

**A Project Report**  
**on**  
**SEMANTIC-AWARE SEARCHING OVER ENCRYPTED**  
**DATA FOR CLOUD COMPUTING**

**Submitted in partial fulfillment of the requirements for the award of the degree**  
**of**

**BACHELOR OF TECHNOLOGY**  
**in**  
**INFORMATION TECHNOLOGY**  
**by**

**B.Manjula(16WH1A1208)**

**K.Ramya(16WH1A1221)**

**B.Sravya(16WH1A1256)**

**Under the esteemed guidance of**

**Ms.P.S.Latha Kalyampudi**  
**Assistant Professor**



**Department of Information Technology**  
**BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN**  
(Accredited by NBA & NAAC 'A' Grade, Approved by AICTE, and  
Affiliated to JNTUH, Hyderabad)  
**Bachupally, Hyderabad – 500090**  
**April 2020**

## **DECLARATION**

We hereby declare that the work presented in this project entitled **“SEMANTIC-AWARE SEARCHING OVER ENCRYPETD DATA FOR CLOUD COMPUTING”** submitted towards completion of the major project in IV-B.Tech of Information Technology at BVRIT HYDERABAD College of Engineering for Women, Hyderabad is an authentic record of our original work carried out under the esteem guidance of Ms. P.S. Latha Kalyampudi, Assistant Professor, department of IT.

**B.Manjula**  
**(16WH1A1208)**

**K.Ramya**  
**(16WH1A1221)**

**B.Sravya**  
**(16WH1A1256)**

# **BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN**

(Accredited by NBA & NAAC 'A' Grade, Approved by AICTE, and Affiliated to JNTUH, Hyderabad)

**Bachupally, Hyderabad – 500090**

## **Department of Information Technology**



### **Certificate**

This is to certify that the Project report on “**Semantic-Aware Searching Over Encrypted Data For Cloud Computing**” is a bonafide work carried out by **B.Manjula(16WH1A1208), K.Ramya(16WH1A1221), B.Sravya(16WH1A1256)** in the partial fulfillment for the award of B.Tech degree in **Information Technology, BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad**, affiliated to Jawaharlal Nehru Technological University, Hyderabad under my guidance and supervision.

The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

#### **Internal Guide**

**Ms. P.S. Latha Kalyampudi**

**Assistant Professor**

**Department of IT**

#### **Head of the Department**

**Dr. Aruna Rao S L**

**Professor & HoD**

**Department of IT**

#### **External Examiner**

## **Acknowledgements**

We would like to express our sincere thanks to **Dr. K. V. N. Sunitha, Principal**, BVRIT HYDERABAD College of Engineering for Women, for providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. Aruna Rao S L, Professor & HoD, Department of IT**, BVRIT HYDERABAD College of Engineering for Women, for all the timely support and valuable suggestions during the period of our project.

We are extremely thankful and indebted to our internal guide, **Ms. P.S. Latha Kalyampudi, Assistant Professor, Department of IT**, BVRIT HYDERABAD College of Engineering for Women, for her constant guidance, encouragement and moral support throughout the project.

Finally, We would also like to thank our Project Co-coordinator, all the faculty and staff of Department of IT who helped us directly or indirectly, parents and friends for their cooperation in completing the project work.

**B.Manjula**  
**(16WH1A1208)**

**K.Ramya**  
**(16WH1A1221)**

**B.Sravya**  
**(16WH1A1256)**

## ABSTRACT

With the growing implementation of cloud computing, a increasing number of users are outsourcing their datasets to the cloud. As it is important to preserve the privacy, the datasets are required to be encrypted before outsourcing. However, the effective utilization of data has become difficult due to the common practice of encryption. For instance, it is difficult to search the given keywords in encrypted datasets. Many schemes have been proposed to make encrypted data searchable based on keywords. However, keyword-based search schemes overlook the semantic representation information of users retrieval, and cannot entirely meet with users search intention. Therefore, to be able to design a content-based search scheme and make semantic search more effective and context-aware is a difficult task. ECSSED, a novel semantic search scheme based on the concept hierarchy and the semantic relationship between concepts in the encrypted datasets is being used. ECSSED uses two cloud servers. One is used to store the outsourced datasets and return the ranked results to data users. The other one is used to compute the similarity scores between the documents and the query and send the scores to the first server. To further improve the search efficiency, we utilize a tree-based index structure to establish all the document index vectors. Employing the multikeyword ranked search over encrypted cloud data as the basic frame to propose two secure schemes. The research results based on the real world datasets show that the scheme is more efficient compared to previous schemes. It is also proved that these schemes are secure under the known ciphertext model and the known background model.

## **TABLE OF CONTENTS**

<b>CHAPTER NO.</b>	<b>CONTENTS</b>	<b>PAGE NO.</b>
	<b>ABSTRACT</b>	v
	<b>LIST OF FIGURES</b>	vii
	<b>LIST OF TABLES</b>	viii
	<b>LIST OF ABBREVIATIONS</b>	viii
<b>1</b>	<b>INTRODUCTION</b>	1
	1.1 Objective	1
	1.2 Existing System	4
	1.3 Proposed System	5
	1.4 Requirement Specifications	6
	1.4.1 Software Requirement	6
	1.4.2 Hardware Requirement	6
<b>2</b>	<b>LITERATURE SURVEY</b>	7
<b>3</b>	<b>SYSTEM DESIGN</b>	12
	3.1 Concept Hierarchy	12
	3.2 Extended Concept Hierarchy	13
	3.3 Semantic Concepts Similarity	14
	3.4 Unified Modeling Language (UML)	15
<b>4</b>	<b>SYSTEM IMPLEMENTATION</b>	18
	4.1 System Model	18
	4.2 Threat Model	20
	4.3 Design Goal	20
	4.4 Sample Code	21
	4.5 Output Screenshots	28
<b>5</b>	<b>CONCLUSION AND FUTURE SCOPE</b>	44
	5.1 Conclusion	44
	5.2 Future Scope	44
	<b>REFERENCES</b>	45

## **LIST OF FIGURES**

<b>Figure No</b>	<b>Figure Name</b>	<b>Page No</b>
2.1	Cloud Architecture	7
3.1	A concept hierarchy for college employees.	12
3.2	An example of our extended concept hierarchy.	13
3.3	A concept hierarchy Tree.	15
3.4	Usecase Diagram of Semantic-aware Searching	16
3.5	Sequence Diagram of Semantic-aware Searching	17
4.1	System model	18
4.2	Work flow of Data Owner	19
4.3	Select Engine in AWS	29
4.4	Elastic Beanstalk in AWS	30
4.5	Application Home Screen	31
4.6	Login page of Data Owner	32
4.7	Uploading Files in Data owner	33
4.8	Retrieving the Symmetric Key	34
4.9	File Encryption	35
4.10	Uploaded Files List	36
4.11	Validation for Redundancy	36
4.12	Login page of Data User	37
4.13	Search Request	37
4.14	Trapdoors in Cloud Server B	38
4.15	OutSourced Data in Cloud Server A	39
4.16	Response To User in Cloud Server A	40
4.17	Decryption Key generation in Data User	41
4.18	Key Request in Data Owner	42
4.19	Download the Output File	43

## **LIST OF TABLES**

<b>Table No</b>	<b>Table Name</b>	<b>Page No</b>
3.1	Semantic Relations in the Concept Hierarchy	12

## **LIST OF ABBREVIATIONS**

<b>Word</b>	<b>Abbreviation</b>
CSP	Cloud Service Provider
PEKS	Public Key Encryption with Keyword Search
PKI	Public Key Infrastructure
MRSE	Multi-Keyword ranked search over encrypted data
ODP	Open Directory Project
IBE	Identity-Based Encryption



## 1.INTRODUCTION

Cloud computing provides many services and applications. Handling of large quantity of data is very difficult. To seek help from the cloud server for data management. So the difficulty in handling the bulk data has addressed. However, the security of the sensitive data becomes a big concern. Cloud service providers can utilize the sensitive information for their own benefits. To address the security concern of the cloud data, need to encrypt the data and then out source it to the cloud. Searching over encryption again becomes a concern. There are many methods proposed for searching over encrypted data over cloud. All the searching method is based on keyword based search, fuzzy keyword search, multi keyword search etc. The keyword based search will not consider the semantic relation between the keywords.

### 1.1 Objective

Cloud computing is a new but gradual maturity model of enterprise IT infrastructure that provides high quality applications and services [1]. The cloud customers can outsource their local complex data system into the cloud to avoid the overhead of management and local storage. However, the security of outsourced data cannot be guaranteed, as the Cloud Service Provider (CSP) possesses whole control of the data. So, it is necessary to encrypt data before outsourcing them into cloud to protect the privacy of sensitive data [2]. Li et al. [3] provided a secure privacypreserving outsourced classification in cloud computing. However, encryption for outsourced data can protect privacy against unauthorized behaviors, it also makes effective data utilization, such as search over encrypted data, a very difficult issue.

Many researchers have proposed a series of efficient search schemes over encrypted cloud data. The general process of search scheme can be divided into five steps: extracting document features, constructing a searchable index, generating search trapdoor, searching the index based on the trapdoor and returning the search results. These search schemes provide different query capabilities, including single keyword search [2, 3], multikeyword search [7, 8], fuzzy keyword search, similarity search [9], and so on. However, all the

existing searchable encryption schemes, which consider keywords as the document feature, do not take the semantic relations between words into consideration, both in the steps of extracting document features and generating search trapdoor. The semantic relations between words are diverse [10], such as synonymy and domain correlation. Considering the potentially huge amount of outsourced data documents in the cloud, the search accuracy and search efficiency are influenced negatively if the semantic relations between words are not handled well. A detailed description of existing problems of the available searchable schemes is given.

In this stage of extracting document features, the data owner computes the weight of each word in a document and then selects  $t$  words with top- $t$  weights as the feature of the document. In the process shown above, every two words with different spelling are assumed uncorrelated, which is unreasonable. For example, two words “trousers”, “pants” are different in the perspective of spelling, but they are semantically similar [11]. It is obvious that the weight of word is influenced if semantic relations between words are ignored and the accuracy of the document features is influenced consequently.

During generating search trapdoor, the trapdoor is generated only based on the search keywords input by the data user, which is inflexible, because it is impossible to extend the search keywords when the data user cannot express his search intention well. In this case, a useless document can be returned for the data user or the really needed documents are not returned [12]. So, it is important to understand the real search intention of the data user to avoid returning unnecessary documents to improve search efficiency, as the size of the document set outsourced into the cloud server is potentially huge.

A search request usually focuses on a theme, and some search words can be considered to be the attribute of the theme, for example, birthday is an attribute of a person. In existing search schemes, an attribute value is usually treated as a keyword that ignores the relationship with the theme and results in larger keyword dictionary, and then negatively influences the search accuracy and efficiency. Therefore, it is a very important and challenging task to implement semantic search over encrypted data.

As proposed an efficient searchable encrypted scheme based on concept hierarchy supporting semantic search with two cloud servers is mentioned. A concept hierarchy tree is constructed based on domain concepts related knowledge of the outsourced dataset. An extended concept hierarchy is used to include more semantic relations between concepts. With the help of extended concept hierarchy, document features are extracted more precisely and search terms are well extended based on the semantic relations between concepts. For each document, two index vectors are generated, one is used to match concepts in the search request and another one is used to determine whether the value for an attribute is satisfied with the search request [13, 14]. Correspondingly, the search trapdoor for a search request also includes two vectors. The reason to choose two cloud servers is that two servers can save much time in search. One is used to compute the similarity between the documents vector and the trapdoors vector. Another one is used to rank results and return them to users. Contributions are summarized as follows:

- A. The study of the problem of the semantic search based on the concept hierarchy by using two cloud servers. The concept hierarchy is extended to store various semantic relations among concepts and used to extend the search terms. To improve the efficiency and security of the search, the retrieval process is split into two independent procedures.
- B. A proposed method to build the document index and search trapdoor based on the concept hierarchy to support semantic search, which filters documents by checking the attribute value and sorts related documents based on the number of matched search terms.
- C. The security analysis indicates that our scheme is secure in the threat models. A tree-based searchable index is constructed to improve search efficiency. Experiments on real world datasets show that our schemes are efficient.

Compared with the conference version [16], the study of the problem of semantic search over encrypted data based on concept hierarchy in the basic frame of MRSE. This version also study that how to do the search by using two cloud servers. In addition to these, more

detailed security analysis of the schemes are done. Meanwhile, an improvement of the experiments is done by doing evaluation of the new schemes.

## 1.2 Existing System

A detailed description of existing problems of the available searchable schemes is given. Previously researchers have proposed a series of efficient search schemes over encrypted cloud data. The general process of search scheme can be divided into extracting document features, constructing a searchable index, generating search trapdoor, searching the index based on the trapdoor and returning the search results [12, 13]. These search schemes provide different query capabilities, including single keyword search.

Firstly, in the stage of extracting document features, the data owner computes the weight of each word in a document and then selects  $t$  words with top- $t$  weights as the feature of the document. In the process shown above, every two words with different spelling are assumed uncorrelated, which is unreasonable. For example, two words “trousers”, “pants” are different in the perspective of spelling, but they are semantically similar [14]. It is obvious that the weight of word is influenced if semantic relations between words are ignored and the accuracy of the document features is influenced consequently.

Secondly, during generating search trapdoor, the trapdoor is generated only based on the search keywords input by the data user, which is inflexible, because it is impossible to extend the search keywords when the data user cannot express his search intention well. In this case, a useless document can be returned for the data user or the really needed documents are not returned [15]. So, it is important to understand the real search intention of the data user to avoid returning unnecessary documents to improve search efficiency, as the size of the document set outsourced into the cloud server is potentially huge.

Thirdly, a search request usually focuses on a theme, and some search words can be considered to be the attribute of the theme, for example, birthday is an attribute of a person. In existing search schemes, an attribute value is usually treated as a keyword that ignores the relationship with the theme and results in larger keyword dictionary, and then

negatively influences the search accuracy and efficiency [14, 15]. Therefore, it is a very important and challenging task to implement semantic search over encrypted data.

**Disadvantages:**

- Enterprise IT infrastructure that provides high quality applications and services .
- The cloud customers can outsource their local complex data system into the cloud to avoid the overhead of management and local storage.
- However, the security of outsourced data cannot be guaranteed, as the Cloud Service Provider (CSP) possesses whole control of the data.

