Contents

`SL No.	Topic	Page No.
1.	Abstract	
2.	Introduction	
2.1	Overview	
2.2	Related Works	
2.3	Objective	
2.4	Methodology	
3.	Discussion	
3.1.	Features	
3.2.	Specification	
3.3.	Results	
3.4.	Conclusion	
3.5.	References	

Abstract

With data being stored more in the cloud, we start worrying more about data theft and data getting leaked. Alongside this we also need to make sure we have efficiency when executing the queries to fetch the documents. We cannot leave the cloud storage as vulnerable therefore, best practices are to encrypt the data that is sent to the virtual storages. However, because of this we end up having a trade-off between computation time and security. Having a method of easy encryption and storing documents in an isolated manner is an easy solution to this problem and yield fast throughput and a decent security. That is the aim in this project, to develop a fast efficient search technique for files in the stored in the cloud.

Introduction

Overview

This report discusses about the work done in developing an efficient privacy preserving ranked search method. Large organizations that generate huge amounts of data prefer to store it on cloud storages. For a data center managing that cloud, it faces multiple challenges. The challenges include but are not limited to scaling the data, protecting and securing the data, efficient storage and retrieval and others. When companies give their data, there is always a fear that the information stored may get leaked, and to reduce the data theft cloud companies prefer encrypting the data. Although encryption to a certain extent does prevent information leakage, it increases the latency in document retrieval because of the time taken due to encrypt and decrypt the documents. Also, when the data in the document is encrypted, the information stored in the document becomes untraceable as well. To elaborate, if the are some documents that had sports as a common identifier, they could be displayed when a query for sports is sent. However, by encrypting the data of the documents as well, this ranking becomes difficult and finding similarities among documents is not possible. Aside from this, with rise in amount of data, the amount of time taken to retrieve data increases exponentially.

Related Works

Cloud companies deal with many kinds of services. Along with those services they also provide storage as virtual storage. Clients usually are careful to storing delicate information on these virtual platforms. Although, cloud companies assure they provide protection and security to the data stored, there are many cases of breach and leakage of information. Having to deal with all this makes it tedious, also with rise in the number of companies and the data stored, storing and retrieving the data becomes onerous. The retrieval and storing time increases exponentially with linear rise in data stored. Having efficient algorithms that can linearly retrieve the data becomes important, especially for real time systems. However, the time increases when data sent has to be encrypted and decrypted. Keeping these ideas in mind, it is suitable to design algorithms and techniques to make this process easier and more reliable from breach yet still keep it fast with increasing data.

As far as the Enrypting and Decrypting is concerned, visual cryptography is a better way to secure the data. From this paper “Visual Cryptography Scheme Based on Substitution Cipher”, we see that Visual Cryptography is fundamentally a cryptographic technique in which decryption is performed by human visual framework.[7] novel visual cryptography conspire in view of a substitution cipher and irregular framework. The plan utilizes two-overlay encryption. In the main crease of encryption, Caeser cipher is utilized to scramble the image push savvy and after that section insightful utilizing a key of the size equivalent to the best regular devisor of the quantity of lines and segments in the secret image.[7] At that point an arbitrary framework is produced and the changed secret image is XORed with this irregular lattice to improve the security. The plan is appeared to be secure and decryption is additionally lossless.

The fundamental thought was to create image shares from a given image so that the shares seem meaningless, yet stacking a predefined number of shares could reveal the image.[3] Visual cryptography is somewhat more worthwhile for execution, contrasted with regular cryptography, since the decryption procedure does not require any computation. Further, the image based data turns out to be more secure, since just the expected beneficiary can uncover the genuine importance of the decoded image.[4] Visual cryptography is utilized in various courses for various applications. Number of image shares help in improving the security. Likewise, every pixel is additionally coded into in excess of one pixel in a portion of the visual encryption plans. This is known as pixel extension and is utilized to build the security.[7] For data hiding, a cover image is utilized and information bits are inserted in the pixel bits. Generally, visual cryptography plans are relatively difficult to crack.

