

N-Gram Fuzzy Keyword Search On Encrypted User Data in Cloud

Saumya Sharma, Amrita Bhagtani, Parth Agarwal, Ankit Mohite

Abstract— In this paper, the different techniques that previously have been used for data storage like one storage server for storing both key and encrypted data, different fuzzy keyword search technique like wildcard based and N (2) gram based technique are discussed how the proposed model with 2 server (security and storage server) and how N (2) gram fuzzy keyword search is better than N(3) gram is explained. We demonstrate encrypting and uploading of user files, downloading decrypted files, performing fuzzy search and ensuring data security per user.

Keywords— Fuzzy search, Data Security, Decrypt Data, Encrypt Data, User Security.

I. INTRODUCTION

In the cloud, the data is stored centrally, and there are various types of data stored in the cloud such as user's files, social accounts, game data, website login, and much more. The cloud is used because it helps the data owners a relief from storing of data at their place, because storing the data on our own side may be fatal some times because of hard-disk failure or any other related problems. So for secure storage and retrieval of information such as user's files and data we need to encrypt the data before that particular data is stored in the cloud.

Traditional searchable encryption schemes allow a user to securely store data over the cloud. Searching over encrypted data through keywords and selectively retrieving files of interest, these techniques support only exact keyword search.

On the other hand, users may make a mistake while typing the name of a file and this may happen very frequently. This significant drawback makes existing techniques unsuitable in Cloud Computing as it greatly affects system usability, rendering user searching

Manuscript received on July 26, 2017.

Saumya Sharma - Thadomal Shahani Engineering College, Mumbai, India (email: sharmasaumya026@gmail.com).

Amrita Bhagtani - Thadomal Shahani Engineering College, Mumbai, India (email: abhagtani29@gmail.com).

Parth Agarwal - Thadomal Shahani Engineering College, Mumbai, India (email: parth_agrawal007@yahoo.com).

Ankit Mohite - Thadomal Shahani Engineering College, Mumbai, India (email: ankit.mohite96@gmail.com)

experiences very frustrating and system efficiency very low.

To overcome this we are using fuzzy keyword search technique in our research. As a result, even if the user while retrieving the file from cloud makes a minor mistake in typing the name of the file then also the desired file is retrieved.

Since the administrator has access to data stored in the cloud, they can unintentionally or intentionally access the client data. Security issues which are of concern to the client can be classified into sensitive data access, data segregation, bug exploitation, recovery, accountability, malicious insiders, and account control issues.

Traditionally the encrypted files along with the key of encryption are stored on the same server. Even if the people running such a service are entirely trustworthy, the encrypted data can be easily compromised when a private key is accessible by others than yourself alone.

In such a scenario the previously used encryption is of no use because others now can access all private keys stored on the server to decrypt the data.

The central server that holds the keys to thousands or even millions of users is of much greater interest to hackers than a personal computer that holds only one or two private keys. Thus, the effort they will put into attacking such a server will also be a lot higher. Several employees can access these keys. The employees, however, are just people. They can be bribed or blackmailed to turn over private keys, which makes it even easier for hackers or third party agencies to gain access.

To overcome all above problems we store encrypted data and key on different servers in the cloud. As a result, even if one server is hacked by the attacker then security won't be affected. Suppose attacker gets access to keys stored on security server he can't do anything with the keys because he doesn't have the data. If he gets the encrypted data stored on storage server then also system can't be cracked as hacker don't have the key to decrypt the data. In any situation, security will be achieved.

So two servers will be used here one is the data store server and the other is security server which makes the work of the hackers difficult as keys and data are stored on different servers and so the hackers cannot exploit or violate the user's data because the hacker can retain either the key or the data but not both. The location of file encryption algorithm is C:\xampp\htdocs\fuzzy\encryption. The encrypted files

are stored in user specific folder that is dynamically created whenever a new user sign ups. That location is C:\xampp\htdocs\fuzzy\includes\<user_id>. All these user folders are securely stored within C:\xampp\htdocs\fuzzy\include.

