Secure Key Store

A dissertation submitted for the Degree of Master of Information Security

H. M. A. U. Bandara
University of Colombo School of Computing
2019

Declaration

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute. To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name: H. M. A. U. Bandara

Registration Number: 2016MIS003

Index Number:16770039

Signature Date

This is to certify that this thesis is based on the work of Mr. H. M. A. U. Bandara under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:

Supervisor Name: Dr. Kasun De Zoysa

Signature Date

Acknowledgements

I would first like to thank my thesis advisor Dr. Kasun De Zoysa, senior lecturer at University of Colombo School of Computing. The door to Dr. Kasun's office was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this paper to be my own work, but steered me in the right the direction whenever he thought I needed it.

I would also like to thank the experts who were involved in the validation survey for this research project. Dr Manju Sri Wikremasinghe (lecturer at University of Colombo School of Computing), Dr. Kasun Gunawardana (Lecturer at University of Colombo School of Computing). Without their passionate participation and input, the thesis could not have been successfully conducted.

Abstract

One of the most powerful factors in this era is **Information**. You can start the third world war if you have correct information, you can switch ON all the nuclear missiles by one click if you have the correct information, you can make a better world with information. That is the power of information. Therefore "security of information" is a big challenge. Encryption is the most common approach to secure the information. It is one of the best way to protect them. Encryption algorithms need a "Key" in order to encrypt data. Security of encrypted information is depend on the Key. With this approach, information is secured with encryption, but everyone misses one part, **Security of the KEY**. All protected data and information will be disclosed if the key is exposed. Therefore security of the Key is a major concern. Not only the security, sharing and carrying the key are also important factors.

This research contains solutions for below mentioned points.

- Security of the key
- Key carrying
- Key sharing

Methodology of this research ensure that saved data will never be stolen.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivations

In today's digital communication, sharing of information increasing significantly. Therefore the security of transmitted data is one of the most important and most challenging aspect of information security.

Cryptography is one of a main solution for the above mentioned problem. Cryptography has two main methods. Encryption and Decryption. Convert of human readable plain text data into unreadable cipher text is called, encryption. Reverse method of encryption is called decryption. Sender encrypt the message and receiver must decrypt it, in order to read the encrypted message.

There are many data encryption algorithms. All of they can be divided into two groups, Asymmetric key and symmetric key encryption algorithms. Most important thing of these algorithms is the key. Asymmetric key algorithms use two different keys to encryption and decryption. Symmetric key algorithms use only one key for both encryption and decryption. It seems very secure. But the problem is, sensitive information may be revealed to the public if the key is compromised. Because in many cases, encryption algorithms are public. Asymmetric

key algorithms are less vulnerable with key sharing regarding to symmetric key algorithms. It only requires "public key" sharing and it is public. But carrying and protecting the private key is the great problem. Therefore both asymmetric and symmetric algorithms have the same problem in different ways and both requires a secured and reliable way of sharing and protecting the keys.

This research will fulfill above mentioned requirement with a new concept.

1.2 Context of the Study

This study mainly focuses on a method of storing "keys" in cloud with highest security. It will fulfill both aspects "Carrying" and "Protecting" of keys. Specialists have proved that, one of the easiest way of stealing information is, steal from person who carries it. Therefore it is a good practice not to carry your cipher "key" with you. Store it in a secured cloud instead of carrying. Because you can access data anywhere, anytime if the data is in cloud. No need of carrying it. You only need internet connection and a compatible device.

Hashing and encryption are the highly used technologies for this research method. String concatenate and addition are also act a major role here. Hashing and encryption methods used here are easy to understand but combination of new features with these methods make them hard to break. Divide a large portion to several unites and storing them in several places increase the security and complexity. But it is not a bad way to store data on behalf of security perspective. Otherwise one of the main security ethics, confidentiality will not be saved.