**1.3 Proposed System**

An efficient searchable encrypted scheme is proposed based on concept hierarchy supporting semantic search with two cloud servers. A concept hierarchy tree is constructed based on domain concepts related knowledge of the outsourced dataset [16]. With the help of extended concept hierarchy, document features are extracted more precisely and search terms are well extended based on the semantic relations between concepts.

For each document, two index vectors are generated, one is used to match concepts in the search request and another one is used to determine whether the value for an attribute is satisfied with the search request. Correspondingly, the search trapdoor for a search request also includes two vectors. The reason to choose two cloud servers is that two servers can save much time in search. One is used to compute the similarity between the documents vector and the trapdoors vector. Another one is used to rank results and return them to users [17].

**Advantages:**

- Consider keywords as the document feature, do not take the semantic relations between words into consideration, both in the steps of extracting document features and generating search trapdoor.

- The semantic relations between words are diverse, such as synonymy and domain correlation [18, 19]. Considering the potentially huge amount of outsourced data documents in the cloud, the search accuracy and search efficiency are influenced negatively if the semantic relations between words are not handled well.

## **1.4 Requirement Specifications**

### **1.4.1 Software Requirements**

The software requirements specify the use of all required software products like the data management system. The required software product specifies the numbers and versions. Each interface specifies the purpose of the interfacing software as related to this software product.

- Operating System : Windows 7/10
- Coding Language : JAVA/J2EE
- Tool : Netbeans 7.2.1
- Database : MYSQL

### **1.4.2 Hardware Requirements**

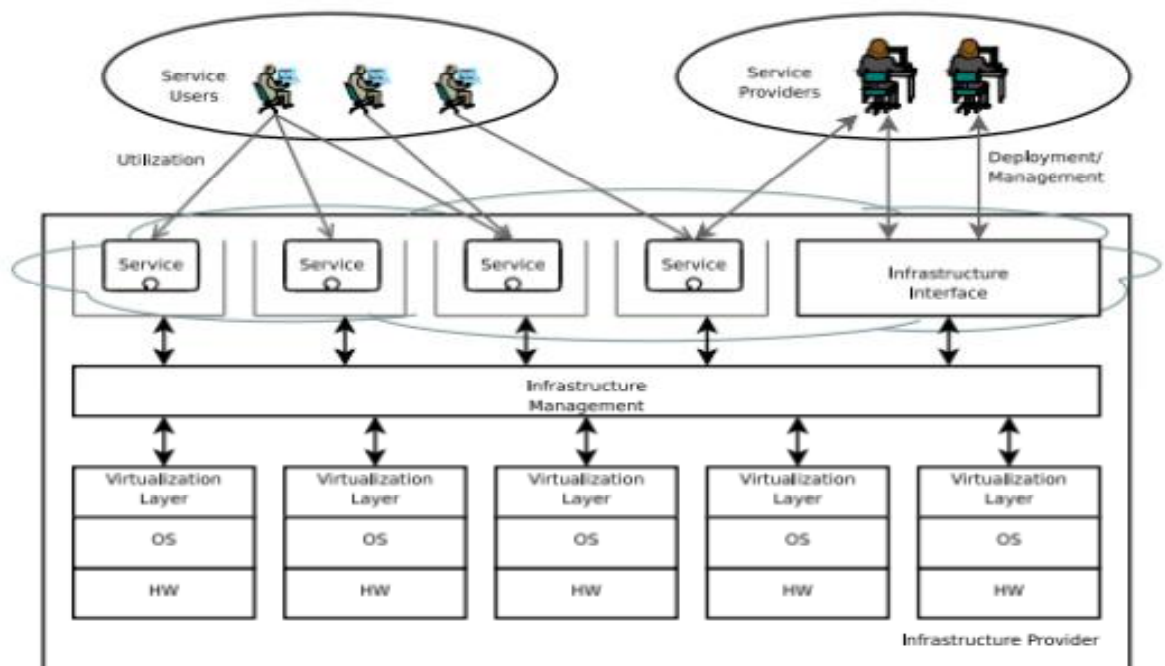
The hardware requirement specifies each interface of the software elements and the hardware elements of the system. These hardware requirements include configuration characteristics.

- System : Pentium Dual Core.
- Hard Disk: 120 GB.
- Monitor : 15 LED
- Input Devices : Keyboard, Mouse
- Ram : 1 GB

## 2.LITERATURE SURVEY

Cloud computing refers to accessing software and storing data in the cloud of the internet. It is a model for enabling convenient, on demand network access to a shared pool of configurable and reliable computing resources that can be rapidly provisioned and released with service provider interaction. The benefits brought by this new computing model include but are not limited to: relief of the burden for storage management, universal data access with independent geographical locations, and avoidance of capital expenditure on hardware, software, and personnel maintenances, etc [1].

As Cloud Computing becomes prevalent, more and more sensitive information are being centralized into the cloud, such as emails, personal health records, company finance data, and government documents, etc. The cloud server may leak data information to unauthorized entities or even be hacked. Many researchers have proposed a series of efficient search schemes over encrypted cloud data. [2], in this to address the problem of semantic retrieval, author propose effective schemes based on concept hierarchy.



**Fig 2.1 Cloud Architecture**

These Service Providers (SPs) make services accessible to the Service Users through Internet-based interfaces. Clouds aim to outsource the provision of the computing infrastructure required to host services. This infrastructure is offered ‘as a service’ by Infrastructure Providers (IPs), moving computing resources from the SPs to the IPs, so the SPs can gain in flexibility and reduce costs as shown in Fig 2.1. The fact that data owners and cloud server are no longer in the same trusted domain may put the outsourced unencrypted data at risk.

To improve accuracy, author extend the concept hierarchy to expand the search conditions. [3], uses an new semantic search scheme based on the concept hierarchy and the semantic relationship in concepts in the encrypted datasets. Cloud Computing is the long dreamed vision of computing as a utility, where cloud customers can remotely store their data into the cloud so as to enjoy the on-demand high quality applications and services from a shared pool of configurable computing resources. Many activities use software services as their business basis.

Today’s mail servers such as IMAP servers, file servers and other data storage servers typically must be fully trusted they have access to the data, and hence must be trusted not to reveal it without authorization which introduces undesirable security and privacy risks in applications. Previous work shows how to build encrypted file systems and secure mail servers, but typically one must sacrifice functionality to ensure security. The fundamental problem is that moving the computation to the data storage seems very difficult when the data is encrypted, and many computation problems over encrypted data previously had no practical solutions [4].

More specifically, our scheme first indexes the documents and builds trapdoor based on the concept hierarchy. [5] the searchable encryption which allows the user to retrieve the encrypted documents that contain the user-specified keywords, where given the keyword trapdoor, the server can find the data required by the user without decryption.

Private-key storage outsourcing allows clients with either limited resources or limited expertise to store and distribute large amounts of symmetrically encrypted data at low cost.



Since regular private-key encryption prevents one from searching over encrypted data, clients also lose the ability to selectively retrieve segments of their data.

They investigate the security of a well-known cryptographic primitive, namely Public Key Encryption with Keyword Search (PEKS) which is very useful in many applications of cloud storage.[6] in this, Identity-Based Encryption (IBE) which simplifies the public key and certificate management at Public Key Infrastructure (PKI) is an another relevant to public key encryption. Privacy-Preserving Smart Semantic Search.

Effective and efficient search capabilities for digital collections has become essential for information management and knowledge discovery. Meanwhile, a growing number of collections are professionally maintained in data centers and stored in encrypted form to limit their access to only authorized users in order to protect confidentiality and privacy. [7], in this, Considering various semantic representation tools, author select Conceptual Graphs (CG) as our semantic bearer because of its great ability of expression and extension. To improve the efficiency of retrieval, author uses Tregex simplify the key sentence and make it more generalizable.

Here transfer of CG into its linear form with some alteration which makes quantitative calculation on CG and fuzzy retrieval in semantic level possible. [8] in this, it is discussed about the general problem of secure computation on an encrypted database and propose a SCONEDB (Secure Computation ON an Encrypted DataBase) model, which captures the execution and security requirements.

Focuses on the problem of k-nearest neighbor (kNN) on encrypted datasets. [9] in this, author creates method for the construction of a concept hierarchy that takes three basic dimensions of term dependence. [10] in this, WordNet provides a more effective combination of traditional lexicographic information and modern computing. The traditional data utilization service based on plaintext keyword search. The trivial solution of downloading all the data and decrypting locally is clearly impractical, due to the huge amount of bandwidth cost in cloud scale systems.

Moreover, aside from eliminating the local storage management, storing data into the cloud serves no purpose unless they can be easily searched and utilized. WordNet is an online lexical database design to use under program control [11]. In this, exploit edit distance to quantify keywords similarity and develop leading technique while constructing fuzzy keyword sets, which reduces the storage. databases and applications are moved to the servers in the large data centers managed by the third-party cloud service providers in the Internet.

Cloud computing has been gradually recognized as the most significant turning point in the development of information technology during the past few years. People are fascinated by the benefits it offers, such as ubiquitous and flexible access, on-demand computing resources configuration, considerable capital expenditure savings, etc. [12] in this, author define and solve the challenging problem of privacy-preserving multi-keyword ranked search over encrypted data in cloud computing (MRSE). Advances in cloud computing and Internet technologies have pushed more and more data owners to outsource their data to remote cloud servers to enjoy with huge data management services in an efficient cost [13].

With analyzing cloud storage data security requirements, focus on data privacy protection in cloud storage and proposed a measure to optimize the efficiency of access control in cryptographic cloud storage. [14] For the main users that use cloud storage services are enterprise and the community users, they have the characteristics in common that they manage their data access rights by the mode of hierarchical classification. Because of this, combined ciphertext policy attribute-based encryption (CP-ABE) algorithm and hierarchical identity-based encryption (HIBE) algorithm, to identify users by both precise identity and attribute in the process of making data access control strategy and use hierarchy when generate keys[15] .

Meaningful words, any system that hopes to process natural languages as people do must have information about words and their meanings. But dictionary entries evolved for the convenience of human readers, not for machines. WordNet provides a more effective combination of traditional lexicographic information and modcomputing. WordNet is an online lexical database designed for use under program control [16]. English nouns, verbs, adjectives, and adverbs are

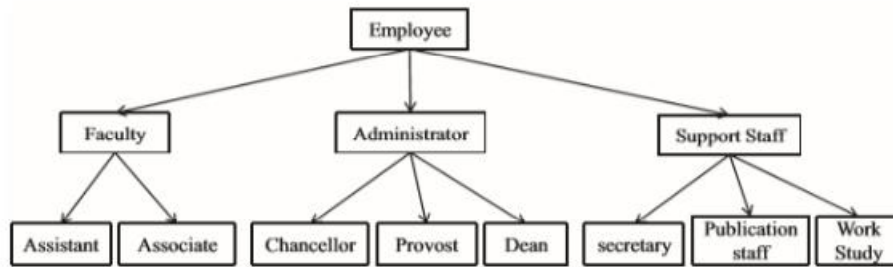
organized into sets of synonyms, each representing lexicalized concept. Semantic relations link the synonym sets [17].

Classifier has been widely applied in machine learning, such as pattern recognition, medical diagnosis, credit scoring, banking and weather prediction. Because of the limited local storage at user side, data and classifier has to be outsourced to cloud for storing and computing [18, 19]. However, due to privacy concerns, it is important to preserve the confidentiality of data and classifier in cloud computing because the cloud servers are usually untrusted. A framework for privacy-preserving outsourced classification in cloud computing (POCC). Using POCC, an evaluator can securely train a classification model over the data encrypted with different public keys, which are outsourced from the multiple data providers [20].

### 3. SYSTEM DESIGN

#### 3.1 Concept Hierarchy

A concept hierarchy is an organized concept set using hierarchical method. In the hierarchy, the concepts at lower levels contain more specific meanings than those at higher levels. At first, the concept hierarchy is generated based on the domain information of the outsourced dataset. And then the dataset is dealt to extend the concept hierarchy. Concept hierarchy can be created by its own, based on the number of distinct values per attribute in the known attribute set. The attribute with the utmost specific values is placed at the lowest level of the hierarchy.



**Fig. 3.1 A concept hierarchy for college employees.**

- I. Auto generate the attribute ordering based upon observation that attribute defining a high level concept has a smaller # of distinct values than an attribute defining a lower level concept
- II. Example : Fig 3.1 shows concept hierarchy for college employees.

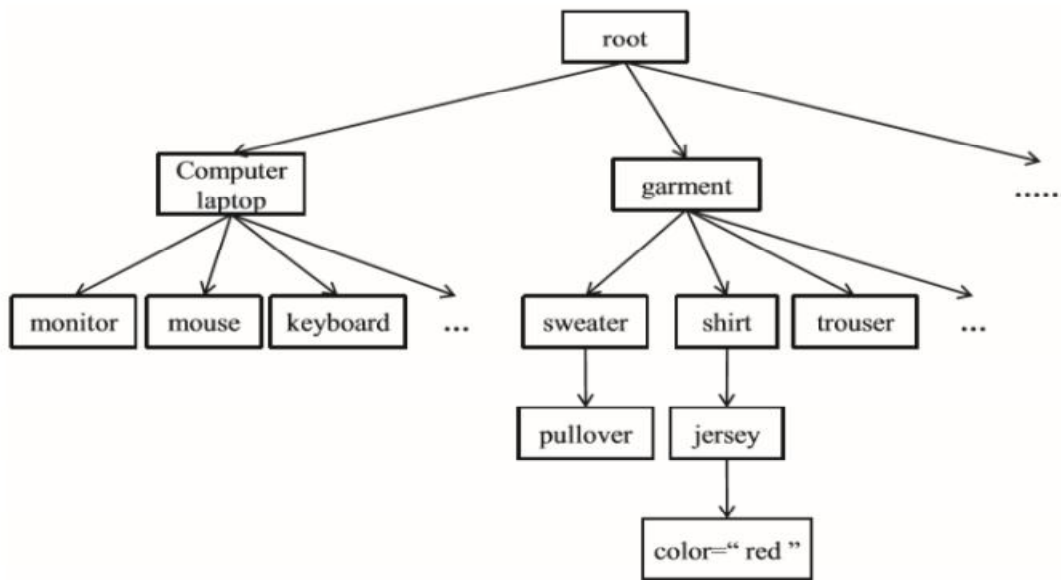
**Table .3.1 The Semantic Relations in the Concept Hierarchy**

Semantic Relation Type	Example
Synonym	Web - Network
Hypernymy/Hyponymy	Fruit - Apple
Meronym / holonym	Forest - tree
Host - Attribute	People - name
Attribute – value	Color - red

The domain concept hierarchy can be obtained by some existing tool, such as WordNet [17], The similarity between two concepts is calculated based on the distance of them in the concept hierarchy. A semantic relation in the concept hierarchy is shown in Table 3.1.