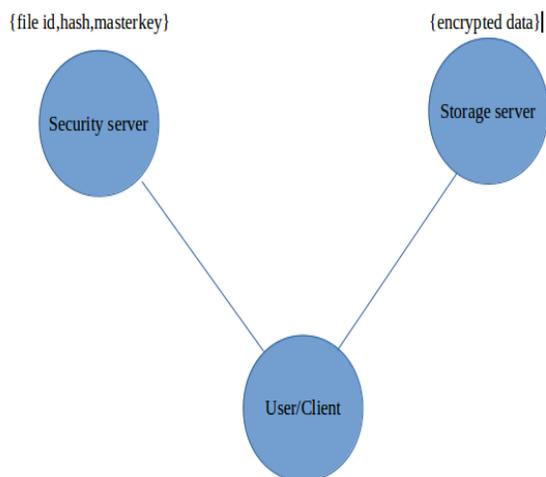


Fig. 1: Architecture of data storage and retrieval

II. BRIEF DESCRIPTION

Generally the cloud services are browser based, therefore any browser enabled device such as for instance laptop, desktop, smart phone, tablets can be used to gain access to these services, the services at providers end may be hosted on any platform, from Windows, Linux, etc., which are accessible via internet. As an example consider a regular income and expenditure application which gives different analysis on expenditure by a person, this application could be executing on cloud providers server, whilst the client browser will allow client to feed in the inputs and visualize the analysis prepared for the inputs provided, these analysis computation is completed at server side. Suppose this application can further create documentation on monthly bases which often can be stored in cloud storage once again relieving the client from storing or processing the file on its side. Because the cloud services are offered via internet, significant factors which play an important role in performance are speed of internet, processing power of the individual. While the cloud providers have server banks, to boost the processing power, multiple server are often used internally by the cloud service provides. This pooling is invisible to the client[1]. On another hand if these heavy tasks were to be executed on client side, it would require investment in hardware, time. Due to cloud, it frees the client from buying expensive hardware and investing his/her valuable time, since time is money [5]. Having studied the overview

of working cloud, let's now understand some of the essential characteristics [6].

1 On-the-fly service:

A consumer can require more capabilities at any movement of time, example processing power for huge task, and these requirement must be accomplished without human intervention and be invisible to client [4].

2 Wide Accessibility:

Generally the cloud service are available via standard network protocols, it promotes different types of clients platforms (like, smart phones, laptops etc) for accessing these services [4].

3 Pooling Of Resources:

The pooling of the resources at cloud providers end is invisible to the end client, and resource assignment is done dynamically depending the need of the client [4].

4 Measured service:

Cloud has enough resources, and amount used by each client is measured by metering capability, and controlled at some level, for optimized resource usage, (like storage).

2.1 Proposed System:

In the proposed design, a hash service data integrity verification, encryption/decryption service, and provision for defining list of people which can access data securely, is provided by a trusted 3rd party which is separate from the storage cloud provider.

2.1.1 Business Model with separate encryption/decryption and hashing service:

The system provides hash, access list, encryption/decryption by a trusted 3rd party over the network in the form of "Software as a Service" (SaaS)[1]. The system has a separate storage service which is also provided as a SaaS. The data storage for each client is done in database in the form of "BLOB". The trusted 3rd party which provides these security services does not store any data at its ends, and stores only master key for each client for data encryption and decryption, and hash of the data which is calculated on client side. To enhance the security, the communication between client and security server is secured using Diffie Hellman key, which is used as a input for AES. This division of responsibility has big effect, as no single provider has access to other data and security key, hash at the same time.

Fig. 1 is an overview of the architecture where storage and encryption/decryption/hash services (security services) are separated. For example (as described in chapter 1, Motivation) a small or medium scale business who wish to store all its account related data in cloud storage, will first calculate the hash of the data, encrypt the data using encryption service and then store the data in storage

provided by separate provider. The system also provides functionality where other users from small scale business Company will be able to access data which is stored in cloud storage. The sessions between client and security server is secured using Base64 as the encryption algorithm.