1.3 Objectives

Introducing a new way for enhance the security of stored data using cryptography and related techniques, is the main concern of this study. Not only storing, but

also retrieving of the stored data is also act a huge role in practical scenario. Because it is not a good practice if data retrieving has issues.

Data storing and data retrieving with integrity and confidentiality makes this study very difficult. Because integrity and confidentiality must be kept in entire process of storing and retrieving. Otherwise this will be useless.

Chapter 2

State of the Art

Problem discussed in this study is a burning problem and some people have come up with similar kind of research and solutions. Below I have mentioned some research which are slightly similar (I took ideas from those research and systems when I build my research) to this study and there are some solution which were helpful.

2.1 IPFS (Inter Planetary File System)

InterPlanetary File System (IPFS) is a protocol. It designed as a network which store and share content through the network. It is a distributed file sysem. IPFS and BitTorrent swarm share same properties in many ways. Therefore it is bit similar to a BitTorrent swarm . IPFS is a distributed file system and connect all computing devices with same file system. The file system can be accessed in several ways. (Eg: FUSE, https)

IPFS maintain the connection of the data using the hash string. You can retrieve the data if you have the hash. All the data in IPFS is publicly available. No confidential data. IPFS doesn't have secure method of storing data. And also

it stores data with direct link to its hash string. Here, this study introduce a good way of maintain the link between data and hash string in secure manner.

2.2 BitTorrent

BitTorrent (abbreviated BT) is a peer-to-peer (P2P) communication protocol used to transmit data over the Internet. BitTorrent is one of the most popular way for moving large files, such as digital video files, songs etc... It is estimated that by February 2009, peer-to-peer network account accounts for between 40% to 70% of Internet traffic (on-site). [1] In February 2013, BitTorrent accounted for 3.35% of the total Broadband, with more than half the total bandwidth delivered. To send or receive files, a user uses BitTorrent client on a computer connected to the Internet. BitTorrent client is a computer program that uses the BitTorrent protocol. BitTorrent Trackers provides a list of available files for downloading and allows the customer to find users who are calling to send seeds. BitTorrent customers are available on different computer platforms and operating systems. In 2013, there were between 15 to 25 million BitTorrent users at any given time. As of January 2012, 500 million BitTorrents are being used by active users.

BitTorrent has the same file shared among several instances and collect the file from them when it comes to retrieve. This research method also has the files shared among several data stores and retrieve them when request is trigered.

2.3 Peer-to-peer file sharing

Peer-to-peer file sharing is the distribution of digital media using peer-to-peer (P2P) network technology. With P2P file sharing, users can access media files such as books, music, movies, and games using P2P software that searches for

other computers connected to P2P networks to find the content they need. These nodes are end-user computers and server.

P2P sharing technology has been developed through a series of development stages. BitTorrent protocol also uses the same. Microsoft uses it for online games, they use it as a content distribution network to produce large amounts of data, without having to pay extraordinary prices for the inherent bandwidth when offering a single site.

A number of factors have led to acceptance and facilitation of sharing resources with others. This includes increasing internet bandwidth, increasing the body mass index and increasing the accessibility of computers. Users can transfer files one or more times from one computer to another via the Internet through various file transfer applications.

This study is using a technology which is bit similar to P2P file sharing when it comes to data storing. It store the data in several components by dividing the data into parts.

Chapter 3

Methodology

Summary

Prime objective of this study is to introduce a trusted way for storing and retrieving of Keys. Hash and Cryptography are the advanced technical solutions which I mostly used here.

Methodology of this research contain two major parts.

- Data storing
- Data retrieving

Both the above mentioned parts are highly important and both must be there.

3.1 Data storing

User must provide a username and a password. That will be the access point of user's stored data. Next step would be the insertion of the data which user need to secure.

Below steps are the necessary steps for the data storing part.