### 3.2. Extended Concept Hierarchy

The text file is unstructured, but the language organizing the text file exists semantic relation, which can be regarded as an implicit structure. An extended concept hierarchy is used to denote the semantic relationship between concepts [17]. The semantic relations contained in our extended concept hierarchy. The main difference between our extended concept hierarchy and concept hierarchy is that, the concept can possess attribute, which can be assigned different values



**Fig. 3.2 An example of our extended concept hierarchy.**

Some concepts can possess some attributes that are also concepts. The concept “doctor” as an example. A “doctor” can have attributes like “name”, “gender”, and so on. The attribute information can make the search result more accurate, as it makes the search request more specific. And as the concept and attribute are organized via the concept hierarchy, the semantic relationship is preserved. Taking Fig. 3.2 as an example to illustrate the extended

concept hierarchy [16, 17]. As shown in Fig. 3.2, the attribute “color” of concept “jersey” has a value “red”. Note that the concept acted as attribute has not relation with other concepts. The hypernymy/hyponymy relation and synonym relation can also be seen in Fig. 3.2. The paragraph shows that the process of generating the extended concept hierarchy. At first, the concept hierarchy is generated based on the domain information of the outsourced dataset.

And then the dataset is dealt to extend the concept hierarchy. The domain concept hierarchy can be obtained by some existing tools, such as WordNet, the hyponymy of which could be regarded as relationship of superclass and subclass, or some existing concept hierarchy, such as web directory ODP(open directory project) . The content of a document usually focus on a subject that can be denoted by concepts and relations of concepts. For example, the sentence “A paper about economy is published in the newspaper on March 5, 2014” is the subject of a document, which can be denoted by concepts and relations among them.

There are many subject extraction techniques with which the subjects of files are obtained. And then the concepts and its relations are generated to extend the concept hierarchy. As WordNet includes all the semantic relationships that adopted in the extended concept hierarchy, utilize it to construct our extended concept hierarchy [17]. The values of a certain attribute concept are determined by the outsourced dataset.

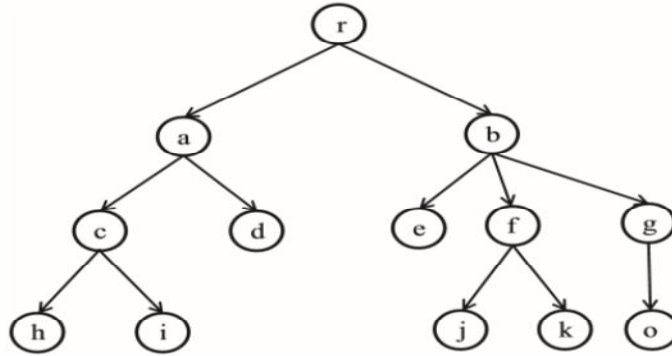
### **3.3.Semantic Concepts Similarity**

A detailed description of how to evaluate the semantic similarity between two concepts is given. The similarity between two concepts will be used in the stage of query to extend search terms. After the concept hierarchy is built, the semantic similarity between two concepts can be calculated. The similarity between two concepts is calculated based on the distance of them in the concept hierarchy. Given two concepts  $c1$  and  $c2$ , are denoted the distance between them by  $dis(c1,c2)$ . The similarity between them can be calculated as  $sim(c1,c2) = 1-dis(c1,c2)$ . The issue of calculating the distance between two concepts has been studied by some previous work. To make some modifications on their original thought

to fit our requirement [18]. Each node  $z$  in the concept hierarchy owns a value, denoted by  $\text{Score}(z)$ ,

$$\text{Score}(z) = 1/2^{\text{kd}(z)}(1)$$

where  $k$  is a factor which is defined as 2 in this paper, and  $d(z)$  is the depth of node  $z$  in the concept hierarchy. Note that for the root node of the concept hierarchy tree, define  $d(\text{root}) = 0$ . For two concepts  $c_1$  and  $c_2$ , and let concept  $cp$  be the closest common parent of them.



**Fig. 3.3 A concept hierarchy Tree.**

$$\text{dis}(c_1, c_2) = \text{dis}(c_1, cp) + \text{dis}(c_2, cp) \quad \text{dis}(c, cp) = \text{Score}(cp) - \text{Score}(c) \quad (2)$$

This evaluation method for semantic similarity provides the feature that the differences between two concepts in lower level are smaller than those between concepts in upper level. And the method also supports our requirement that the similarity between “father” and “son” should be greater than that between “brothers” as shown in Fig 3.3.

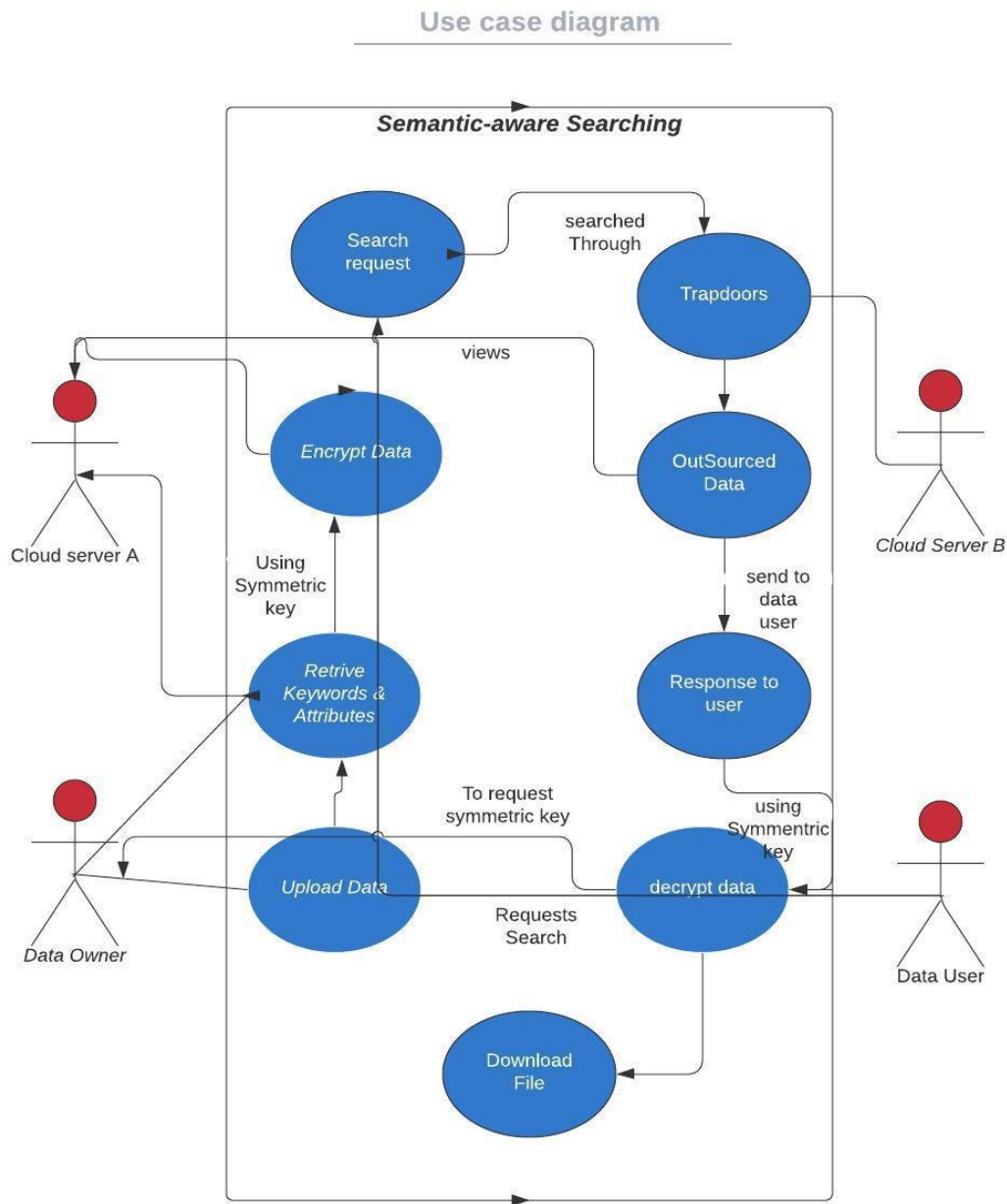
### 3.4. Unified Modeling Language (UML)

It is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems.

#### 3.4.1. Use Case Diagram:

A use case diagram is a dynamic or behavior diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions,

services, and functions that the system needs to perform.



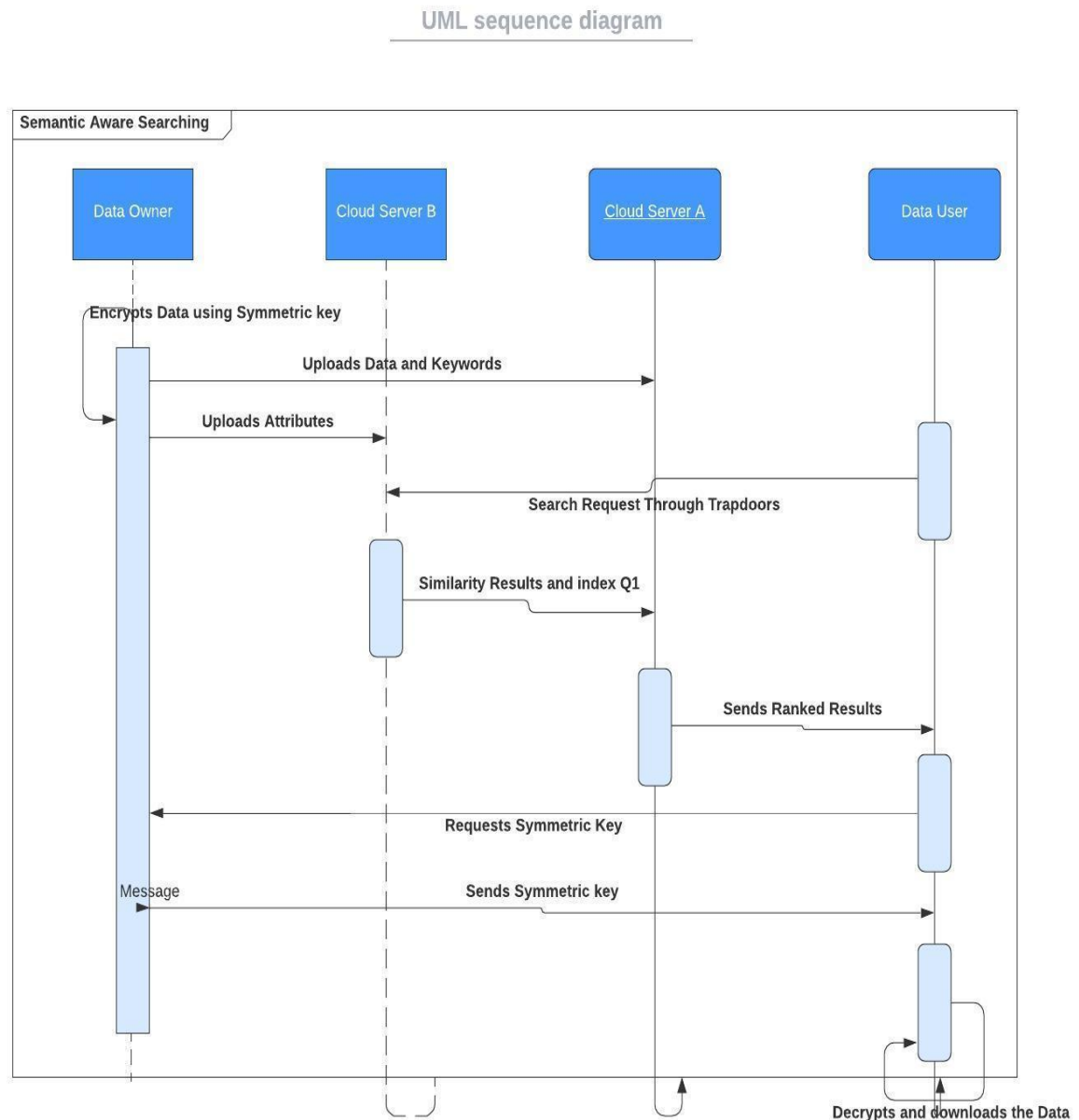
**Fig 3.4 Use case Diagram of Semantic-aware Searching**

The UML use case diagram here, is used to diagrammatically represent the complete work flow of the Application through action figures and process diagrams. It is a simpler representation of the flow of action that occurs in the application as shown in Fig 3.4.



### 3.4.2. Sequence Diagram:

It simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. One can also use the terms event diagrams or event scenarios to refer to a sequence diagram.