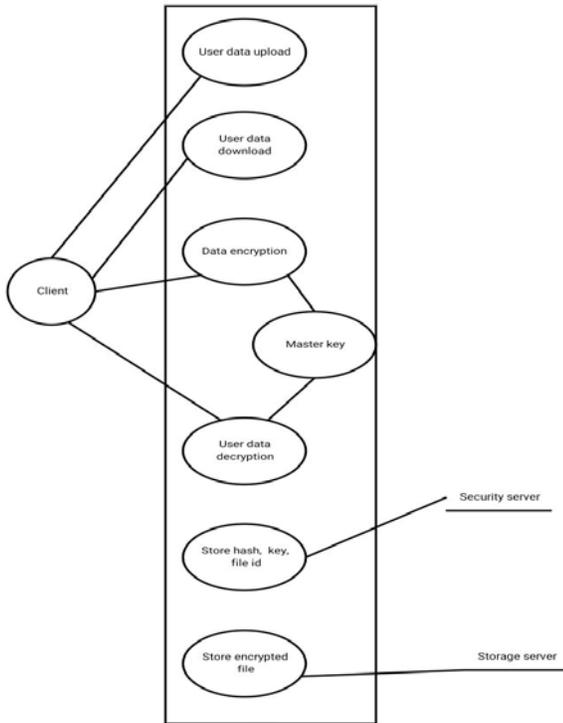


Fig 2. Use Case

2.1.2 Methodology and Analysis:

1. User uploads the data to cloud
2. User downloads data from the cloud
3. Users data is encrypted before uploading
4. A master key is used for encryption of the data
5. Using this master key the data is decrypted when data is downloaded.
6. Security server stores the hash code, master key and file id.
7. Storage server stores the encrypted file.

2.2 Design Approach:

In this section some of the advantages of algorithms used for encryption (i.e. Base64) and used for fuzzy search (N-gram with ranking) are discussed.

2.2.1 Selection of Base64 (B64):

Broadly speaking the encryption/decryption can be done via symmetric key or asymmetric key[9]. In

symmetric algorithms, both parties share the secret key for both encryption/decryption, and from privacy perspective it is important that this key is not compromised, because cascading data will then be compromised. Symmetric encryption/decryption require less power for computation. On the other hand asymmetric algorithms use pairs of keys, of which one key is used for encryption while other key is used for decryption.

Generally the private key is kept secret and generally held with the owner of data or trusted 3rd party for the data, while the public key can be distributed to others for encryption. The secret key can't be obtained from the public key. In our case since the encryption/decryption is performed on trusted 3rd party server, symmetric key is used, and it delegates the burden of key management to the trusted 3rd party. If key management where to be done at clients end it would mean,

1. Either they have to remember the big key
2. Store the key in all devices/machine which will be used to access the cloud services, which make user device a bottleneck.
3. Individual owner has to take the responsibility of sharing the key with specific authorized group of user which he/she define.

While on the other hand using symmetric key encryption the master key or private key usage which

Id	String
1	blue
2	blunder
3	blunt
4	flank
5	flu
6	fluence
7	fluent
8	flunker

an	bl	ce	de	en	er	fl	ke	la	lu	nc	nd	nk	nt	ue	un
4	1	6	2	6	2	4	8	4	1	6	2	4	3	1	2
	2			7	8	5			2			8	7	6	3
	3					6			3					7	8
						7			5						
						8			6						
									7						
									8						

(a)

(b)

Input query string for search : Flunk

fl	lu	un	nk
4	1	2	4
5	2	3	8
6	3	8	
7	5		
8	6		
	7		
	8		

String id	frequency	score
1	1	3
2	2	4
3	2	2
4	2	2
5	2	2
6	2	4
7	2	3
8	4	2

Rank list	
3	blunt
4	flank
5	flu
8	flunker
1	blue
7	fluent
2	blunder
6	fluence

(c)

(d)

(e)

Fig. 3 : (a) List of file name in cloud (b) N(2) gram list of cloud file name (c) N(2) gram list of searched

query (d) frequency and score (e) displayed file rank-wise

would be stored in security cloud provider per user gives the client the advantage like,

1. freedom from remembering any key.
2. Client can use any device/machine to access the data stored in cloud.
3. the client need not worry as to how the data will be shared securely, the client just need to define the individual whom he/she wants to share the data with.