An assortment of plans have been proposed over the most recent two decades on visual secret sharing. The greater part of the exploration work is centered around limiting pixel development, differentiate upgrade, Boolean task based plans, multi-secret sharing, reversible data hiding plans,

probabilistic visual secret sharing plans, shading image cryptography and data hiding. In data hiding plans, the image is utilized as a bearer for the data and it is covered up so that image seems, by all accounts, to be unaltered.[7]The image is called cover image or stego image and the procedure of data hiding in an image is additionally named as steganography. Be that as it may, if the data is implanted in an image, it causes image distortion. Huge plans have been proposed for effective data hiding in images with the principle point of limiting the image mutilation, while keeping the security of the installed data. Substitution Cipher is a procedure that changes a given content into pointless data bits.There are many cryptographic substitution algorithms like Caeser Cipher, Monoalphabetic Cipher, Playfair Cipher, Hill Cipher, Polyalphabetic Ciphers, One Time Pad etc.[7]

Another better new secure technique for cryptography is using Combination of Substitution & Transposition Cipher.A new cipher technique which consists of the three step process-substitution, transposition and substitution again. Using this technique, the plain text can be converted into the cipher text which is the combination of various symbols defined in a given table that in turn makes the plain text unpredictable.

From this paper "A New Algorithm Combining Substitution & Transposition Cipher Techniques for Secure Communication" we can say that Broadly, cipher techniques are divided into two types: Substitution Cipher and Transposition Cipher.[1] Substitution cipher is a method in which units of plaintext are replaced or substituted by different characters. Whereas in Transposition cipher, the position of characters of plaintext are shifted or permuted according to some format or algorithm. The blend of both techniques in a single algorithm have high chances of producing a secret encrypted message

The algorithm that we have proposed has used a series of both substitution and transposition ciphers in the process of encryption and decryption. Our algorithm consists of three steps: Step 1. Substitution Step 2. Transposition Step 3. Substitution.[1]

In "Enabling Efficient Verifiable Fuzzy Keyword Search over Encrypted Data in Cloud Computing" by XINRUI GE describes about the keyword search in the cloud encrypted data. Searchable encryption can support data user to selectively retrieve the cipher documents over scrambled cloud data by keyword-based search.[2]The majority of the current searchable encryption conspires just spotlight on the correct keyword search. At the point when data user makes spelling blunders, these plans neglect to restore the consequence of intrigue. In searchable encryption, the cloud server may restore the invalid outcome to data client for sparing the computation cost or different reasons. In this manner, these correct keyword search plans find minimal down to earth significance in genuine applications. Keeping in mind the end goal to address these issues, we propose a novel verifiable fuzzy keyword search conspire over scrambled cloud data.[2] Verifiable correct keyword search plan, stretch out this plan to the fuzzy keyword search scheme. In the fuzzy keyword search scheme, we employ the linked list a secure index to achieve the efficient storage. We construct a linked list with three nodes for each exact keyword

and produce a fuzzy keyword set for it. To lessen the computation cost and the storage space, we produce one index vector for each fuzzy keyword set, instead of each fuzzy keyword. To oppose pernicious practices of the cloud server, we create a confirmation name for each fuzzy keyword to check the credibility of the returned ciphertexts.[4] Through security investigation and analysis assessment, we demonstrate that our proposed plans are secure and efficient.

At the point when the data user makes spelling mistakes, the conventional correct keyword search schemes can't restore the records of intrigue. It extraordinarily influences the framework usability. To address this issue, Li et al.[8] firstly proposed a fuzzy keyword search conspire over encoded cloud data utilizing the alter distance to gauge the closeness of two keywords. This plan receives the special case innovation to build the fuzzy keyword sets. Kuzu et al.[9] proposed a similitude searchable symmetric encryption plot, which makes utilization of minhash in view of Jaccard distance to help fault tolerant keyword search. Wang et al.[10] proposed the first multi-keyword fuzzy keyword search plot in light of the Bloom filter and the LSH function, in which

the keyword is changed into a bi-gram set. Wang et al.[11] proposed a range fuzzy keyword search conspire by utilizing a score table structure to rank the archives. Fu et al.[12] proposed a multi-keyword fuzzy ranked search scheme based on scheme[10]. This plan outlines another strategy for keyword change in view of the uni-gram, and considers the keyword weight while choosing a sufficient coordinating file set.

