keep certain strategies in mind during the design phase. It is best to strategically design an application for a small device before you begin coding. Correcting the code because you failed to consider all of the "gotchas" before developing the application can be a painful process. Here are some design strategies to consider:

\* Keep it simple. Remove unnecessary features, possibly making those features a separate, secondary application.

\* Smaller is better. This consideration should be a "no brainer" for all developers. Smaller applications use less memory on the device and require shorter installation times. Consider packaging your Java applications as compressed Java Archive (jar) files.

\* Minimize run-time memory use. To minimize the amount of memory used at run time, use scalar types in place of object types. Also, do not depend on the garbage collector. You should manage the memory efficiently yourself by setting object references to null when you are finished with them. Another way to reduce run-time memory is to use lazy instantiation, only allocating objects on an as-needed basis. Other ways of reducing overall and peak memory use on small devices are to release resources quickly, reuse objects, and avoid exceptions.

**4.Configurations overview**

The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. Currently, two configurations exist for J2ME, though others may be defined in the future:

\* **Connected Limited Device Configuration (CLDC)** is used specifically with the KVM for 16-bit or 32-bit devices with limited amounts of memory. This is the configuration (and the virtual machine) used for developing small J2ME applications. Its size limitations make CLDC more interesting and challenging (from a development point of view) than CDC. CLDC is also the configuration that we will use for developing our drawing tool application. An example of a small wireless device running small applications is a Palm hand-held computer.

\* **Connected Device Configuration (CDC)** is used with the C virtual machine (CVM) and is used for 32-bit architectures requiring more than 2 MB of memory. An example of such a device is a Net TV box.

**5.J2ME profiles**

**What is a J2ME profile?**

As we mentioned earlier in this tutorial, a profile defines the type of device supported. The Mobile Information Device Profile (MIDP), for example, defines classes for cellular phones. It adds domain-specific classes to the J2ME configuration to define uses for similar devices. Two profiles have been defined for J2ME and are built upon CLDC: KJava and MIDP. Both KJava and MIDP are associated with CLDC and smaller devices. Profiles are built on top of configurations. Because profiles are specific to the size of the device (amount of memory) on which an application runs, certain profiles are associated with certain configurations.

A skeleton profile upon which you can create your own profile, the Foundation Profile, is available for CDC.

**Profile 1: KJava**

KJava is Sun's proprietary profile and contains the KJava API. The KJava profile is built on top of the CLDC configuration. The KJava virtual machine, KVM, accepts the same byte codes and class file format as the classic J2SE virtual machine. KJava contains a Sun-specific API that runs on the Palm OS. The KJava API has a great deal in common with the J2SE Abstract Windowing Toolkit (AWT). However, because it is not a standard J2ME package, its main package is com.sun.kjava. We'll learn more about the KJava API later in this tutorial when we develop some sample applications.

**Profile 2: MIDP**

MIDP is geared toward mobile devices such as cellular phones and pagers. The MIDP, like KJava, is built upon CLDC and provides a standard run-time environment that allows new applications and services to be deployed dynamically on end user devices. MIDP is a common, industry-standard profile for mobile devices that is not dependent on a specific vendor. It is a complete and supported foundation for mobile application

development. MIDP contains the following packages, the first three of which are core CLDC packages, plus three MIDP-specific packages.

\* java.lang

\* java.io

\* java.util

\* javax.microedition.io

\* javax.microedition.lcdui

\* javax.microedition.midlet

**SYSTEM STUDY**

**FEASIBILITY STUDY**

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

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**ECONOMICAL FEASIBILITY**

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

### TECHNICAL FEASIBILITY

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

**SOCIAL FEASIBILITY**

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

**SYSTEM REQUIREMENTS**

# H/W System Configuration:-

# Processor - I3/Intel Processor

* RAM - 4GB (min)
* Hard Disk - 160GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse
* Monitor - SVGA

**S/W System Configuration:-**

* Operating System : Windows 7/8/10
* Web Server : Tomcat 7.0
* Front End : HTML, CSS,JavaScript,JSP

Database : My SQL 6.0

IDE : NETBEANS 8.1

* Database Connectivity : JDBC

### SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**TYPES OF TESTS**

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

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

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

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**System Test**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

***Black Box Testing***

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**6.1 Unit Testing:**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

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

**Test objectives**

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

**Features to be tested**

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

# 6.2 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**6.3 Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**CONCLUSION**

In this paper, we propose a secure IoT data outsourcing scheme, which can support both real-time aggregate statistical analysis and fine-grained access control of outsourced IoT data. By utilizing Corrigan-Gibbs et al.’s computation of aggregate statistics - Prio and Beaver’s multi-party computing (MPC) protocol, fog servers can perform aggregation such as addition, multiplication, and variance on the IoT data uploaded by the data owner, without knowing the original data. Ciphertext-policy attribute-based encryption (CP-ABE) helps us realize fine-grained access control, only allowing the user whose attributes set satisfies the access policy to recover the corresponding data. The security analysis shows that our scheme ensures correctness and data confidentiality. The extensive performance analysis and experiment demonstrate the efficiency of our scheme, meaning it is suitable for the resource-constrained IoT devices such as the MI phone used in our experiment, thus can be further used in real-time health monitoring and many other IoT environments. For our future work, we will try to seek ways to protect the confidentiality of results computed by fog servers, which is not considered in this scheme. Another problem is that the storage overhead in our scheme increases with the number of shares that a data is divided into, which is also the focus of our further research.