2.2.2 Selection of N-Gram with ranking:

Fuzzy Search: approximate string matching

Eg. Language will be corrected to language

Scenario:

1. User want search keyword language
2. User misspelled it as language and clicked on search button
3. Data in the database is in encrypted form.
4. Now we will try to search the encrypted data for inputted keyword language. Which will converted to language and display result.
5. This is the technique which will help us to match the keyword language with encrypted keywords in the database.

The Fig 3 shows how exactly fuzzy keyword search works:

(a) File names like blue, blunder etc. are stored in the cloud.

(b) The N(2) gram list of the files names are formed and each gram forms an index. For example blue is broken as bl,lu,ue and each of it works as an index. Similarly fluent is broken as fl,le,ue,en,nt. With each index all the associated file id are stored. For example for bl index, file id 1 2 and 3 contains bl so we write 1,2 and 3 in the list having bl as index.

Now the search string is given as input.

(c) This contains the N(2) gram list of the search query string with the file ids that match with the grams. For example in flunk, fl is there in 4,5,6,7 and 8.

(d) This consists of the frequency of the grams that match. For example file id 1 matches only 1 time (fl) so the frequency is 1. Similarly file id 4 has frequency 2 (fl and nk). Score is calculated based on the number of grams that are not matched. So in case of file id 1, total grams to be matched was 4 but only 1 matched so the score is 3.

(e) This displays the list of files in rank order. Lower the score of the file, higher is its rank.

2.2.3 Choosing the value of N:

The reason for choosing N=2 instead of N=3 is because of higher accuracy of ranking of similar words and shorter words. The more the grams of a given word, the more efficient the algorithm will be in providing more accurate results of similar words. For example, consider the word in file as principal, and the input query string for search is principle, then,

For N=2, the grams are :

PRINCIPLE : pr, ri, in, nc, ci, ip, pl, le

PRINCIPAL : pr, ri, in, nc, ci, ip, pa, al

Thus, pairs that are not matching are : 2

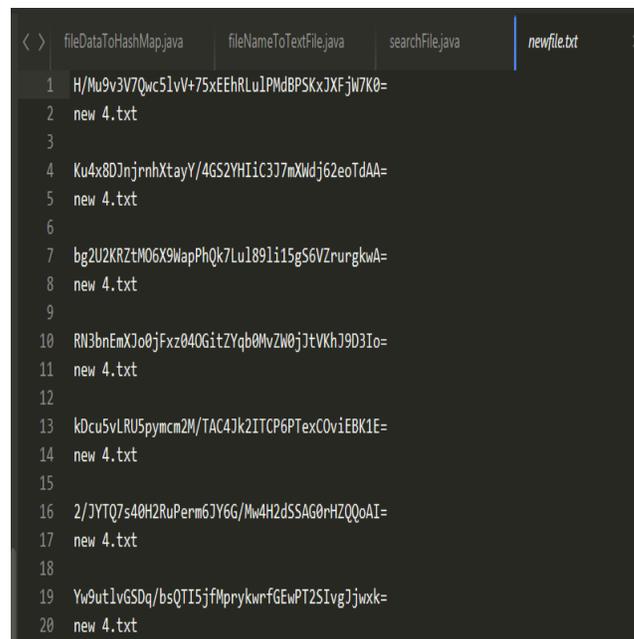
For N=3, the grams are:

PRINCIPLE : pri, inc, cip, ple

PRINCIPAL : pri, inc, cip, pal

Thus, pairs that are not matching are : 1

Since the number of pairs that do not match are greater for N=2 than for N=3, we can conclude, N=2 provides better accuracy of results.



```

fileDataToHashMap.java  fileNameToTextFile.java  searchFile.java  newfile.txt
1 H/Mu9v3V7Qmc51vV+75xEeHRLu1PMdBPSKxJXFjW7K0=
2 new 4.txt
3
4 Ku4x8DjnjrnHxtayY/46S2YHIiC3J7mXWdj62eoTdAA=
5 new 4.txt
6
7 bg2U2KRZtM06X9WapPhQk7Lu1891i15gS6VZrungskwA=
8 new 4.txt
9
10 RN3bnEmX3o0jFxz040GitzYqb0MwZw0jJtVKHj9D3Io=
11 new 4.txt
12
13 kDcu5vLRU5pymcm2M/TAC4Jk2ITCP6PTexCOviEBK1E=
14 new 4.txt
15
16 2/JYTQ7s40H2RuPerm6JY6G/Mw4H2dSSAG0rHZQoAI=
17 new 4.txt
18
19 Yw9utlvGSDq/bsQTI5jfMprykrfGEnPT2SIVgJjwxk=
20 new 4.txt