An efficient privacy-preserving search method over encrypted cloud data that uses minhash capacities. The greater part of the work in writing can just help a solitary element search in questions which decreases the viability. One of the primary points of interest of our proposed method is the capacity of multi keyword search in a solitary question.[5] The proposed method is demonstrated to fulfill versatile semantic security definition. We likewise consolidate a successful positioning capacity that depends on term frequency inverse document frequency (tf-idf) estimations of keyword document sets. Our investigation exhibits that the proposed plot is turned out to be privacy-preserving, efficient and compelling.

The issue of privacy-preserving keyword search is tended to by different work in writing. Related work can be dissected in two noteworthy gatherings: single keyword and multi keyword search. While the client can search for a solitary element for every query in the previous, the last empowers search for a combination of a few keywords in a solitary query.[3] The arrangement approach is to figure a safe ordering tag of the data by applying bucketization (a bland type of data apportioning) which keeps the server from learning definite qualities yet at the same time enables it to check if a record fulfills the query predicate. Queries are assessed in an estimated way where the returned set of records may contain some false-positives.[3] These records at that point should be removed by the customer which involves the computational overhead of our plan.

We build up a bucketization strategy for noting multidimensional range queries on multidimensional data.[4] For a given bucketization plot we infer cost and exposure chance measurements that gauge customer's computational overhead and revelation chance individually. Given a multidimensional dataset, its bucketization is acted like a streamlining issue where the objective is to limit the danger of divulgence while keeping query cost (customer's computational overhead) underneath a specific client indicated edge esteem. We give a tunable data bucketization algorithm that enables the data proprietor to control the tradeoff between exposure hazard and cost. We likewise consider the tradeoff attributes through a broad arrangement of trials on genuine and manufactured data.

Objective

There are two goals we aim to achieve.

1. To make a model that can take in data and store it efficiently. The data stored should be able to encrypt the data securely. Should be able to rank the pages and should be able to call the pages when a query is past based on relevance.
2. The second is to make a simple search method that takes the query and uses it to find all the relevant documents pertaining to the company that stored it, and to also display those documents that are relevant to the query given.

Methodology

To implement the above goals the following methodology needs to be followed:

1. Creating the design model for the requirements
2. Deciding on the encryption technique to follow
3. Developing a method to rank the encrypted documents
4. Having the documents stored as some easy cluster
5. Designing query tokenizer and efficient retrieval
6. Integration the design and develop the code

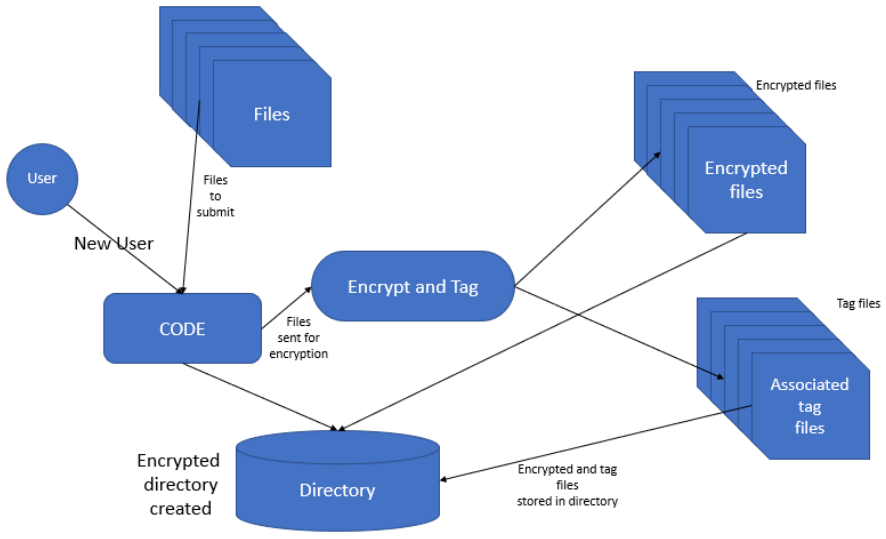
The Idea is to create a directory specific to the user which is associated to the files uploaded by the user. For simplicity a simple search can be used to find the directory however, in complex scenarios the same can be reduced using a hash look up. After directory is created the files get stored in the directory as encrypted data. While encrypting the contents of the files, an associated tag file will get created. The job of the tag file is to hold the associated keywords related to the content file. This should improve the searching efficiency.