- Generate 512 bits hash string, using concatenated "username, salt and password".
- Divide the hash string into "n" number of parts. From this moment onwards I get 4 as number "n". It will make this method easy to understand. After dividing hash string in to 4 parts, one part will be 128 bits in size.
- Calculate the ASKII sum of the characters of each part.
- Get the modulo four(4) of the sums separately.
- Encryption of the data which need to be stored.
- Divide the encrypted data into 4 parts.
- Select particular server for store the encrypted data.
- Save Encrypted data with the relevant hash string, at particular database(server).

When a new user try to store(secure) their data in the system, user will be asked for "username and password" first. Then user must insert the data. System will be generated a 512 bits hash string using concatenated "usernam + salt + password" and it will be divided to 4, 128 bits parts for storing purpose.

Data will be prepared in another way for store. Data will be encrypted using "username + password" as the key and encrypted data will be divided into 4 parts similar to the process of hash string dividing. In parallel, modulo four(4) of ASCII sum of the characters of all 4 hash chunks will be calculated to choose which server to store the hash chunk and particular encrypted data chunk. All data will be stored according to alphabetic order of hash chunks.

You can get the overall idea of data storing from below Figure 3.1. It has the architecture of data storing and note that retrieving process also quite similar to the storing process.

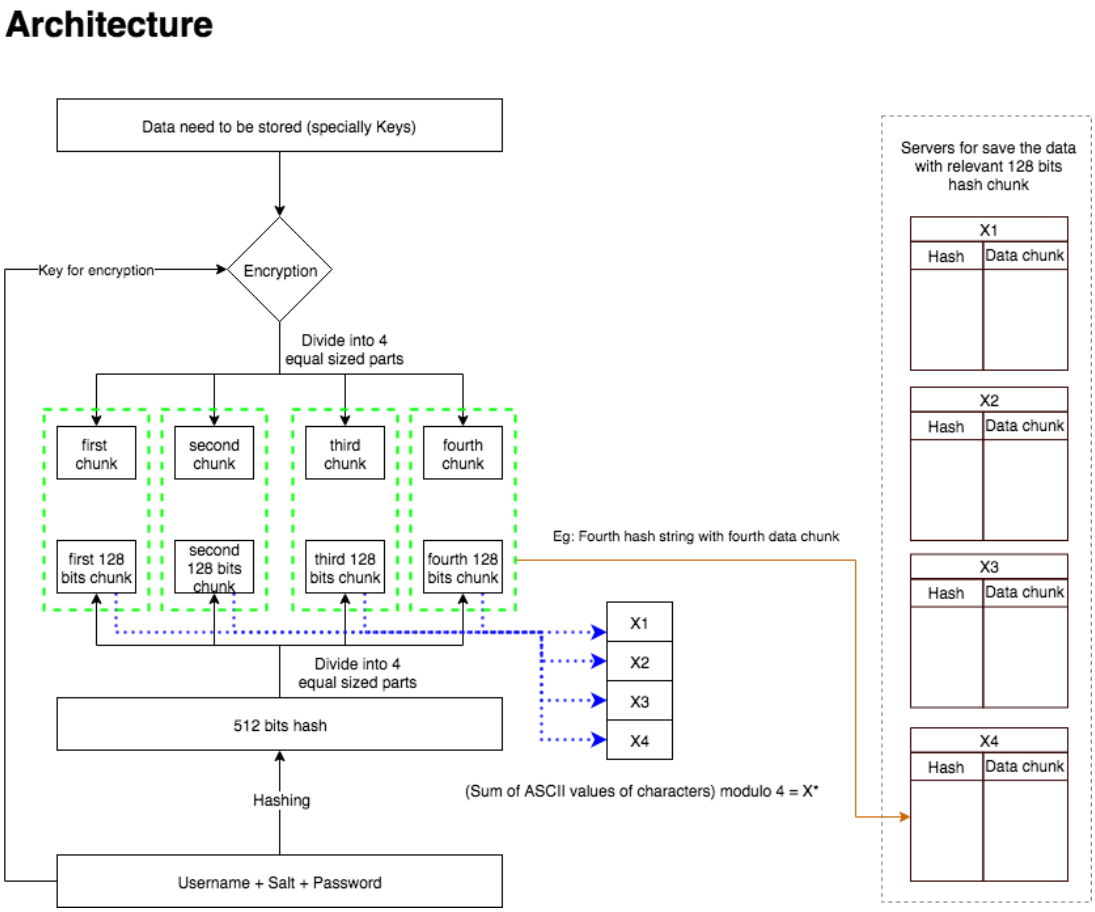

Figure 3.1: Data storing architecture

3.2 Data retrieving