```

Fig. 4 newFile

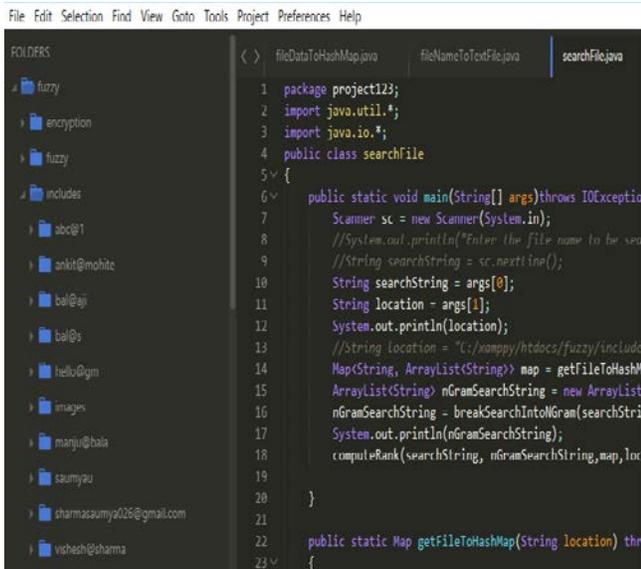


Fig.5 Individual User Folders containing encrypted data

III. RESULTS AND DISCUSSIONS

3.1 Password Encryption

Once the user sign-ups the password is stored in MySQL database in an encrypted format. Encryption algorithm used for encrypting password is MD5. A MySQL query is created in PHP file for storing the data of users in the database.

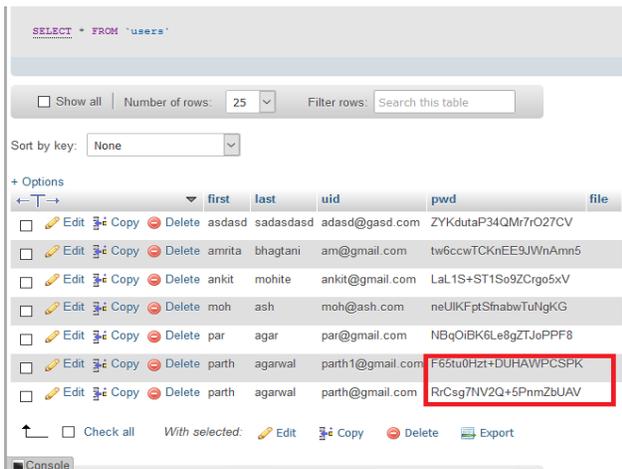


Fig. 6 Encrypted passwords in database

3.2 File Upload

This page consists of browse button for selecting a file for computer and a text box for entering the keywords. There is an upload button through which the file is uploaded

on the cloud. Once the valid user is logged in or a new user comes to site then a new folder is created for that user. All the files uploaded by this user are stored in this folder. The files are in an encrypted format. The encryption algorithm used is Base 64.



Fig. 7 uploading files in database

3.3 File Download

All the files uploaded by the user are displayed on the website. A valid user can access only his folder and not of anyone else. Once the user signs in's he can see all the files in his folder he can use the keyword textbox to get the required file, here fuzzy search is been used. Once the desired file is displayed on the webpage the user can download it using the download button. The is downloaded file is available to user in original i.e. non encrypted form.