Discussion

The algorithm is a simple cypher approach. It is described below with the following process flow diagram.

The new user on registering gets a directory associated. The files to be submitted are then given. Every file given goes through a cypher. So, the contents of the files are now encrypted. These files are now in

two forms, tag and encrypted content. When a look up needs to be done, we refer the tag files and do a similarity check. If the keywords are present in the tag files then it is selected. Based on the number of occurrences the files are sorted and displayed.



Features

1. Easy usability
2. Tag based searching
3. Isolation of unrelated actions
4. Straight forward
5. Should be fast
6. Should save time in search while returning relevant data
7. Localized data content

The aim is to create a model that can be set as such so that the UI acts only as an interface to point to the data to store, or a simple search page that can retrieve documents and display it. The idea is to make as user-friendly as possible.

Specification

There are two separate parts in the model. The first is to store the data as storage. The second is to be able to search for the documents if uploaded. Since we are focusing on only user specific documents, the user information must be known.

This is done at time of storing the data. So is a user is storing the data, some information such as company name or username which is unique must be provided. Following this, the data to be stored is submitted.

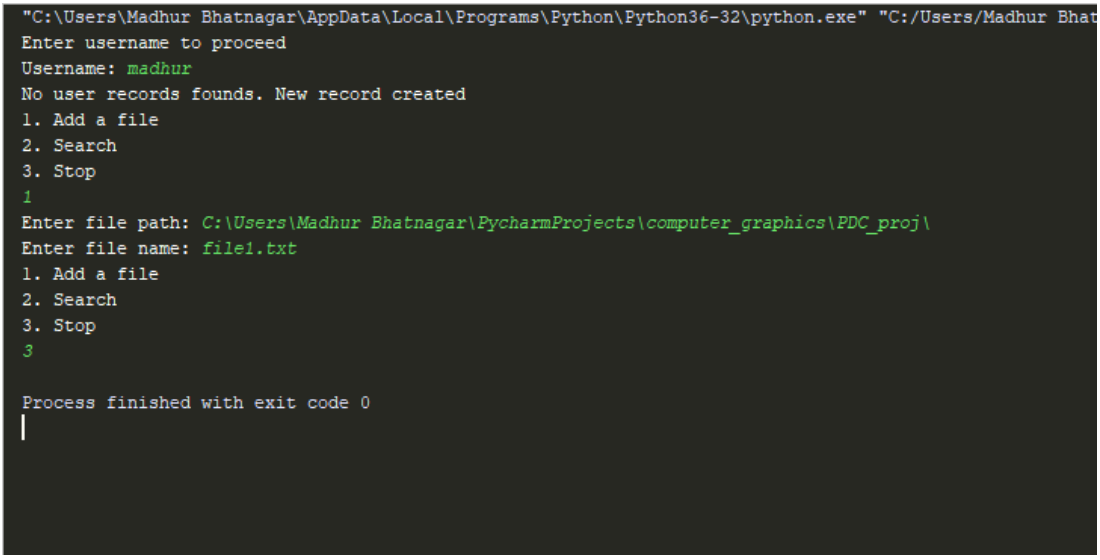
While searching, the unique ID must be known as well and should be given as well while searching. This ID is unique to the company. After the ID is known the data searched should be entered as a query. This makes clustering and retrieval easier. The user does not know the encryption or decryption pattern nor the ranking methods used. The user should see list of relevant documents associated to the query.