User must give the username and password when retrieving data. Just like data storing, concatenated "username + salt + password" use to generate 512 bits hash and it will be divided into 4 parts. Then generate the modulo 4 of sum of the ASCII values in hash chunks. It will tell the server and through the hash chunks system will be able to brows the encrypted data chunk. After retrieving all four encrypted data chunks, they will be concatenated and decrypted using same "username + password". Finally user will have the stored data without any trouble. Refer table 3.1 to know how data is being stored in system.

Server X	
Hash chunk	Encrypted data chunk
3d58a719c6866b0214f96b0a67b37e51	CZWHkkOn9g8i1rdJkPsGn3QGxjJYrm9v58Zglw
a91e233ce0be126a08f35fdf4c043c61	wnpxgcANgyLXEBhVIPnIvNPlilYM74NJB73gL
26f40139bfbc338d44eb2a03de9f7bb8	bHiS5wodxRGcmlhUA3lnXT+ucE7vOPZSDFK
eff0ac260b3629811e389a5fbee8a894	YYDO438BI++t0c68vb5RljvLvyRaFCgzRMjaW
7014bed0e0d43f0f3d76123c5c5c39e5	moET9pJ1zcg7PXDT8gP5jhpZOKqZaJ5oVvBRn
27b40812b93032c1acf82ae826454ce2	bn0wmHKYn0UQr9Pf8Py+Dxq196nE5Zz0Lshl3K
d2b944787f6f984ffd9e6bd1e14cfba5	bAVJU6r20vKm7TTCqcUcc5YzSkk+xnkfNX+H
41c824728b6a78301f253cd397c63ef0	Z5CWUVE2CV0BiIPU8iuIh8Z/j1SGUOn3j94yHk

Table 3.1: Data storage

You can get the overall idea of data retrieving from Figure 3.2.

When user comes to retrieve the stored data, system will provide what user needs. What happened when user insert a wrong username or a wrong password. Normal system may alert user about the incorrectness. But this system is something different. It will provide bogus data for bogus username and password combinations. That is the beauty of this algorithm. On the other hand, this system is highly robust when securing users data. No one can stole correct data from the system.

