MATEC Web of Conferences **40**, 09003 (2016)
DOI: 10.1051/matecconf/20164009003

Enhanced Obfuscation Technique for Data Confidentiality in Public Cloud Storage

S. Arul Oli[1] , Dr. L. Arockiam[2]

[1] *Doctoral Research Scholar, Department of Computer Science, St. Joseph's College (Autonomous), Tiruchirappalli, India*
[2] *Associate Professor, Department of Computer Science, St. Joseph's College (Autonomous), Tiruchirappalli, India*

Abstract. With an advent of cloud computing, data storage has become a boon in information technology. At the same time, data storage in remote places have become important issues. Lot of techniques are available to ensure protection of data confidentiality. These techniques do not completely serve the purpose in protecting data. The Obfuscation techniques come to rescue for protecting data from malicious attacks. This paper proposes an obfuscation technique to encrypt the desired data type on the cloud providing more protection from unknown hackers. The experimental results show that the time taken for obfuscation is low and the confidentiality percentage is high when compared with existing techniques.

1 Introduction

We are in the middle of an insurgency in cloud computing. The rapid deployment in cloud gives tremendous advantages to cloud users. Cloud Computing (CC) is defined as "a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or services provider interaction" [1]. In nutshell, CC could be comprised as '3 4 5', i.e., Three service models as SPI, namely Software as a service, Platform as a service, Infrastructure as a Service, Four deployment models as P2CH namely Private, Public, Community and Hybrid clouds and Five essential characteristics as on-demand self-service, broad network access, resource pooling, rapid elasticity, measured service. These essential characteristics promote lot of benefits in cloud computing utilization, working in various P2CH cloud environments, keeping SPI as base models.

Many techniques are available to protect sensitive data of users from attackers and service providers. Our approach is to protect data confidentiality from service providers, and to ensure that service providers cannot tamper the data stored in cloud environments. CC fascinates the users with many advantages. However several important problems in encryption process are resolved through data obfuscation method.

The term obfuscation means attempting "to transform a program into an equivalent one that is harder to reverse engineer" [2]. The earliest work was done by Cohen [3] who called this as "security through obscurity" and advocated program evolution as the technique to increase complexity. This was the first of many techniques for data obfuscation where obscurity is maximized and execution time is minimized.

The goal of data obfuscation is to insert complex code or data structures into programs, thus making the programs supposedly harder to analyse and making sensitive data difficult to tamper. Many research findings on obfuscation proved that data obfuscation made the intruders harder to understand and even impossible in some cases.

Various obfuscation techniques are available to protect the confidentiality of data. For an example the word 'GOD' is encoded as 'R09E' using base64 encoding. It is converted into ASCII characters, ASCII characters into binary bits and grouped into 6 digits. The 6 digits are converted into decimal values and then equivalent character of it. The base32 encoding, ASCII encoding and hexadecimal encoding are few examples for obfuscation where the data are converted into non-readable format. These examples are used in this paper for comparison with the proposed obfuscation technique.

This paper is organized as follows: Section 2 deals with the related works which support the obfuscation method. This section leads to Motivation and Derivation, and Objective of this paper in section 3 and 4 respectively. Section 5 gives the Methodology followed by the Proposed Obfuscation technique in section 6. Section 7 gives the experimental results. Section 8 concludes the paper.

2 Related Works

The authors [4] proposed a model to encrypt and to obfuscate data on client side before uploading into cloud database and proposed a mechanism to query over

Article available at http://www.matec-conferences.org or http://dx.doi.org/10.1051/matecconf/20164009003

encrypted and obfuscated data on server side and decryption and de-obfuscation is performed on client side. Anitha et al. proposed a method to provide protection of data with cipher key stored through metadata at the data server [5]. The cipher key generation is time consuming which is proportional to the number of attributes in the metadata as well the algorithms used for cipher key generation.