Results

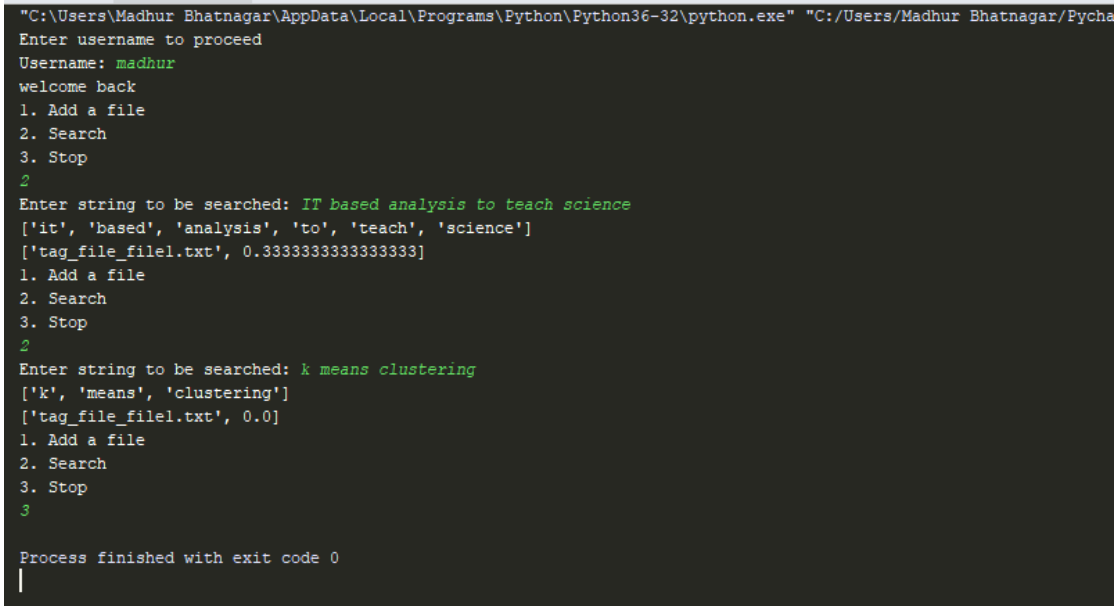
On running the code, we get the following outputs, demonstrated by screenshots. As mentioned there are two parts, first is allocation and the second is the search.

The Code for all is given below the screenshots.

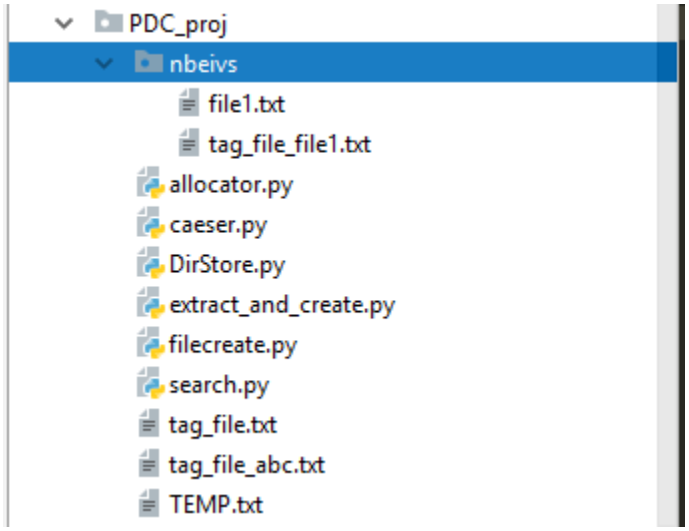
Allocation



Search



It can be seen that the new files are created in an encrypted document.