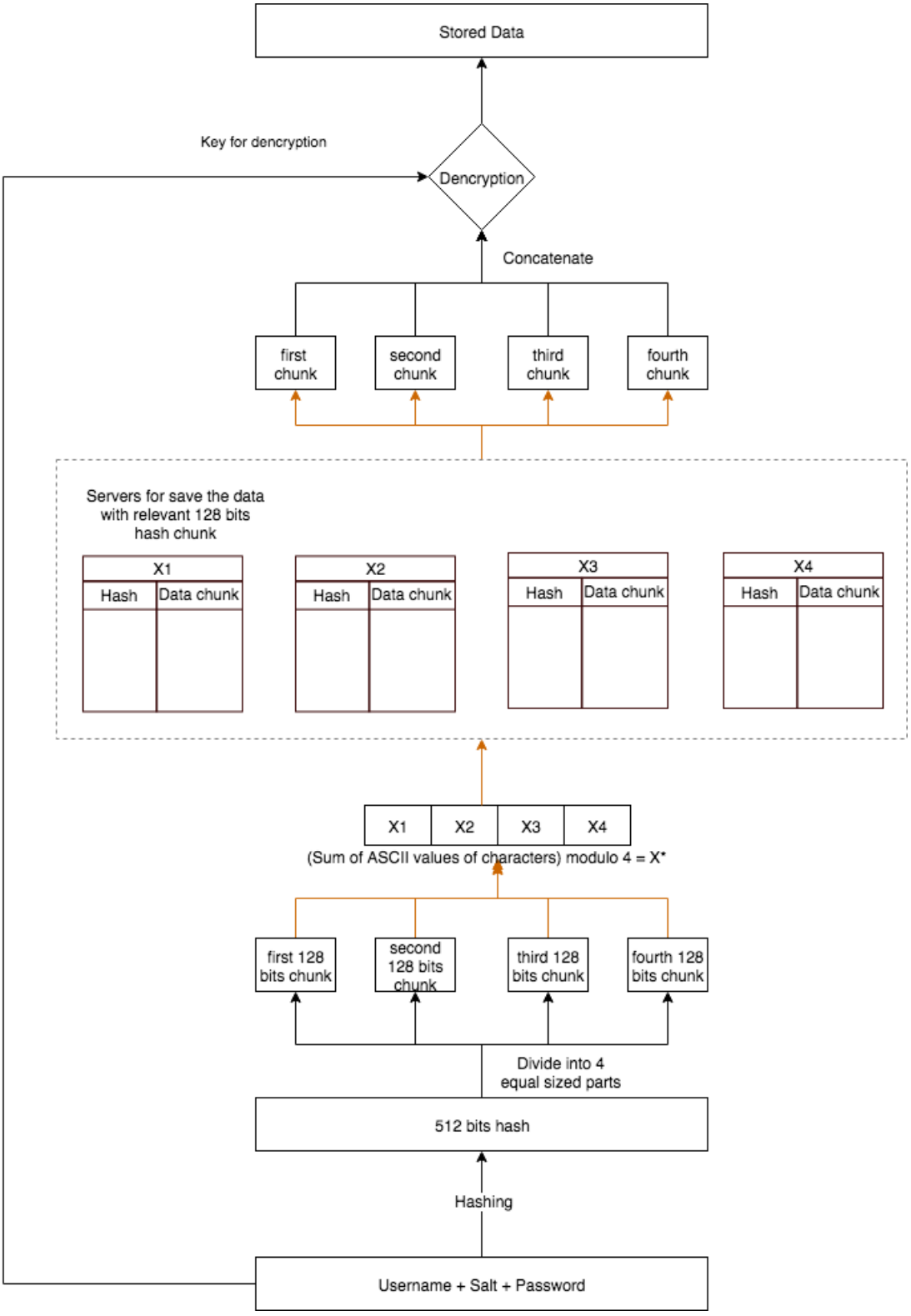

Figure 3.2: Data retrieving architecture

Chapter 4

Implementation

System requires several methodologies when it comes to implementation. Data storing and retrieving both are little bit similar. Mostly data retrieving mechanism slightly equal to reverse of data storing mechanism. Below I have mentioned basic pseudo codes for data storing and data retrieving functions without advance features.

Pseudo code for data store

Insert username

Insert password

Generate hash of username + salt + password

Divide hash to n number of parts

for i = 1 to n:

    xi = ASCII sum of each part modulo n

Insert Key

Encryption algorithm

Encrypt the Key using username + password

Divide encrypted Key in to n number of parts

for I = 1 to n:

yi = (divided key parts)i

Select the database xi

Store hash and key parts in relevant xi database

Pseudo code for data retrieve

Insert username

Insert password

Generate hash of username + salt + password

Divide hash to n number of parts

for i = 1 to n:

xi = ASCII sum of each part modulo n

Select relevant xi database

Retrieve the key part related to hash string part

Concatenate all retrieved key parts

Decrypt the Key using username + password

Selecting how many data stores to use, is a decision need to be taken with considering so many factors.