**REFERENCES**

[1] On The Edge: Solving The Challenges Of Edge Computing In The Era Of IoT (2018), [Online]. Available: <https://data-economy.com/on-the-edgesolving-> the-challenges-of-edge-computing-in-the-era-of-iot/.

[2] C. Li and B. Palanisamy, Privacy in Internet of Things: from Principles to Technologies[J]. IEEE Internet of Things Journal, 2019, 6(1): 488-505.

[3] B. D.Weinberg, G. R. Milne, Y. G. Andonova, and F. M. Hajjat, Internet of Things: Convenience vs. privacy and secrecy[J]. Business Horizons, 2015, 58(6): 615´lC624.

[4] L. M. Vaquero and L. Rodero-Merino, Finding your Way in the Fog: Towards a Comprehensive Definition of Fog Computing[C]. ACM SIGCOMM Computer Communication Review, 2014, 44(5): 27-32.

[5] M. Chiang and T. Zhang, Fog and IoT: An Overview of Research Opportunities[ J]. IEEE Internet of Things Journal, 2016, 3(6): 854-864.

[6] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, Fog Computing and Its Role in the Internet of Things[C]. MCC’12 Proceedings of the first edition of the MCC workshop on Mobile cloud computing, Helsinki, Finland, August 17, 2012: 13-16.

[7] V. Goyal, O. Pandey, A. Sahai, and B. Waters, Attribute-based encryption for fine-grained access control of encrypted data[C]. Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, 2006: 89-98.

[8] J. Bethencourt, A. Sahai, and B.Waters, Ciphertext-Policy Attribute-Based Encryption[C]. 2007 IEEE Symposium on Security and Privacy (SP’07), Berkeley, CA, USA, 2007: 321-334.

[9] Q. Huang, Y. Yang, and L. Wang, Secure Data Access Control With Ciphertext Update and Computation Outsourcing in Fog Computing for Internet of Things[J]. IEEE Access, 2017, 5: 12941 - 12950.

[10] H. Abie and I. Balasingham, Risk-Based Adaptive Security for Smart IoT in eHealth[C].ACM 7th International Conference on Body Area Networks - Oslo, Norway, 2012: 269-275.

[11] C. Doukas, I. Maglogiannis, V. Koufi, F. Malamateniou, and G. Vassilacopoulos, Enabling Data Protection through PKI encryption in IoT m- Health Devices[C]. Proceedings of the 2012 IEEE 12th International Conference on Bioinformatics & Bioengineering (BIBE), Larnaca, Cyprus, 11-13 November 2012.

[12] G. Sun, S. Huang, W. Bao, Y. Yang, and Z. Wang, A privacy protection policy combined with privacy homomorphism in the Internet of Things[C]. International Conference on Computer Communication & Networks. IEEE, 2014.

[13] N. K. Giang, M. Blackstock, R. Lea, and V. C. M. Leung, Developing IoT applications in the Fog: A Distributed Dataflow approach[C]. IEEE 2015 5th International Conference on the Internet of Things (IOT), Seoul, South Korea, 2015: 155-162.

[14] S. Sarkar and S. Misra, Theoretical modelling of fog computing: a green computing paradigm to support IoT applications[J]. Iet Networks, 2016, 5(2): 23-29.

[15] B. Farahani, F. Firouzi, V. Chang, M. Badaroglu, N. Constant, and K. Mankodiya, Towards fog-driven IoT eHealth: Promises and challenges of IoT in medicine and healthcare[J]. Future Generation Computer Systems, 2017, 78(2): 659-676.

[16] S. M. A. Oteafy and H. S. Hassanein, IoT in the Fog: A Roadmap for Data-Centric IoT Development[J]. IEEE Communications Magazine, 2018, 56(3): 157-163.

[17] S. Chen, H. Xu, D. Liu, B. Hu, and H.Wang, A Vision of IoT: Applications, Challenges, and OpportunitiesWith China Perspective[J]. IEEE Internet of Things Journal, 2014, 1(4): 349-359.

[18] P. Porambage, M. Ylianttila, C. Schmitt, P. Kumar, A. Gurtov, and A. V. Vasilakos, The Quest for Privacy in the Internet of Things[J]. IEEE Cloud Computing, 2016, 3(2): 36-45.

[19] L. Chen, S. Thombre, K. Jarvinen, E. S. Lohan, A. Al\_en-Savikko, H. Leppakoski, M. Z. H. Bhuiyan, S. Bu-Pasha, G. N. Ferrara, S. Honkala, J. Lindqvist, L. Ruotsalainen, P. Korpisaari, and H. Kuusniemi, Robustness, Security and Privacy in Location-Based Services for Future IoT: A Survey[J]. IEEE Access, 2017, 5: 8956-8977.

[20] K. Fan, W. Jiang, H. Li, and Y. Yang, Lightweight RFID Protocol for Medical Privacy Protection in IoT[J]. IEEE Transactions on Industrial Informatics, 2018, 14(4): 1656-1665.

[21] T. Gong, H. Huang, P. Li, K. Zhang, and H. Jiang, A Medical Healthcare System for Privacy Protection Based on IoT[C]. 2015 Seventh International Symposium on Parallel Architectures, Algorithms and Programming (PAAP), Nanjing, China, 2015.