The privacy manager called obfuscation discussed in [6] reduces risks of data misuse of cloud users where a key chosen by user is not communicated to service provider, making it to de-obfuscate user's data. The privacy manager is also discussed [7] in conforming to privacy law in order to control policy-based obfuscation and de-obfuscation of personal, sensitive, or confidential data within cloud system. The plug-in log of obfuscation keys with the use of Double Authentication and Hybrid Obfuscation Technique, the maximum security and privacy for data is ensured in [8]. The 'divide and rule' policy for safe and secured cloud environment is limited due to the implementation of two clouds being active and responsive all the time without direct communication.

The performance results of data confidentiality through scalable approach [9] reveal that the overhead for data obfuscation and de-obfuscation appear to increase linear with the size of input data. The architecture [10] provided a user-centric trust model to ensure and to control the security of user's sensitive data instead of leaving them entirely on server-centric for implementation. The key is not revealed to the service provider in this obfuscation method for protecting security of data. This approach of obfuscation for security is further proposed in [11] for protection by increase of complexity to withstand malicious attacks.

3 Motivation and Derivation

From the above literatures and available findings, it is obvious that the protection of data from malicious threats is a prime concern in enhancing the security of data. Many research findings also proved that different techniques have not fully protected the confidentiality of data. Hence confidentiality factor needs to be considered more importance in spite of many advantages in cloud storage.

In spite of many confidentiality techniques, the obfuscation technique plays a vital role to protect the data from malicious attacks. This obfuscation technique is in a way "a black box" so to say in protecting the data from reverse engineering. Motivated by this factor, a novel obfuscation technique is proposed in this paper for protecting the data before uploading into cloud storage.

Data security is an aspect of concern in cloud computing. The data could be placed anywhere in physical form in cloud storage only left with the service provider having full liberty to access to data with its own control and privileges. This aspect leads more concerns with regard to data confidentiality. Since the users have no control over the data in cloud storage and providers have more privileges for data access, it is a temptation for the providers to mishandle the data or data has potential danger of being tampered by unauthorized users.

The maintenance of secret keys is rather difficult and worse when the keys are mishandled with other providers. So maintaining the secret keys with the users, both the data and the keys are safe with provider and with user respectively. So, it is obfuscated and then uploaded into cloud for protecting the data confidentiality. Hence security is enhanced. By enhancing the confidentiality techniques through the proposed obfuscation method, the security is enhanced. The size of the data is minimized. The time taken for storage is reduced.

4 Objective

The objective of this paper is to experiment and bring out the good results of data security by use of obfuscation method. By enhancing the confidentiality of data by obfuscation, the user feels more reliable with service provider. The user further makes it more confident that as long as the secret key is with him, the data in cloud storage is safe and unhacked. By proving less time and more security with this method, the user makes it sure that this obfuscation method is more suitable for data confidentiality.

5 Methodology

Data stored in the cloud are protected for maintaining security. Security of the data is maintained by ensuring confidentiality technique. The confidentiality of data stored in the cloud is ensured by the obfuscation technique. The proposed technique obfuscates the data before uploaded into the cloud. Figure 1. Represents the methodological diagram of obfuscation.