Source Codes

Main Code

```
import allocator
import caeser
import extract_and_create
import os
import search

#Login
Auth = None

print("Enter username to proceed")
usname = input("Username: ")
dirname = allocator.crypt(usname)
path = os.getcwd()
dirs = os.listdir(path)

if dirname in dirs:
    print("welcome back")
    Auth = 1
else:
    print("No user records founds. New record created")
    os.mkdir(dirname)
    Auth = 1



def Addfile(dirname, path,name):
    os.rename(path+"/"+name, os.getcwd()+"/"+dirname+"/"+name)
    npath = os.getcwd()+"/"+dirname
```

```
    extract_and_create.extract(dirname,npath,name)
    data = []
    f = open(npath+"/"+name,"r")
    for i in f:
        data.append(caeser.getTranslatedMessage('e',i,len(dirname)))
    f.close()
    f = open(npath+"/"+name,"w")
    for i in data:
        f.write(i)
    f.close()

if Auth == 1:
    while Auth == 1:
        print("1. Add a file\n2. Search\n3. Stop")
        option = int(input())
        if option == 3:
            break
        if option == 1:
            path = input("Enter file path: ")
            name = input("Enter file name: ")
            Addfile(dirname,path,name)
        if option == 2:
            search.search(os.getcwd()+"/"+dirname)
```

Allocator

```
def crypt(a):
    length = len(a)
    encrypt = []
    for i in a:
        i = i.lower()
        encrypt.append(chr((ord(i)-97+1%26)+97))
    return "".join(encrypt);
```

Caeser

```
MAX_KEY_SIZE = 26

def getKey():
     key = 0
     while True:
         print('Enter the key number (1-%s)' % (MAX_KEY_SIZE))
         key = int(input())
         if (key >= 1 and key <= MAX_KEY_SIZE):
             return key



def getTranslatedMessage(mode, message, key):
     if mode[0] == 'd':
         key = -key
```

```
    translated = ''
    for symbol in message:
        if symbol.isalpha() or symbol.isdigit():
            if symbol.isdigit():
                symbol=str(symbol)
            num = ord(symbol)
            num += key
            if symbol.isupper():
                if num > ord('Z'):
                    num -= 26

                elif num < ord('A'):
                    num += 26

            elif symbol.islower():
                if num > ord('z'):
                    num -= 26

                elif num < ord('a'):
                    num += 26
            translated += chr(num)

        else:
            translated += symbol
    return translated
```

Extract and Create

```
def extract(dirname,path,name):
    f = open(path+"/"+name, 'r')
    data = None
    for x in f:
        data = str.lower(x)

    data = str.split(x)
    stem = ['in', 'to', 'for', 'and', 'the', 'by', 'of', 'a', 'on', 'an', 'is',
'this', 'as', 'be', 'do', 'these',
            'any', 'which', 'their',
            'there', 'are', 'thus', 'from', 'it', 'at', 'or', 'can', 'that', 'will']
    special = [',', '.', '!', '(', ')', '\'']

    newdata = []
    dataset = set()

    for i in data:
        if i not in stem:
            newdata.append(str.lower(i))
    for i in newdata:
        set.add(dataset, i)

    data = ""

    for i in newdata:
        data += i

    f.close()
```

```
    # to create tag file

    f = open(dirname+"/"+"tag_file_"+name, "w")
    for i in dataset:
        sen = str(i) + " " + str(data.count(i)) + "\n"
        f.write(sen)

    f.close()
```

Search

```
import os

def find(keylist, wordData, wordCounts):
    length = len(keylist)

    finCount = 0
    for i in keylist:
        index = None
        if i in wordData:
            index = wordData.index(i)
            finCount += (int(wordCounts[index])*1.0)/(length*1.0)
    return finCount/(length*1.0)


def search(path):
    a = input("Enter string to be searched: ")
    a = str.lower(a);
    keywords = str.split(a)
    print(keywords)
    dirs = os.listdir(path)
    tag = []
    tagdata = []
    result = []

    for name in dirs:
        tagdata = []
        if name.endswith(".txt") and "tag" in name:
            f = open(path+"/"+name, 'r')
            for x in f:
                sen = str.lower(x)
                sen = str.replace(sen, "\n", "")
                tagdata.append(sen)

            wordData = []
            wordCount = []
            for words in tagdata:
                word = words.split(" ")
                wordData.append(word[0])
                wordCount.append(word[1])

            result.append(name)
            result.append(find(keywords, wordData, wordCount))
            f.close()
    print(result)
    return result
```