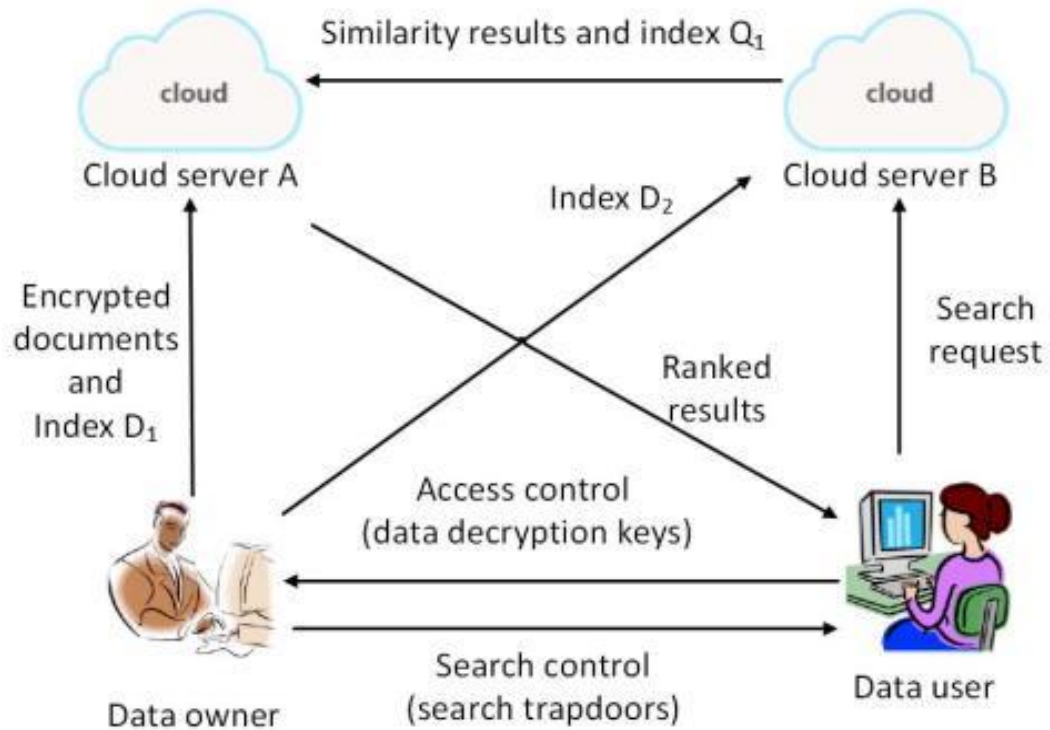
**Fig 3.5 Sequence Diagram of Semantic-aware Searching**

The Sequence diagram here, diagrammatically represents the proper work flow of the whole application in an systematical manner and more understandable method as shown in the Fig 3.5.

## 4. SYSTEM IMPLEMENTATION

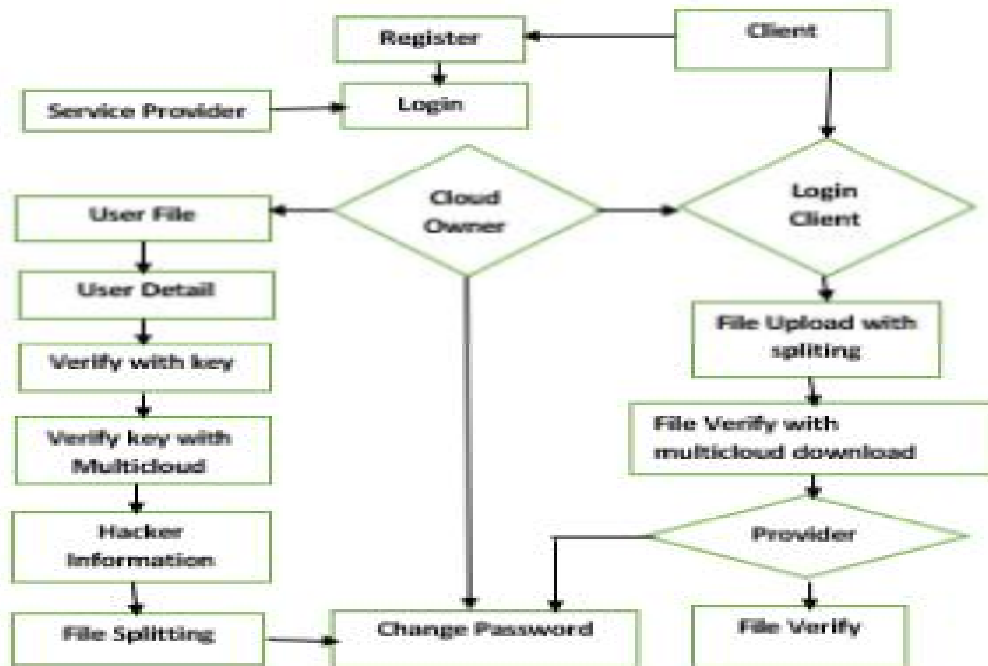
### 4.1 System Model

An innovation on this version is that to use two cloud servers to search, to make a new system model. There are four entities in our setting as shown in Fig. 4.1: the data owner, the data user, the cloud server A and the cloud server B.



**Fig. 4.1 System model**

**Data owner:** The data owner encrypts the data held locally and uploads it to the cloud server. A concept hierarchy is constructed based on the domain concepts related knowledge of the dataset and two index vectors for each document of the dataset are generated based on the key concepts of the document and the concept hierarchy [8]. Then, the searchable index which is constructed with all the index vectors is sent to the cloud A.



**Fig. 4.2 Work flow of Data Owner**

As shown in the Fig 4.2 the work flow of the Data owner is given in detail representing each process step.

**Data users:** The authorized data user makes a search request. Then, the trapdoors which related to the keywords are generated. At last, the data user sends the trapdoors to the cloud B.

**Cloud Server A:** The cloud server A has two functions. One is storing the outsourced dataset. The other one ranks the results from the cloud B and returns the certain encrypted documents that satisfy the search criterion to data users.

**Cloud Server B:** The cloud server B is used to compute the similarity scores between documents vector and trapdoors vector when it receives the trapdoor. After computing, the cloud B submits these results to the cloud A.

## 4.2 Threat Model

It is a simply introduced threat model. The scheme mainly refers to the dual-servers framework[18] and the MRSE framework [7]. Here it is considered the cloud server.

To be semi-honest, which is adopted by most previous works that is to say, who honestly executes the protocol as it is defined and correctly returns the search results, but who may also try to infer private information by analyzing the outsourced dataset, searchable index and query evaluation. And assume that there is no collusion between two cloud servers.

The known ciphertext model means that the cloud servers can access the encrypted information which contains the files and indexes outsourced by data owners, but the servers can not understand the plaintext information in the lower layer of the ciphertext. Known Background Model, this is a more powerful model, the cloud servers should have much more accessible information compared with known ciphertext model.

These information include the encrypted information and the relationship between given search requests (trapdoors) and the data set about the statistical information [12]. As the instance which can be attacked in this situation, the cloud servers can infer/recognize some retrieved keywords by using the known trapdoor information and the frequency of documents/keywords.

## 4.3 Design Goal

Increase in the part of design goals to make this paper more clear. In order to ensure that our solutions can be implemented accurately and efficiently under the above-mentioned threat models, our schemes must meet two requirements: semantic retrieval based on concept hierarchy and privacy preserving.

The semantic retrieval based on concept hierarchy means that our scheme can compute the similarity scores between the data and the search request and return the ranked results which satisfied the search requests of users [13]. Description of privacy preserving in detail.

In the search processes under the cloud servers, our schemes must meet the following privacy protection:

- A. **Data privacy:** When data documents to users is provided, there is a need to ensure the privacy of the document security which is data privacy. To address this problem, the traditional symmetric cryptography has been proposed. The advantage of this encryption is that one can use a symmetric key encrypted the data documents before outsourcing.
- B. **Index privacy:** Index privacy is that the cloud servers can not guess the correspondence between the keywords and the encrypted documents through the encrypted index.
- C. **Concept privacy:** It is believed that the concepts and the keywords are linked to a certain degree. Therefore, there is a need to ensure that the security trapdoor is generated does not reveal the keywords and the query information of users.
- D. **Trapdoor unlinkability:** While the cloud servers retrieve documents, it is able to access the generated trapdoors. Therefore, one should guarantee that the randomness of trapdoor generation [14]. At the same time, there is a need to ensure that the same queries associate with a lot of different trapdoors. In this way, the cloud server can not get relationships which exist in these trapdoors.

#### 4.4 Sample Code

##### **DataBase Connection:**

```
package databaseconnection;
import java.sql.*;

public class databasecon
{
    static Connection co;
    public static Connection getconnection()
    {

        try
        {
            Class.forName("com.mysql.jdbc.Driver");
            co=DriverManager.getConnection("jdbc:mysql://localhost:3306/"+cloud+"","root","root");
        }
    }
}
```

```

co = DriverManager.getConnection("jdbc:mysql://semanticdatabase.cfwxx2uvbirh.us-east-
2.rds.amazonaws.com:3306/semanticsearch","semantic","semantic#123");
    }
    catch(Exception e)
    {
        System.out.println("Database Error"+e);
    }
    return co;
}
}

```

**To upload and store files requested by DataOwner:**

```

package Fileupload;

import java.io.IOException;
import java.io.InputStream;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import java.sql.*;
import java.io.*;
import java.util.*;
import javax.servlet.ServletException;
import javax.servlet.annotation.MultipartConfig;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.*;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.Part;
import java.security.*;
import java.security.spec.*;
import java.nio.file.Path;
import java.nio.file.Paths;
import databaseconnection.*;
@WebServlet("/UploadAnyFiles")
@MultipartConfig(maxFileSize = 16177215) // upload file's size up to 16MB
public class UploadAnyFiles extends HttpServlet {

    Connection conn = null;

    protected void doPost(HttpServletRequest request,

```

```
        HttpServletResponse response) throws ServletException, IOException {
        // gets values of text fields
        HttpSession session=request.getSession();
//out.println(user);
        try{

                conn =databasecon.getConnection();
                Statement st2 = conn.createStatement();

        InputStream inputStream = null,inputStream2 = null;// input stream of the upload file
        // obtains the upload file part in this multipart request
        Part filePart = request.getPart("file");

                String fileName = getFileName(filePart);

        //Get all the parts from request and write it to the file on server

        if (filePart != null) {

                // prints out some information for debugging

                System.out.println(filePart.getName());
                System.out.println(filePart.getSize());
                System.out.println(filePart.getContentType());
                System.out.println("fileName"+fileName);
                // obtains input stream of the upload file

                inputStream = filePart.getInputStream();
                //inputStream2 = filePart.getInputStream();

        }

                //      byte[] buffer = new byte[inputStream2.available()];
                //      inputStream2.read(buffer);

                //      File targetFile = new File(fileName);
                //      OutputStreamoutStream          =          new
        FileOutputStream(targetFile);
                //      outStream.write(buffer);

                Statement st1=conn.createStatement();
                st1.executeUpdate("delete from loadfile");
                String sql = "insert into loadfile values(?,?)";
                PreparedStatement statement = conn.prepareStatement(sql);
```

---

```
statement.setString(1, fileName);
if (inputStream != null) {
    // fetches input stream of the upload file for the blob column
    statement.setBinaryStream(2, inputStream,(int)
filePart.getSize());
}

// sends the statement to the database server
int row = statement.executeUpdate();
System.out.println("r="+row);
if (row > 0) {
    Random randomGenerator = new Random();
    int randomInt=0;
    randomInt = randomGenerator.nextInt(10000000);

    getServletContext().getRequestDispatcher("/access.jsp?id="+randomInt+"
").forward(request, response);
}

} catch (Exception e1) {e1.printStackTrace();}
finally {
    if (conn != null) {
        // closes the database connection
        try {
            conn.close();
        } catch (SQLException ex) {
            ex.printStackTrace();
        }
    }
    // sets the message in request scope
    // request.setAttribute("Message", message);

    // forwards to the message page

}
}
```

```
private String getFileName(final Part part) {

    final String partHeader = part.getHeader("content-disposition");

    String[] sections = partHeader.split(";");

    for (String content : sections) {
```



```
        if (content.trim().startsWith("filename")) {  
            return content.substring(content.indexOf('=') + 1).trim()  
        }.replace("\\\"", "");  
    }  
}  
  
return null;  
}  
}
```

**Occurance of each word in the uploaded file:**

```
package ct;  
import java.io.*;  
import java.util.*;  
  
public class Occurance {  
    public static Vector main(String str) throws IOException {  
        Vector v=new Vector();  
        LinkedHashMap<String, Integer> wordcount =  
            new LinkedHashMap<String, Integer>();  
        try {  
  
            str = str.toLowerCase(); // convert to lower case  
            String[] words = str.split("\\s+"); //split the line on whitespace, would return an array of  
            words  
  
            for( String word : words ) {  
                if( word.length() < 5 ) {  
                    continue;  
                }  
                Integer occurrences = wordcount.get(word);  
                if( occurrences == null ) {  
                    occurrences = 1;  
                } else {  
                    occurrences++;  
                }  
                wordcount.put(word, occurrences);  
            }  
        }  
    }  
}
```

```
    }

    }
    catch(Exception e){
    System.out.println(e);
    }

    ArrayList<Integer> values = new ArrayList<Integer>();
    values.addAll(wordcount.values());

    Collections.sort(values, Collections.reverseOrder());

    int last_i = -1;

    for (Integer i :values.subList(0,3)) {
        if (last_i == i) // without duplicates
            continue;
        last_i = i;

        for (String s :wordcount.keySet()) {
            if (wordcount.get(s) == i) // which have this value
            {

                if(i>2){
                    System.out.println(s+ " " + i);
                    v.add(s+" Occured "+ i);
                }
            }
        }
        return v;
    }
}
```

### **Secure Hash Algorithm (SHA)-256:**

**SHA-256** is a member of the SHA-2 cryptographic hash functions designed by the NSA. SHA stands for Secure Hash Algorithm. Cryptographic hash functions are mathematical operations run on digital data by comparing the computed "hash" to a known and expected hash value, a person can determine the data's integrity. A one-way hash can be generated from any piece of data, but the data cannot be generated from the hash.

```
package ct;
import java.io.UnsupportedEncodingException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.io.*;
public class SHA_256
{
    public static String hashCode(String filedata)
    {byte[] hashedBytes=null;

        try {
            MessageDigest digest = MessageDigest.getInstance("SHA-256");
            hashedBytes = digest.digest(filedata.getBytes("UTF-8"));

            } catch (Exception ex) {
                System.out.println(ex);
            }

            return convertByteArrayToHexString(hashedBytes);
        }
        static String convertByteArrayToHexString(byte[] arrayBytes) {
            StringBuffer stringBuffer = new StringBuffer();
            for (int i = 0; i<arrayBytes.length; i++) {
                stringBuffer.append(Integer.toString((arrayBytes[i] & 0xff) + 0x100, 16)
                .substring(1));
            }
            return stringBuffer.toString();
        }
    }

    public static void main(String a[]){

        //System.out.println(SHA.SHA_1("hi","h"));
    }
}
```