- Size of the hash string part. (size will be reduce if large number of data stores selected)
- Retrieving time of stored Key.
- Security of the Key may increase according to number of data stores, but it will be a time consuming task when data retrieving.

Basic Pseudo code for select database

Insert username

Insert password

Generate hash of username + salt + password

Divide hash to n number of parts

for i = 1 to n:

    xi = ASCII sum of each part modulo n

Chapter 5

Experiment and System Behaviour

Summary

This research study has a simple but powerful concept of data protection. But it requires simple calculations and functions. Below you may see all the functionalities it has.

- Hashing
- Encryption
- String concatenate and divide
- Data storing and retrieving

This methodology requires a hash function which generate 512 bits long string (). It will minimize probability of happening collisions. As above mentioned, hash string and encrypted Key will be divided in to 4 parts. One part will be 128 bits in size. When it is stored, hash string will act as the index. It will help to retrieve

the Key. That requirement can be covered with "blake2b512" , "sha3-512" and "sha512" hashing algorithms. Encryption can be covered with AES (Advanced Encryption Standard) encryption algorithm.

There is a serious problem with store hashes, Collisions. When a collision happened system will ask the user to choose a new username, password pair. But the probability of collisions is far away than real situation. According to wikipedia's Birthday attack article, 128 bits long string will be collided with another string with 0.1% of probability if generated string count equal to 8.3 * 10^{17}.

Imagine all the person in earth will store a Key in this system (At current moment world population is close to 8 * $10^9 = 8Billion$).

If there are that much of strings, probability of a collision is (according to birthday attack) less than (1 / 10^{18}).

Below Figure 5.1 will give a clear idea about probability of collisions with the string count.

Bits	Possible outputs (H)	Desired probability of random collision (2 s.f.) (p)									
		10^{-18}	10^{-15}	10^{-12}	10^{-9}	10^{-6}	0.1%	1%	25%	50%	75%
16	2^{16} (~6.5×10^4)	<2	<2	<2	<2	<2	11	36	190	300	430
32	2^{32} (~4.3×10^9)	<2	<2	<2	3	93	2900	9300	50,000	77,000	110,000
64	2^{64} (~1.8×10^{19})	6	190	6100	190,000	6,100,000	1.9×10^8	6.1×10^8	3.3×10^9	5.1×10^9	7.2×10^9
128	2^{128} (~3.4×10^{38})	2.6×10^{10}	8.2×10^{11}	2.6×10^{13}	8.2×10^{14}	2.6×10^{16}	8.3×10^{17}	2.6×10^{18}	1.4×10^{19}	2.2×10^{19}	3.1×10^{19}
256	2^{256} (~1.2×10^{77})	4.8×10^{29}	1.5×10^{31}	4.8×10^{32}	1.5×10^{34}	4.8×10^{35}	1.5×10^{37}	4.8×10^{37}	2.6×10^{38}	4.0×10^{38}	5.7×10^{38}
384	2^{384} (~3.9×10^{115})	8.9×10^{48}	2.8×10^{50}	8.9×10^{51}	2.8×10^{53}	8.9×10^{54}	2.8×10^{56}	8.9×10^{56}	4.8×10^{57}	7.4×10^{57}	1.0×10^{58}
512	2^{512} (~1.3×10^{154})	1.6×10^{68}	5.2×10^{69}	1.6×10^{71}	5.2×10^{72}	1.6×10^{74}	5.2×10^{75}	1.6×10^{76}	8.8×10^{76}	1.4×10^{77}	1.9×10^{77}

Figure 5.1: Birthday attack

Hashing, encryption and Data storing/retrieving all has a common approach when dividing the strings. You may get a clear idea about string dividing from below Figure 5.2.

```
username+salt+password
Hash string:  3d58a719c6866b0214f96b0a67b37e51a91e233ce0be126a08f35fdf4c043c6126f40139bfbc338d44
eb2a03de9f7bb8eff0ac260b3629811e389a5fbee8a894
First hash chunk: 3d58a719c6866b0214f96b0a67b37e51
Second hash chunk: a91e233ce0be126a08f35fdf4c043c61
Third hash chunk: 26f40139bfbc338d44eb2a03de9f7bb8
Fourth hash chunk: eff0ac260b3629811e389a5fbee8a894
```