Conclusion

Therefore, we see the results of both Allocation and Searching. Allocation creates a new folder for those that do not have a directory and welcomes the user back users who directories are created. It takes the file path and the file name and stores them in the directory of the user. While storing the file in the directory and the content of the file is encrypted and the corresponding tag file is created. Also, we have the search method, where the current directory is the file to be looked into. The tag files are searched with the keywords given as query. The query is tokenized and searched in the tag. The corresponding occurrences are taken and check and normalized and stored in a list as result. The result list holds the tag files and the relevance to the query. Therefore, we see the two different components in execution. We conclude the method to have a simple encryption to store the files and simple algorithm to search for the corresponding files.

References

[1]Bhargava, U., Sharma, A., Chawla, R., & Thakral, P. (2017). A new algorithm combining substitution & transposition cipher techniques for secure communication. *2017 International Conference on Trends in Electronics and Informatics (ICEI)*. doi:10.1109/icoei.2017.8300777

[2]Fuzzy keyword search over encrypted data in cloud computing. (2018). *International Journal of Recent Trends in Engineering and Research*, *4*(1), 375-379. doi:10.23883/ijrter.2018.4047.q0z6z

[3]Hore, B., Mehrotra, S., Canim, M., & Kantarcioglu, M. (2011). Secure multidimensional range queries over outsourced data. *The VLDB Journal*, *21*(3), 333-358. doi:10.1007/s00778-011-0245-7

[4]Kuzu, M., Islam, M. S., & Kantarcioglu, M. (2012). Efficient Similarity Search over Encrypted Data. *2012 IEEE 28th International Conference on Data Engineering*. doi:10.1109/icde.2012.23

[5]Merlin. N, G. (2017). Fuzzy Based Implementation of Multi - Keyword Ranked Search over Encrypted Cloud Data in Secure Cloud Environment. *International Journal of Scientific Research and Management*. doi:10.18535/ijsrm/v5i12.06

[6]Wang, P., Wang, H., & Pieprzyk, J. (2009). An Efficient Scheme of Common Secure Indices for Conjunctive Keyword-Based Retrieval on Encrypted Data. *Information Security Applications*, 145-159. doi:10.1007/978-3-642-00306-6_11

[7]Yadav, G. S., & Ojha, A. (2013). A novel visual cryptography scheme based on substitution cipher. *2013 IEEE Second International Conference on Image Information Processing (ICIIP-2013)*. doi:10.1109/iciip.2013.6707673

[8] J.Li,Q.Wang,C.Wang,N.Cao,K.RenandW.J.Lou, "Fuzzy keyword search over encrypted data in cloud computing," in International Journal of Engineering ResearchandApplications,vol.4,No.7,pp.1-5,2010.

[9] M. Kuzu, M. S. Islam, and M. Kantarcioglu, "Efficient Similarity Search over Encrypted Data," presented at theIEEEInternationalConferenceonDataEngineering pp. 1156-1167, 2012.

[10] B. Wang, S. C. Yu, W. J. Lou, and Y. T. Hou, "Privacypreserving multi-keyword fuzzy search over encrypted data in the cloud," presented at the INFOCOM, 2014 Proceedings IEEE pp. 2112-2120, 2014.

[11] J. Wang, X. Yu, and M. Zhao, "Privacy-Preserving Ranked Multi-keyword Fuzzy Search on Cloud Encrypted Data Supporting Range Query," in Arabian Journal for Science and Engineering, vol. 40, pp. 23752388, 2015.

[12] Z.J.Fu,X.L.Wu,C.W.Guan,X.M.Sun,andK.Ren, "Toward Efficient Multi-Keyword Fuzzy Search Over Encrypted Outsourced Data With Accuracy Improvement," in IEEE Transactions on Information Forensics and Security, vol. 11, No. 12, pp. 2706-2716, 2017