### **User Data Access Object(DAO):**

The data access object in a computer software which is as an object which is responsible for providing abstract interface for communication to a specific form of database. Through the method of mapping, this is able to call the persistence layer and the DAO then provides a certain type of data operations.

```
package in.techyari.tutorial.dao;

import in.techyari.tutorial.model.User;
```

```
import in.techyari.tutorial.dao.Database;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.sql.PreparedStatement;

public class UserDao {

    public ArrayList<User> getUsers(String s) throws InstantiationException,
        IllegalAccessException, ClassNotFoundException, SQLException {

        ArrayList<User> userList = new ArrayList<User>();
        Database db = new Database();
        Connection connection = db.getConnection();

        try {
            PreparedStatement ps = connection.prepareStatement("SELECT *
FROM user_keywords WHERE keyword like ? and ocount>5 ");
            ps.setString(1, "%" + s + "%");
            ResultSet rs = ps.executeQuery();

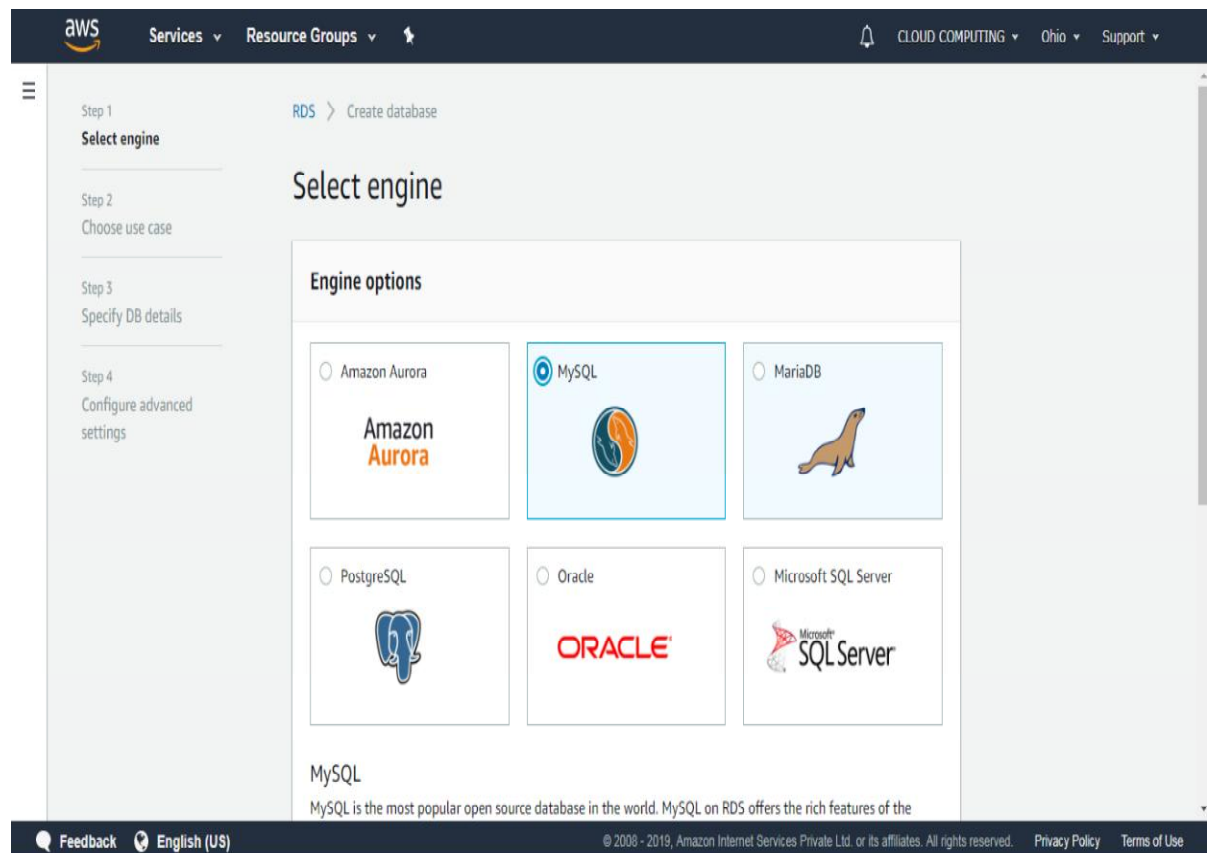
            while(rs.next()){
                User user = new User();
                user.setId(rs.getString("ocount"));
                user.setName(rs.getString("keyword"));
                userList.add(user);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return userList;
    }
}
```

#### 4.5 Output Screenshots

The semantic search-ware technique is tested using searching a keyword through with the output file is returned as an output. For the accessibility of the application in multiple Devices have used Amazon Web Services (AWS), through which the whole application is accessed through a URL link. This procedure is implemented by deploying the project into AWS where RDS for Database and Elastic Beanstalk for deploying are used.

**4.5.1 Amazon Web Services (AWS):** Amazon Web Services is a subsidiary of Amazon that provides on-demand cloud computing platforms and APIs to individuals, companies, and governments.

- **Relational Database Services (RDS) (MySQL):** In this service a database is created for storing the application data and generate data base endpoint for connecting MySql data base from our application as shown in Fig 4.3.



**Fig 4.3 Select Engine in AWS**

- **Elastic Beanstalk :** AWS Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with Java and servers such as Apache Tomcat as shown in Fig 4.4.

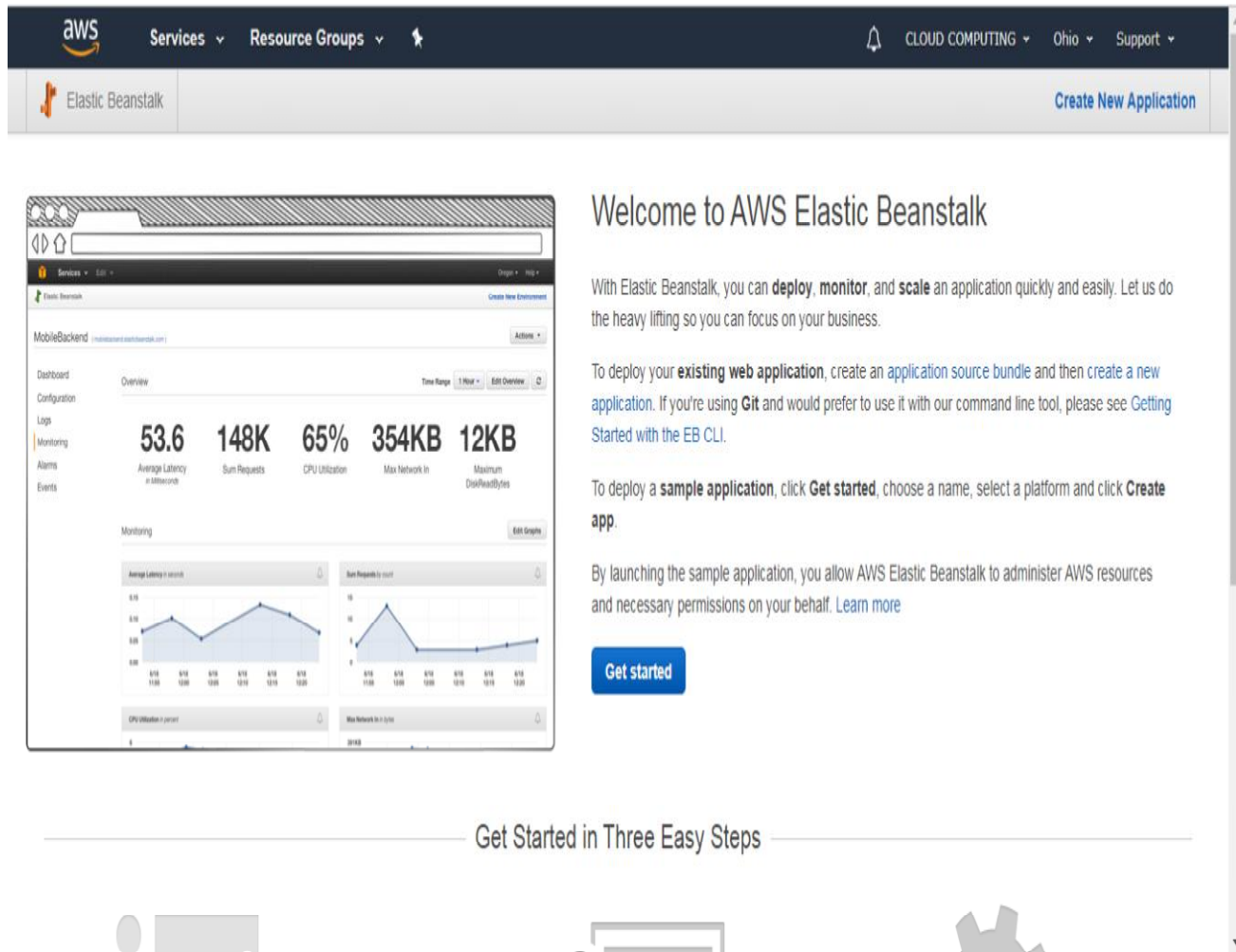


Fig 4.4 Elastic Beanstalk in AWS

From successful implementation of AWS, user will be provided with an URL as the output.

URL: <http://semanticsearch-env.eba-apwfsek2.us-east-2.elasticbeanstalk.com/>

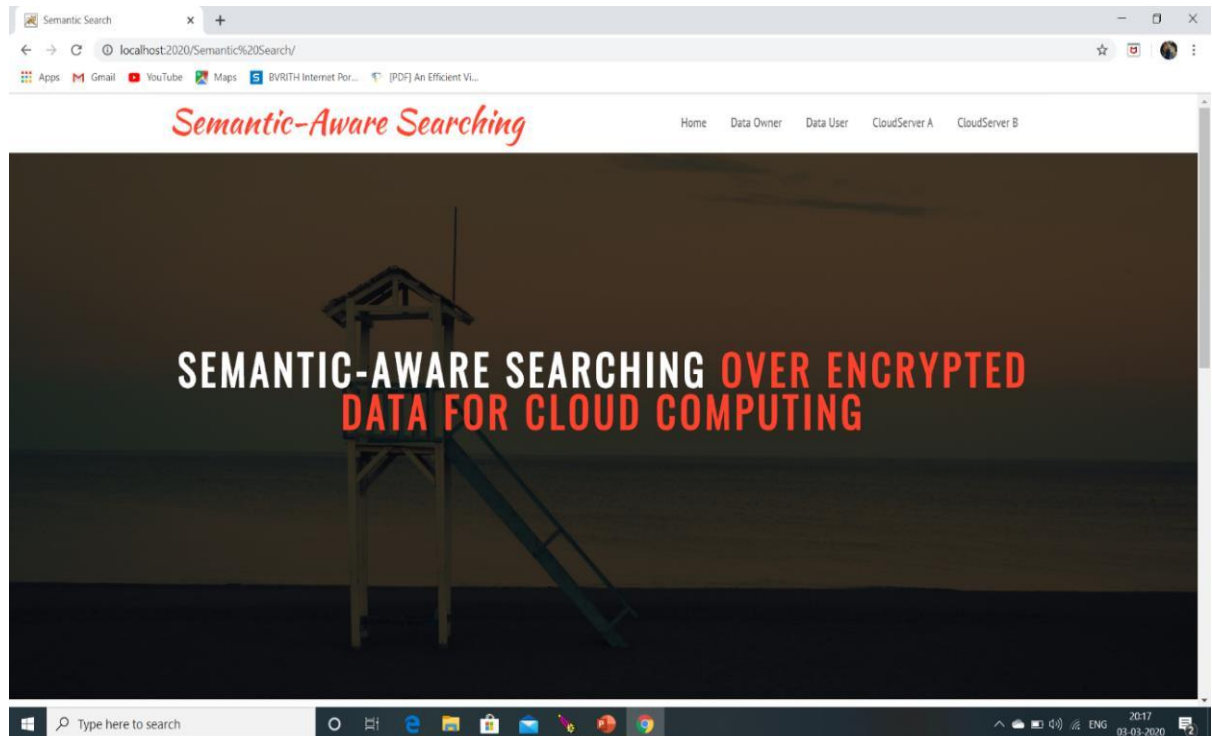
As the user opens the URL provided above the Application is opened viewing the homepage.

#### 4.5.2 Semantic-Aware Searching

The general process of search scheme can be divided into five steps: extracting document features, constructing a searchable index, generating search trapdoor, searching the index based on the trapdoor and returning the search results.

**Application-Home Screen:**

The home screen of the application is the webpage that opens when the URL is clicked and consists of the modules which can be accessed here. Each module can be opened and following functions can be implemented.

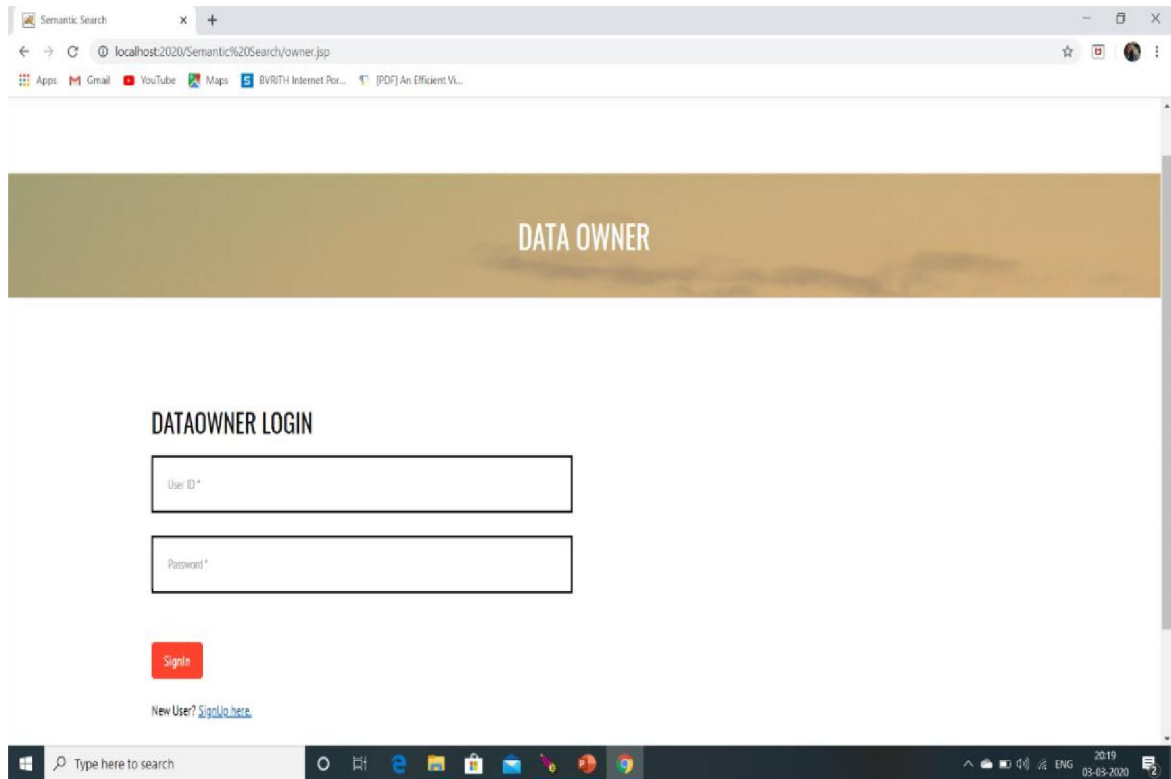


**Fig 4.5 Application Home Screen**

The home screen represents all the modules i.e Dataowner, Datauser, Cloud server A, Cloud server B. The first step is for the user to login into the Data Owner to Upload and store the files. This is done by clicking on the Data owner tab as shown in Fig 4.5.