Figure 5.2: String dividing

5.1 New features

- System will not store full hash of the password like normal systems. System uses hash of the "username + salt + password" and divide it into n number of chunks and stored in several data stores.
- Normal systems will alert for bogus username and passwords. But this system will provide bogus data for bogus username and password combination. Therefore Brute-force attack is not possible.
- Provided data will be stored in different places by dividing to n number of chunks.

Chapter 6

Conclusions

System

Security of the data is a high concern and people needs trustful method to deal with it. Carrying and protecting of Key (confidential data) is a high risk and this moment is the easiest way to steal data from Carrier. Best solution for this problem is save the data in highly protected cloud. If you can protect your data in cloud, there is no need of carrying and data stolen will never be happened because you never carry data. This study came with the solution and it has new features (mentioned above chapter 5.1) for enhance the protection of cloud data.

Even though data stores are hacked or illegally obtained by another party, there is possibility for them to not getting the right data in to their hands. Because data which are stored in databases are not connected in any manner other than "username + password" combined hash string. Without the hash string intruders are blind with stolen data.

6.1 Future works

- Increasing number of data stores significantly enhance the security of data. As well as increasing the size of hash string also enhance the security. These

both aspects need to be covered, without making data retrieving part a time consuming task.

- At the moment protection of stored data is depend on username and password. This is "Knowledge factors", Factors the user must know in order to log in are considered a knowledge factor. This can be anything from a username, password, or pin number. The challenge with these factors is that they can be weak in terms of security because they can be shared or guessed. But this would be enhanced with "Inheritance factor" which is "Using a person's biological characteristics is known as an inheritance factor. Any biometric authentication process, such as fingerprint scanning and face recognition, would fall into this category" and "Possession factors", Anything that the user must have in order to log in is known as a possession factor. One-time password tokens, key fobs, ID cards, and physical tokens are all considered possession factors.

References

[1] G. Hamilton, "Symmetric key distribution," p. https://www.computing.dcu.ie/ hamilton/teaching/CA642/notes/KeyDistribution.pdf.

[2] I. P. F. System p. https://ipfs.io/.

[3] I. P. F. System p. https://en.wikipedia.org/wiki/InterPlanetaryFileSystem.

[4] pp. Share files in IPFS – https://data–flair.training/blogs/ipfs–blockchain/.

[5] pp. How does bitTorrent work: https://www.howtogeek.com/141257/htg–explains–how–does–bittorrent–work/.

[6] p. How does bitTorrent work (cont): https://computer.howstuffworks.com/bittorrent.htm.

[7] pp. Step by step blockchain technology: https://blockgeeks.com/guides/what–is–blockchain–technology/.

[8] "Wiki," p. Blockchain : https://en.wikipedia.org/wiki/Blockchain.

[9] pp. Peer–to–Peer Electronic Cash System – Bitcoin.org: https://bitcoin.org/bitcoin.pdf.

[10] p. Hashing algorithms: https://en.wikipedia.org/wiki/Secure$_{H}ash_{A}lgorithms$.

[11] pp. Sorting algorithms: https://www.sciencedirect.com/topics/engineering/sorting–algorithm.

[12] pp. Sorting: https://www.cs.cmu.edu/ adamchik/15–121/lectures/Sorting%20Algorithms/sorting.html.

[13] p. Birthday attack wiki: https://en.wikipedia.org/wiki/Birthday$_a ttack$.

[14] pp. Birthday attack sciencedirect: https://www.sciencedirect.com/topics/computer–science/birthday–attack.

[15] pp. Birthday attack geeksforgeeks: https://www.geeksforgeeks.org/computer–network–birthday–attack/.