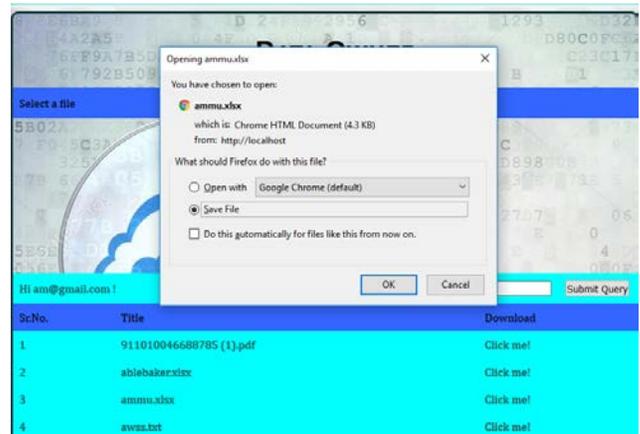


Fig. 8 downloading files from database

3.4 Fuzzy keyword search

User searches for a particular file and the clicks submit to retrieve the results.



Fig. 9(a) Search for a file



Fig. 9(b) Search Results

IV. CONCLUSION

In this work, efficient and secure file uploads and downloads is been implemented. Once the files are uploaded they are saved in an encrypted format. For ensuring user security individual user folders are created as per their user id's. Each file uploaded by the user is saved within that file. Once the files are downloaded they are retrieved in an original non-encrypted format. All the files uploaded by the user are displayed on the webpage. If the number of files is large then the user can narrow down the search results by typing in the keyword in the search textbox. If the user makes a mistake while typing then the search results include the files which are closely related to the keyword entered, this the concept of fuzzy search. We use N-Gram algorithm coded in java for performing fuzzy search in our research.

ACKNOWLEDGEMENT

Any accomplishment requires the efforts of many people and our project is no different.

We would like to thank our internal guide Prof. Shachi Natu for her overwhelming support during the entire education of our project and for being the task-master and the mentor. We would like to express our gratitude towards him for her constant encouragement, support and guidance throughout the project.

It is only right to express our sincere gratitude to our Professors, BE IT Batch of 2017, our classmates whose help and support were instrumental and essential in the development and successful completion of this phase of our project. We also thank the teaching staff and non-teaching staff of the IT Department who have helped us from time to time with necessary resources and otherwise deserve a special mention.

Ultimately, we would like to thank the open-source community, world over, who have taken painstaking efforts and provided us with the useful information available on the Internet.

REFERENCES

- [1] L. Liu, C. Zhang, S. Yao, S. Wang, and W. Zhou, "Fuzzy Keyword search with safe index over Encrypted cloud computing," *TELKOMNIKA Indonesian Journal of Electrical Engineering*, vol. 11, no. 10, Oct. 2013.
- [2] N. Smetanin and V. my complete profile, "Fuzzy string search," 2011. [Online]. Available: <http://ntz-develop.blogspot.in/2011/03/fuzzy-string-search.html>. Accessed: Nov. 15, 2016.
- [3] A. Meharwade and G. A. Patil, "Efficient Keyword search over Encrypted cloud data," *Procedia Computer Science*, vol. 78, pp. 139–145, 2016.
- [4] H. Bast and M. Celikik, "Efficient fuzzy search in large text collections," *ACM Transactions on Information Systems*, vol. 31, no. 2, pp. 1–59, May 2013.
- [5] J. Wang et al., "Efficient verifiable fuzzy keyword search over encrypted data in cloud computing," *Computer Science and Information Systems*, vol. 10, no. 2, pp. 667–684, 2013.
- [6] "Multi-keyword ranked search over Encrypted cloud data supporting synonym query," *International Journal of Science and Research (IJSR)*, vol. 5, no. 6, pp. 2044–2048, Jun. 2016.
- [7] W. Zhou, L. Liu, H. Jing, C. Zhang, S. Yao, and S. Wang, "K-gram based fuzzy Keyword search over Encrypted cloud computing," *Journal of Software Engineering and Applications*, vol. 06, no. 01, pp. 29–32, 2013.
- [8] N. Shekokar, K. Sampat, C. Chandawalla, and J. Shah, "Implementation of fuzzy Keyword search over Encrypted data in cloud computing," *Procedia Computer Science*, vol. 45, pp. 499–505, 2015.
- [9] Pourush, Naresh Sharma, and Manish Bhardwaj. "Enhanced Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data." *American Journal of Networks and Communications* 4.3 (2015): 25-31.