**Login page-Data owner:**

In the data owner module, first the user has to login or register to access the respective module and implement the following functions. The data owner is used to upload any data with security measures.



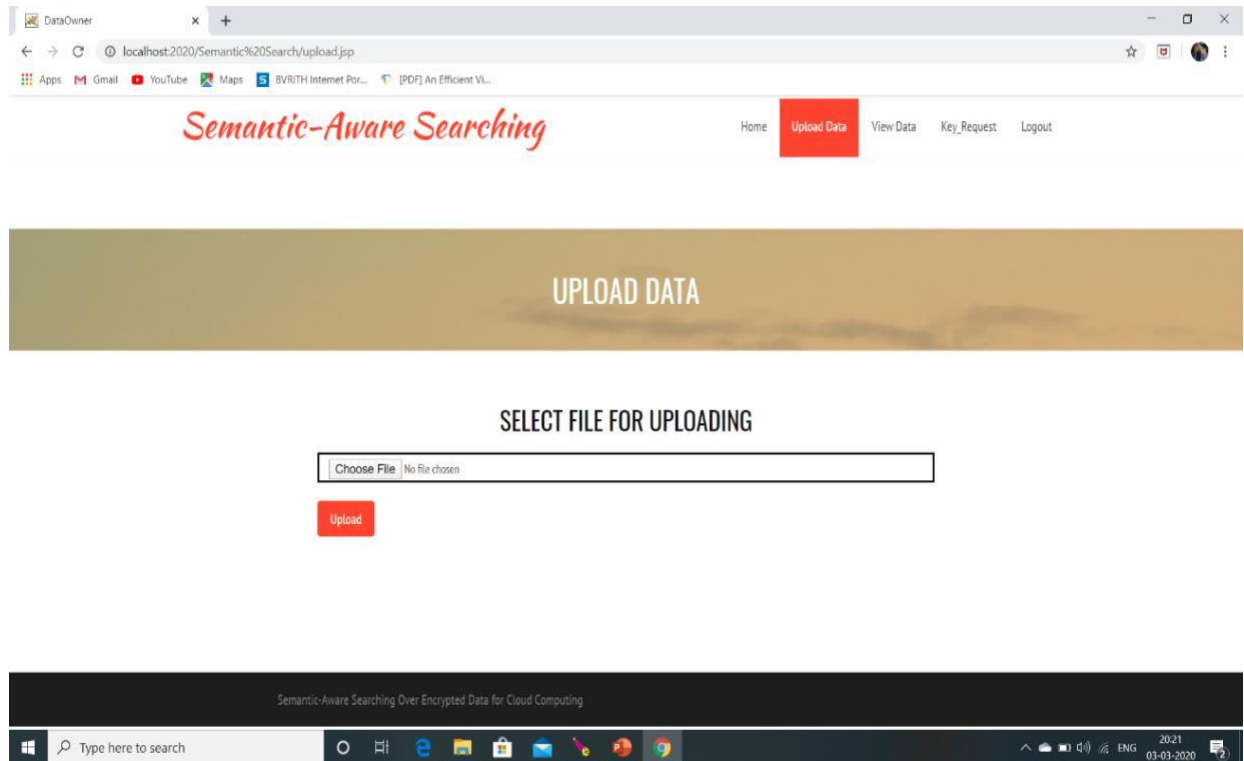
**Fig 4.6 Login page of DataOwner**

As shown in the Fig 4.6, User can login by giving user id and password, if they are a new user, they can Sign up by clicking on hyperlinked sign up text. Where user gives few valid details to create an account.



**Data Owner-Upload Data:**

To upload data, a file has to be chosen from the device and can be uploaded into the application data storage. The user chooses and clicks on the upload button and the file to be uploaded appears.

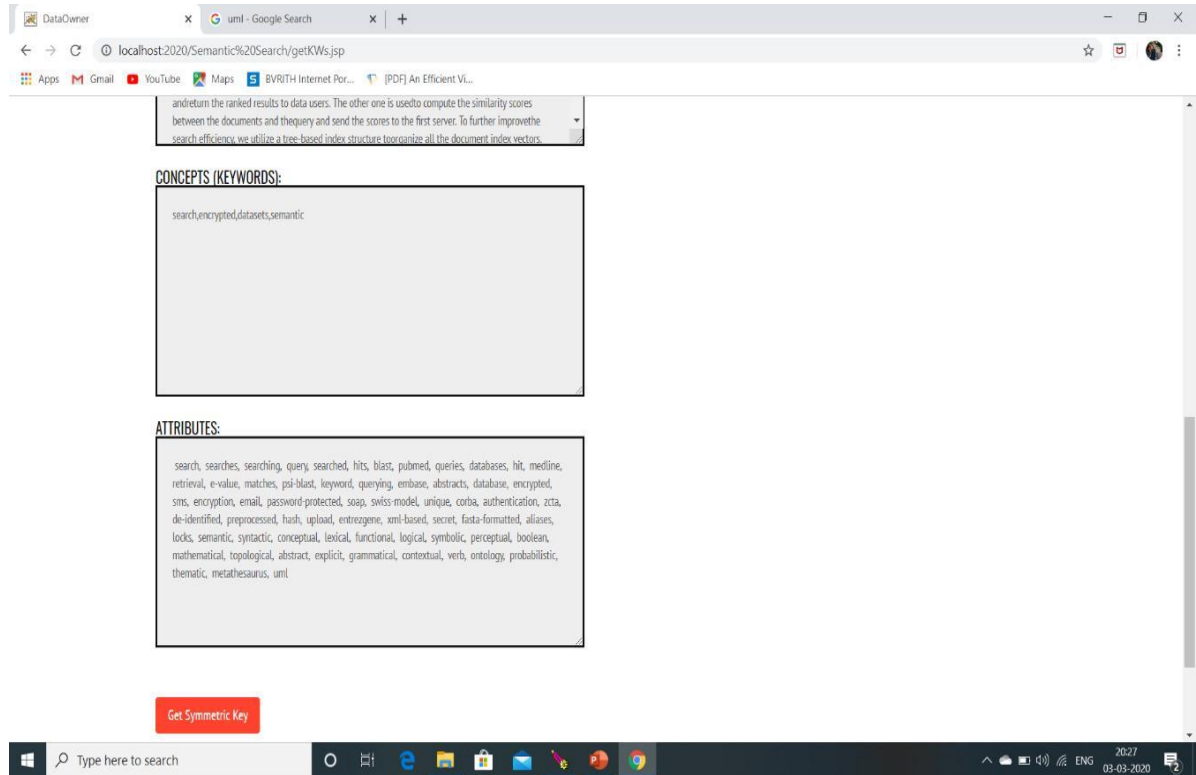


**Fig 4.7 Uploading File in Dataowner**

One of the functions in DataOwner module is to Upload the file. Where the data owner can upload files into the cloud. This is implemented as shown in Fig 4.7.

### Upload Data-Extract Keywords & Attributes:

Once the user clicks the upload button, this page occurs where the keywords and attributes from the uploaded file are retrieved and a button to get the symmetric appears to encrypt the file before storing.

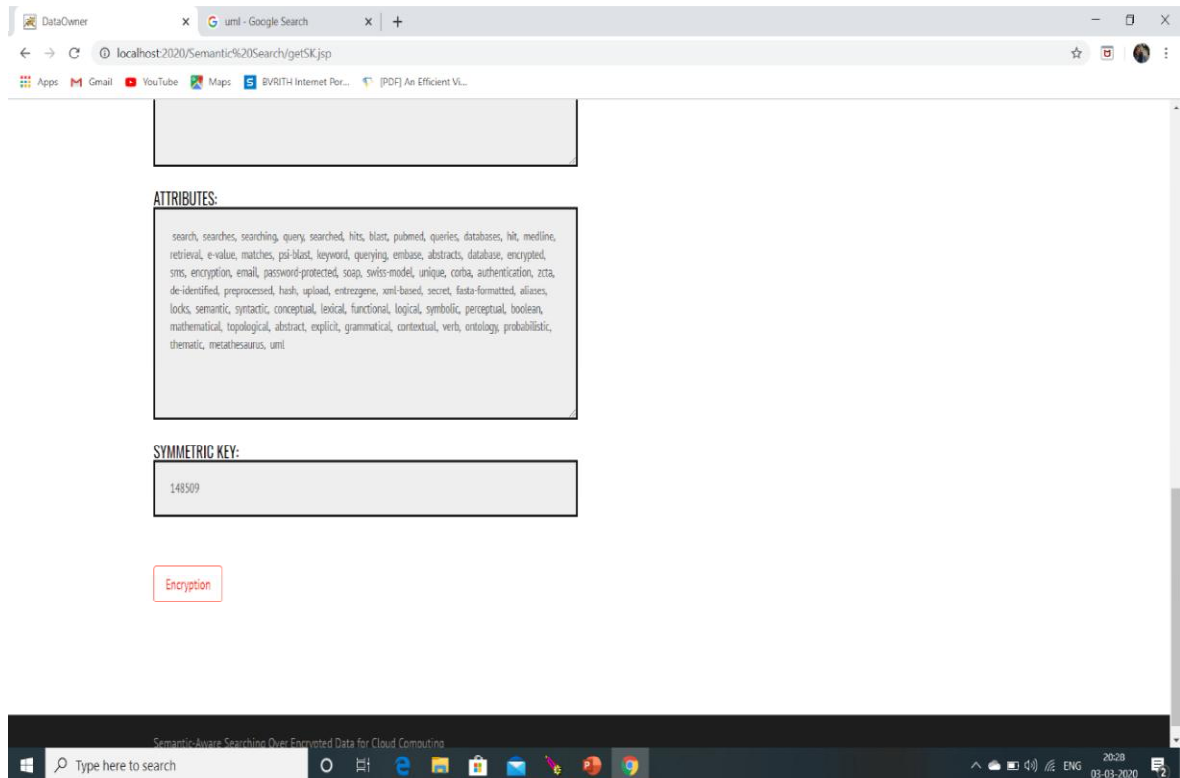


**Fig 4.8 Retrieving the Symmetric Key**

To make the search accurate, Keywords and attributes are extracted as shown in Fig 4.8, from the file and are sent to cloud server A & B respectively. Then an Button is provided to retrieve Symmentric key.

### Upload Data-Encrypt File:

As the symmetric key is retrieved, the file including the keywords and attributes are encrypted once the encryption button is clicked.



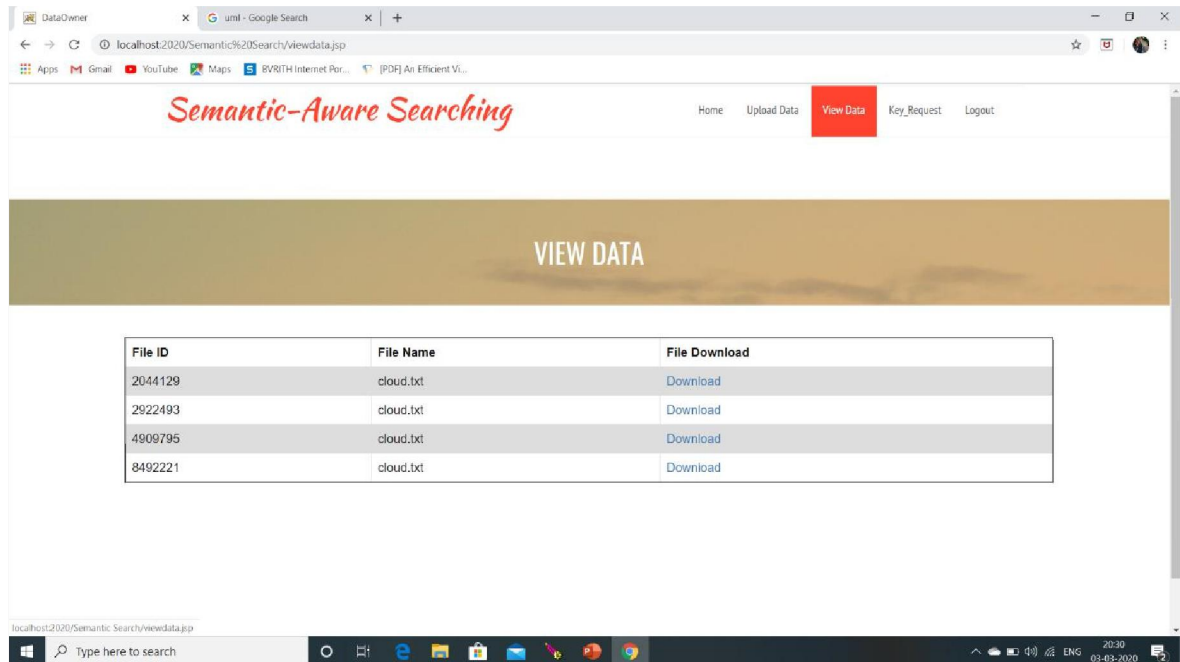
**Fig 4.9 File Encryption**

After generating Symmetric key for encrypting the file, the whole file including the attributes and keywords is encrypted using traditional symmetric encryption Algorithm as shown in Fig 4.9.

### Data Owner-View Data:

One of the functions of Data owner is to view the files that have been uploaded, here same file is uploaded multiple times, this is restricted in the application by using SHA-256.

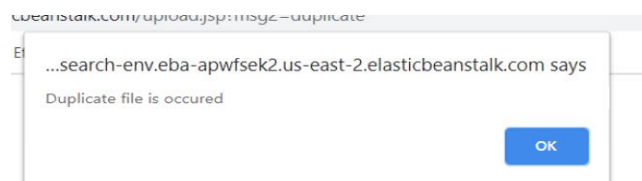
Therefore, redundancy is removed.



File ID	File Name	File Download
2044129	cloud.txt	<a href="#">Download</a>
2922493	cloud.txt	<a href="#">Download</a>
4909795	cloud.txt	<a href="#">Download</a>
8492221	cloud.txt	<a href="#">Download</a>

**Fig 4.10 Uploaded Files List**

As shown in Fig 4.10, Same file is uploaded multiple times. To remove this redundancy SHA-256 is used, after which when the user tries to upload same file more than once, a validation occurs as shown in Fig 4.11.

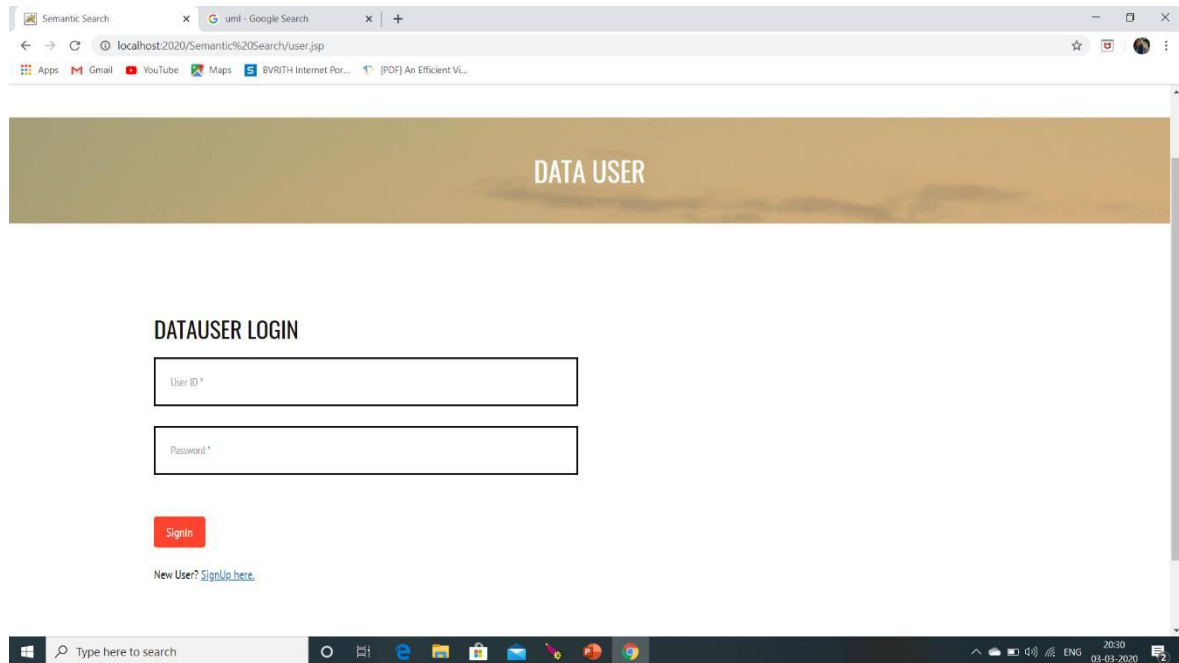


**Fig 4.11 Validation for Redundancy**

When the validation is popped up, press ok and continue uploading required files which are not duplicates as shown in Fig 4.11.

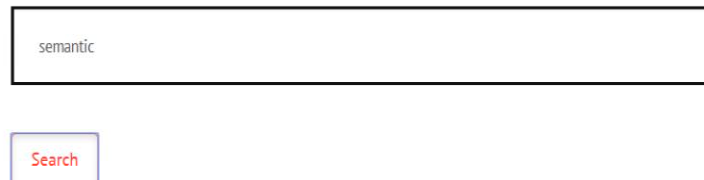
**Data User-Login:**

The second module datauser, can be used by first login in or registering. This can be done by signing up after which multiple Datauser functions are visible.



**Fig 4.12 Login page of DataUser**

The user can login into the Datauser module to search for the required file using the search request function enabled in the module as shown in Fig 4.12.

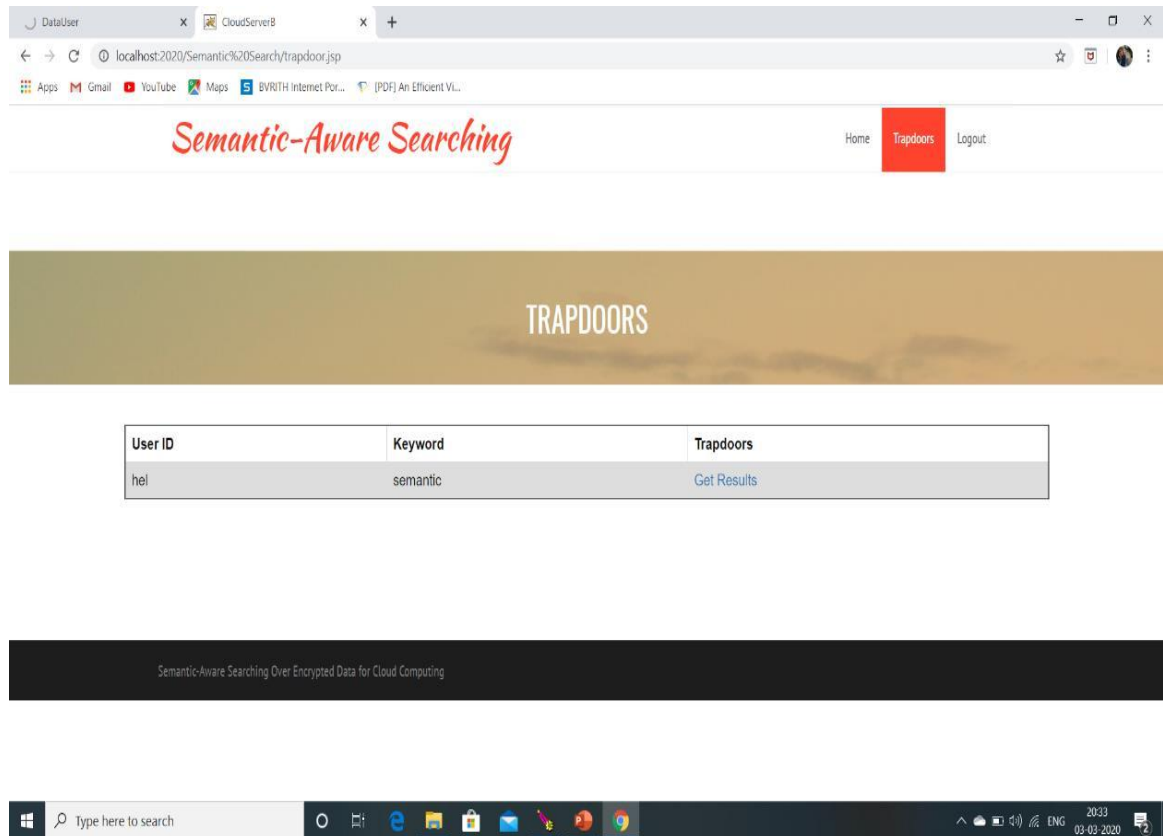


**Fig 4.13 Search request**

User types the keywords to search for the required file which is more accurate to the concept he requires as shown in Fig 4.13.

### Cloud server B-Trapdoors:

The third module, cloud server B can be used to visit the trapdoors of the cloud, through which the keywords are searched from the files uploaded.

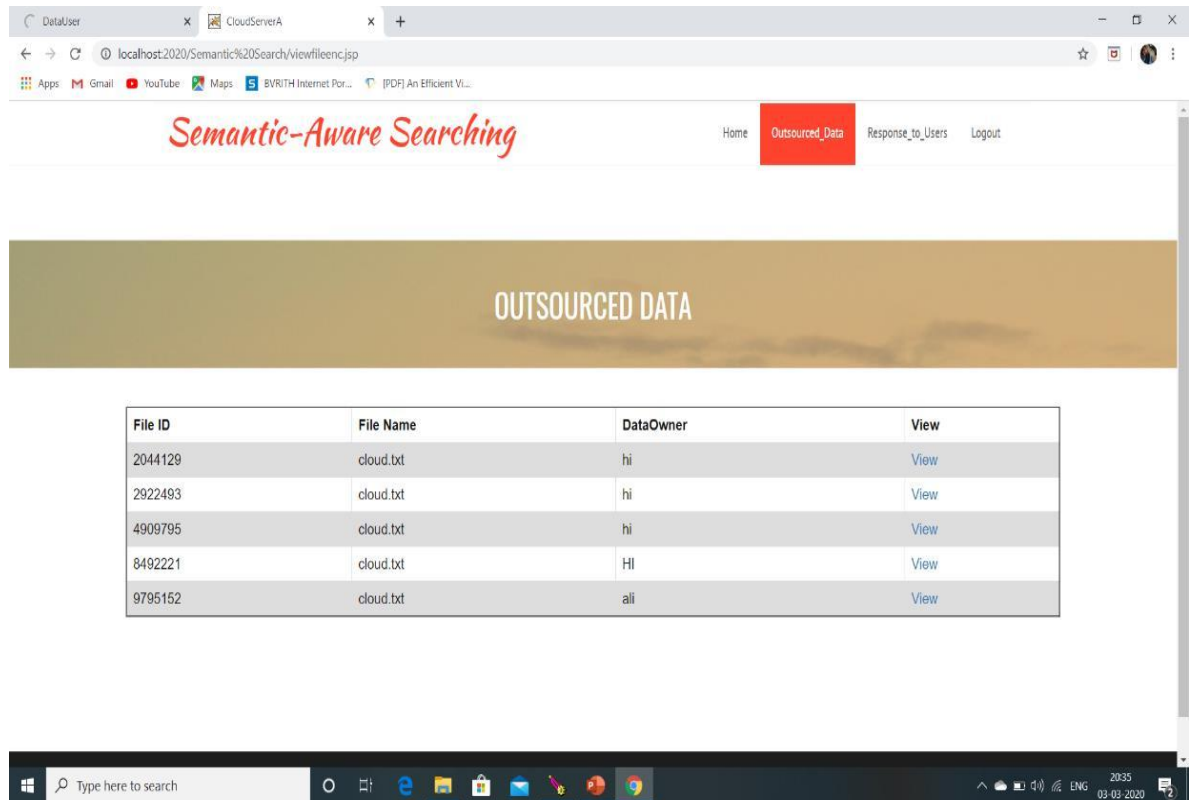


**Fig 4.14 Trapdoors in Cloud server B**

Once the user searches for the file using keywords, the keywords are sent to cloud server B, where Trapdoors are used to find the most relevant files as shown in Fig 4.14.

### Cloud server A-Outsourced Data:

In cloud server A, one of the functions is to view the outsourced data, which can be sent to data user as per requirements. It shows the file ID, File name and Dataowner name as the details of the files retrieved.



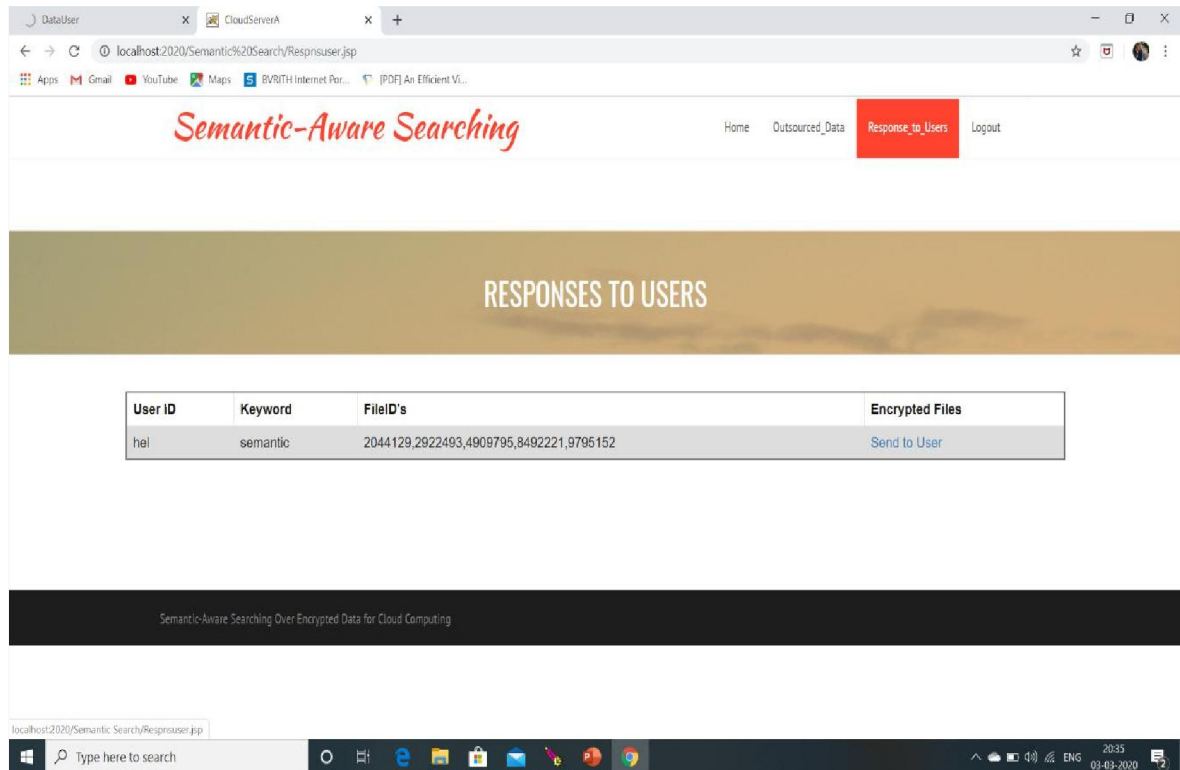
File ID	File Name	DataOwner	View
2044129	cloud.txt	hi	<a href="#">View</a>
2922493	cloud.txt	hi	<a href="#">View</a>
4909795	cloud.txt	hi	<a href="#">View</a>
8492221	cloud.txt	HI	<a href="#">View</a>
9795152	cloud.txt	ali	<a href="#">View</a>

**Fig 4.15 OutSourced Data in cloud server A**

As get results is clicked in the trapdoors webpage, the most relevant files are generated to Cloud server A, where Outsourced Data shows the files as shown in Fig 4.15.

**Cloud server A-Response to users:**

In cloud server A, another function is to respond to the users by sending the file that is asked by the user but in encrypted form. It provides user ID, keyword, file id and encrypted file as details of each file.



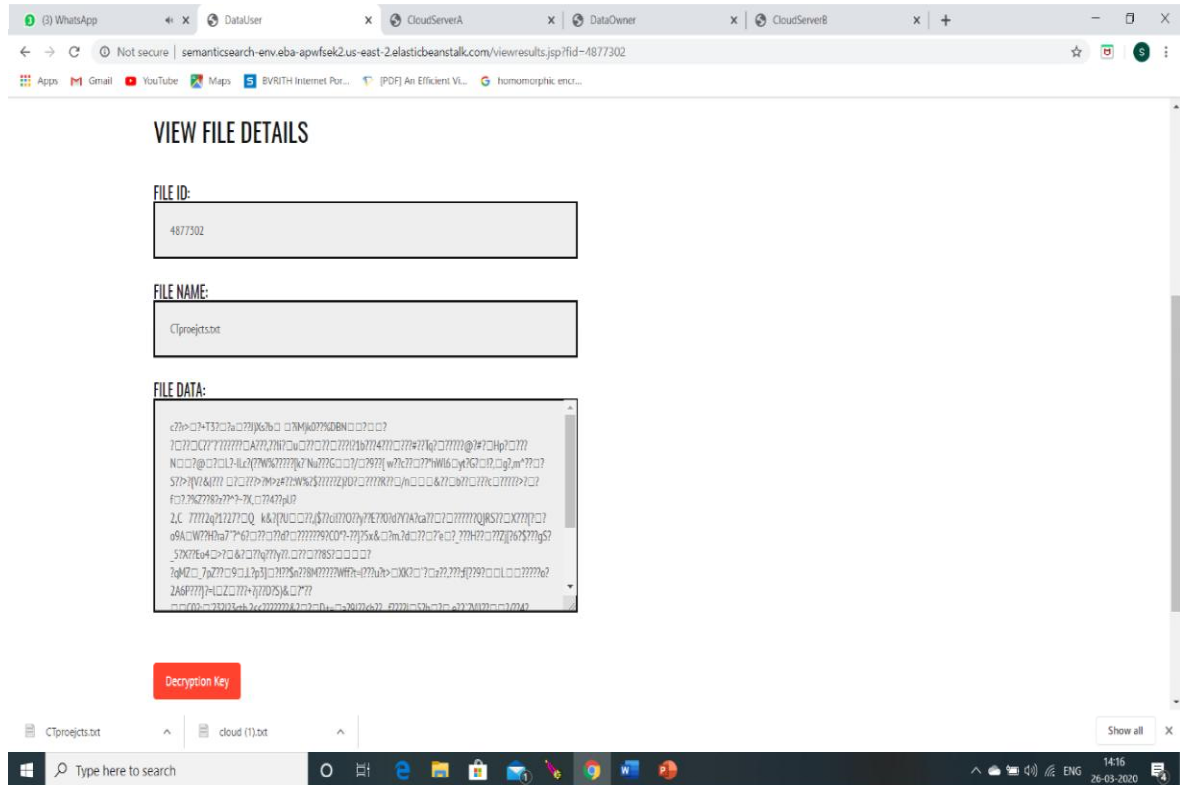
**Fig 4.16 Response To Users in Cloud Server A.**

In cloud server A, as the user opens the response\_to\_Users shown in Fig 4.16, the relevant encrypted file the datauser asked for is shown and can be sent to the datauser.



**Data User-Decrypt Data:**

After this, the data user loads the page and retrieves the file what was asked in the encrypted form where file ID, file name, file data are shown and a button to receive the symmetric key is given.

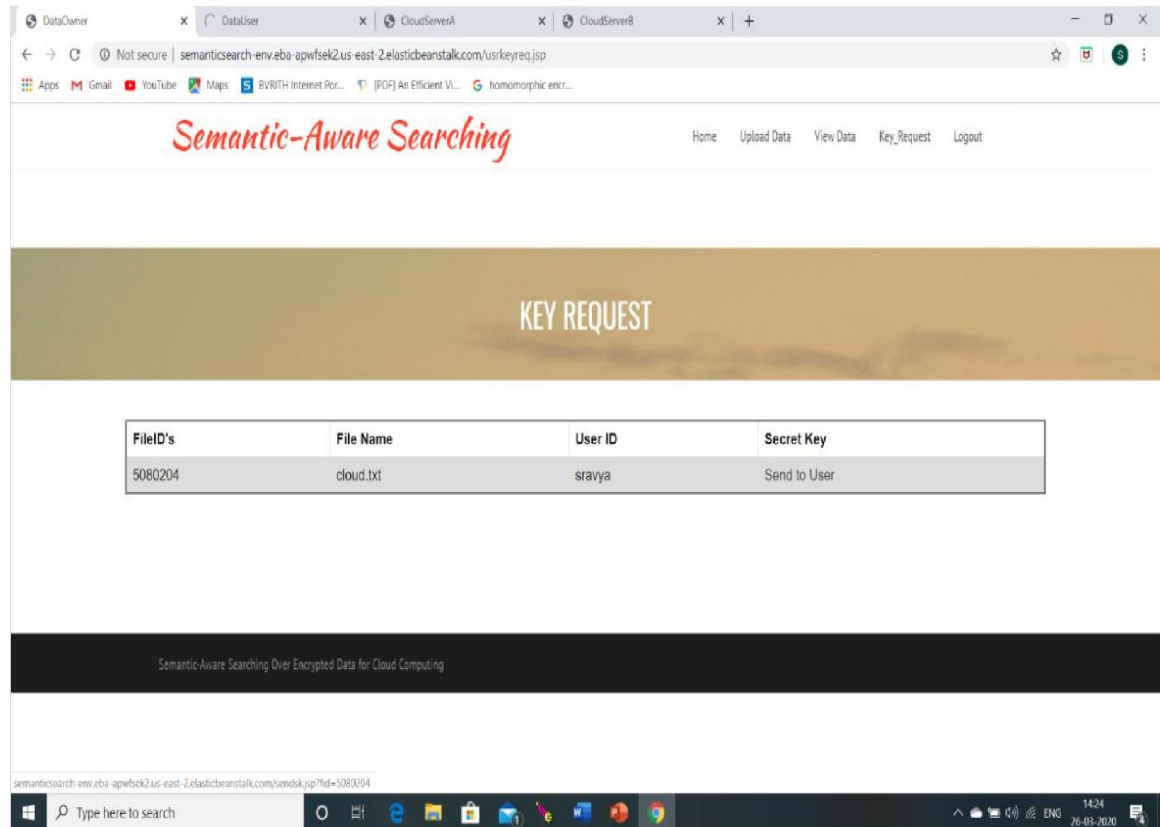


**Fig 4.17 Decryption key generation in DataUser.**

The file received from cloud server A is encrypted as shown in Fig 4.17, To decrypt it Decryption key is requested to the DataOwner.

**Data Owner-Key Request:**

After clicking on the Decryption key button, the data owner receives the request from data user for symmetric key by providing file id, file name, user id, secret key.



**Fig 4.18 Key Request in DataOwner**

The key request function as shown in Fig 4.18, in DataOwner enables the user to send the decryption key for the datauser to decrypt the file.

**Output:**

As the decryption key is sent to the user, the decryption button is clicked in the data user webpage, and the complete file is decrypted including the file id, file name and the file data.

FILE ID:  
5080204

FILE NAME:  
cloud.txt

FILE DATA:  
With the increasing adoption of cloud computing, a growing number of users outsource their datasets to cloud. To preserve privacy, the datasets are usually encrypted before outsourcing. However, the common practice of encryption makes the effective utilization of the data difficult. For example, it is difficult to search the given keywords in encrypted datasets. Many schemes are proposed to make encrypted data searchable based on keywords. However, keyword-based search schemes ignore the semantic representation information of users' retrieval, and cannot completely meet with users search intention. Therefore, how to design a content-based search scheme and make semantic

Download

<< BACK

**Fig 4.19 Download Output file**

After decryption key is sent, the datauser decrypts the file and downloads the file obtained as requested by the user, shown in Fig 4.19.

## 5. CONCLUSION AND FUTURE SCOPE

### 5.1 Conclusion

As an effective schemes based on concept hierarchy is proposed, the problem of semantic retrieval is rectified. Different solutions use two cloud servers for encrypted retrieval and make contributions both on search accuracy and efficiency. To improve accuracy, An extended version of the concept hierarchy to expand the search conditions is used. In addition, a tree-based index structure is constructed to organize all the document index vectors, which are built based on the concept hierarchy for the aspect of search efficiency. The security analysis shows that the proposed scheme is secure in the threat models. Experiments on real world dataset illustrate that our scheme is efficient.

### 5.2 Future Scope

The difficulty of searching of the given keywords in encrypted datasets is solved and can be aimed to implement a search scheme which is more accurate and efficient in a homomorphically encrypted cloud environment.

## REFERENCES

- [1] Zhangjie Fu, Lili Xia, Xingming Sun, Alex X. Liu, Guowu Xie, “Semantic-aware Searching over Encrypted Data for Cloud Computing”, IEEE Transactions on Information Forensics and Security, 2019.
- [2] Z. Fu, X. Sun, S. Ji, and G. Xie,” Towards efficient content-aware search over encrypted outsourced data in cloud,” Proc. of IEEE INFOCOM 2016, pp.1-9,2018.
- [3] J. Li, J. Li, and X. Chen,” Identity-based encryption with outsourced revocation in cloud computing,” Computers, IEEE Transactions on, vol.64, no.2, pp.425437,2017.
- [4] Z. Fu, F. Huang, K. Ren, J. Weng, and C. Wang.” Privacy preserving smart semantic search based on conceptual graphs over encrypted outsourced data,” IEEE Transactions on Information Forensics and Security, vol.12, no.8, pp.1874-1884,2018.
- [5] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, “Fuzzy keyword search over encrypted data in cloud computing,” in Proc. of IEEE INFOCOM’10 MiniConference, San Diego, CA, USA, March 2016, pp. 1–5.
- [6] L. M. Vaquero, L. Roderio-Merino, J. Caceres, and M. Lindner, “A break in the clouds: towards a cloud definition,” ACM SIGCOMM Comput. Commun. Rev., vol. 39, no. 1, pp. 50–55, 2015.
- [7] W. K. Wong, D. W. Cheung, B. Kao, and N. Mamoulis, “Secure knn computation on encrypted databases,” in Proc. of SIGMOD, 2015, pp. 139–152
- [8] N. Cao, C. Wang, and M. Li,” Privacy-preserving multi-keyword ranked search over encrypted cloud data,” Parallel and Distributed Systems, IEEE Transactions on, vol.25, no.1, pp.222-233,2014.
- [9] J. Wang, X. Chen, X. Huang, I. You, and Y. Xiang, “Verifiable auditing for outsourced database in cloud computing,” IEEE Trans. Comput., vol. 64, no. 11, pp. 3293–3303, Nov. 2015.
- [10] F. Cheng, Q. Wang, and Q. Zhang, “Highly efficient indexing for privacy-preserving multi-keyword query over encrypted cloud data,” in Proc. Int. Conf. Web-Age Inf. Manage., 2014, pp. 348–359.
- [11] Z. Fu, F. Huang, K. Ren, J. Weng, and C. Wang, “Privacy-preserving smart semantic search based on conceptual graphs over encrypted outsourced data,” IEEE Trans. Inf. Forensics Security, vol. 12, no. 8, pp. 1874–1884, Aug. 2017.

- [12] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, “New algorithms for secure outsourcing of modular exponentiations,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 9, pp. 2386–2396, Sep. 2017.
- [13] Z. Fu, X. Wu, Q. Wang, and K. Ren, “Enabling central keyword-based semantic extension search over encrypted outsourced data,” *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 12, pp. 2986–2997, Dec. 2017.
- [14] J. Li, J. Li, and X. Chen, “Identity-based encryption with outsourced revocation in cloud computing,” *IEEE Trans. Comput.*, vol. 64, no. 2, pp. 425–437, Feb. 2015.
- [15] R. Chen, Y. Mu, G. Yang, F. Guo, and X. Wang, “Dual-server public-key encryption with keyword search for secure cloud storage,” *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 4, pp. 789–798, Apr. 2016.
- [16] P. Li, J. Li, Z. Huang, C.-Z. Gao, W.-B. Chen, and K. Chen, “Privacy-preserving outsourced classification in cloud computing,” *Cluster Comput.*, vol. 20, no. 1, pp. 1–10, Jan. 2017, doi: 10.1007/s10586-017-0849-9.
- [17] J. Li, X. Chen, F. Xhafa, and L. Barolli, “Secure deduplication storage systems supporting keyword search,” *J. Comput. Syst. Sci.*, vol. 81, no. 8, pp. 1532–1541, 2015.
- [18] A. Kiayias, O. Oksuz, A. Russell, Q. Tang, and B. Wang, “Efficient encrypted keyword search for multi-user data sharing,” in *Proc. Eur. Symp. Res. Comput. Secur.*, 2016, pp. 173–195.
- [19] Z. Fu, X. Sun, S. Ji, and G. Xie, “Towards efficient content-aware search over encrypted outsourced data in cloud,” in *Proc. IEEE INFOCOM*, Apr. 2016, pp. 1–9.
- [20] J. Li, Y. Zhang, X. Chen, and Y. Xiang, “Secure attribute-based data sharing for resource-limited users in cloud computing,” *Comput. Secur.*, vol. 72, pp. 1–12, Jan. 2018, doi: 10.1016/j.cose.2017.08.007.