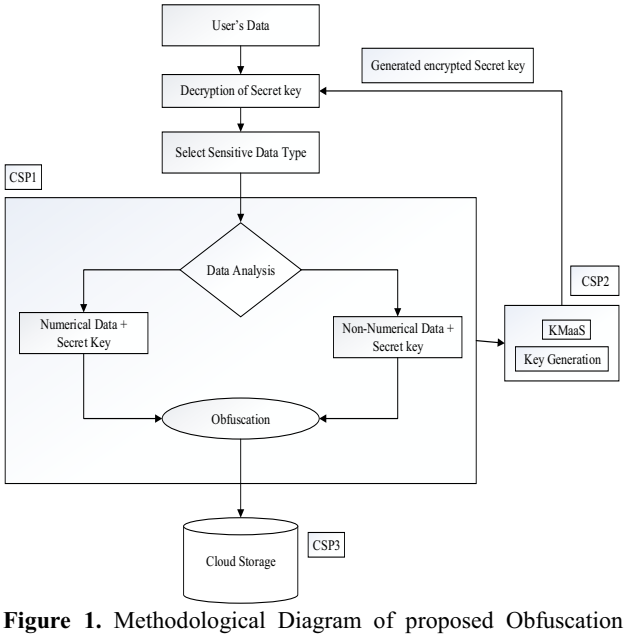

Figure 1. Methodological Diagram of proposed Obfuscation Technique

The proposed technique uses a key to obfuscate the data. The key is generated in the key management as a service. The generated key is securely sent to the user

using symmetric cryptosystem. This secret key is used for obfuscation and de-obfuscation of data in user's side. The user chooses the type of sensitive data for obfuscation. The proposed technique obfuscates the selected sensitive data with the secret key received from the cloud key management as a service. The same key is used for de-obfuscation. User should keep the key secure. At the completion of obfuscation, the data are uploaded on to cloud storage. Figure 2. Represents the over-all idea of cloud architecture where various service models function as different service model.

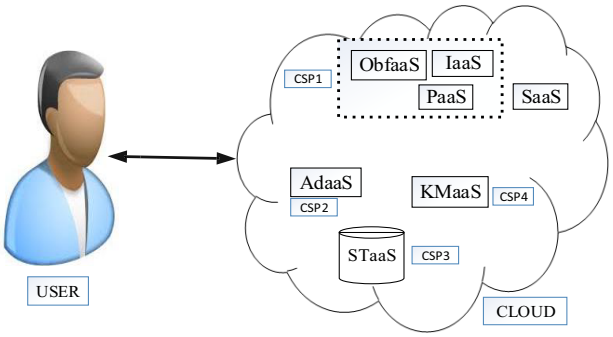

Figure 2. Cloud Environment with different services

In the provider's side, each provider functions as different providers. In the above diagram, Cloud Service Provider (CSP) functions as Obfuscation as a Service (ObfaaS) for data obfuscation. The data is stored in the CSP3 where it functions as Storage as a Service (STaaS). While secret key is shared between the users, the CSP2 functions as Audit as a Service (AdaaS) for authentication of users. While maintaining the key and data in cloud service provider, the CSP4 functions as Key Management as a Service (KMaaS). Each provider invokes the command and request for various operations and demands of users to utilise cloud resources.

6 Proposed Obfuscation Technique

The proposed technique is an obfuscation technique. Generally, obfuscation is a technique like encryption but it uses different mathematical methods or some critical programming logics to hide original data. The proposed technique obfuscates the data before they are uploaded into the cloud. The proposed technique uses different methods like ascii(), binary(), rotate(), two complex() and decimal() to obfuscate the original data. It also uses a key which is used to rotate the bits. The steps in the proposed obfuscation technique is as follows:

Step 1. The given original text is converted into ASCII codes
Step 2. ASCII codes are converted into binary values 0s' and 1s'
Step 3. Generate a key k
Step 4. Rotate the bits k numbers of time from right to left
Step 5. Calculate 2's complement
Step 6. Divide the bits into 8 bits
Step 7. Convert the binary into decimal
Step 8. Convert the decimal into ASCII character codes

The pseudo code for the proposed obfuscation technique is as follows:

Pseudo Code for Proposed Obfuscation Technique

```
Obfuscation_text (PT)
        Declaration
            PT ← Plaintext
            S ← Size of PT
            AC ← ASCII codes of PT
            BIN ← Binary code for AC
            BUFF ← Buffer Variable
            K ← Key for Bits Rotation
            RD ← Rotated bits
            TC ← Two's Complement
            N ← Number of 8 bits binary
            BIT ← 8 bits Binary
            DEC ←Decimal value
            CT ← Ciphertext
1.  Start
2.  S ←sizeof(PT) // method to find the size of the PT
3.  for i ← 1 to S
    AC[i] ← ascii(PT[i]) //method to convert into ASCII
    BIN[i] ←binary(AC[i]) //method to convert into 8 bits
    binary
    BUFF ←append(BIN[i]) // Buffer variable for combine all
    the binaries
    next i
4.  Generate a key from cloud K
5.  for i ← 1 to K
    RD ← rotate(BUFF) // method to rotate the bits one time
    next i
6.  TC ← two_comple(BUFF) //method to calculate two's
    complement
7.  N ← S/8
8.  for i ← 1 to N
    BIT[i] ←div_bits(TC) //method to divide the binaries into 8
    bits
    DEC[i] ←decimal(BIT[i]) // method to convert the 8 bits
    into decimal
    CT[i] ← ascii(DEC[i])
    next i
9.  CT ←Ciphertext
End
```

7 Experimental Results

The proposed algorithm is implemented in JAVA. The plaintext is given as the input. The name of the file is "Plaintext_File.txt".

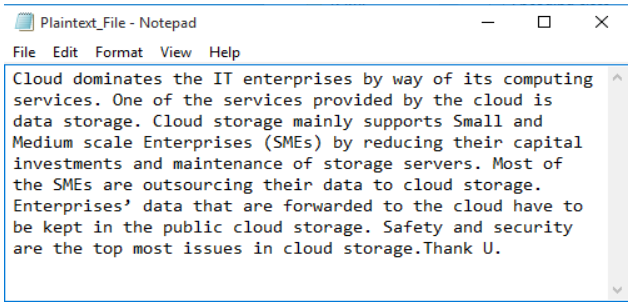

Figure 3. Plaintext File for input

Figure 3. shows the plaintext file for input. This file is obfuscated using the proposed technique and the obfuscated text are stored in another file called

"Obfuscatedtext_File.txt". Figure 4. shows the obfuscated text as the result of proposed technique.

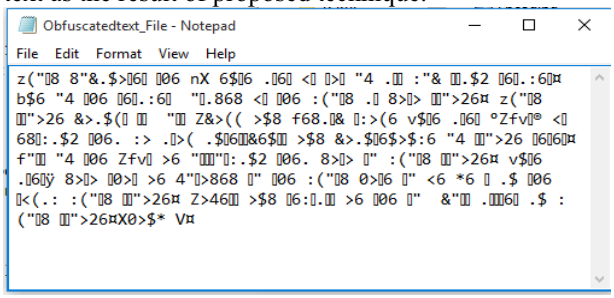

Figure 4. Obfuscated File as Output

The obfuscated file is uploaded to the cloud. To access the obfuscated data, it should be de-obfuscated. The de-obfuscated data stored in a file is called "De-obfuscated.txt". Fig 5. shows the de-obfuscated data file.

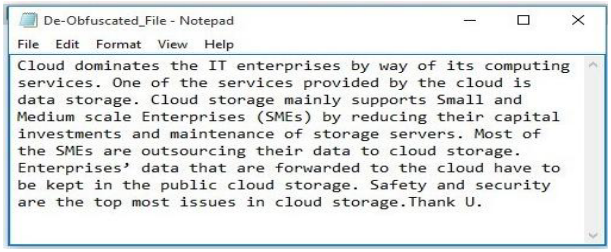

Figure 5. De-obfuscated text file

The users' data are obfuscated before it is uploaded and de-obfuscated once retrieved from the cloud. Thus, the obfuscation is done in the user's machine connected to the cloud. Time taken for obfuscation and de-obfuscation is calculated in the user's machine. Once the data are submitted, then they are obfuscated and uploaded to the cloud server. Security levels of the existing and proposed obfuscation techniques are computed in cloud server. Security level is analysed by using a security analysis tool called Hackman. Performance and security level of proposed technique is compared with existing techniques.

Table 1 represents the performance comparison of obfuscation with existing techniques such as Hexadecimal Encoding, Base 32, and Base 64. The time taken by the existing and proposed obfuscation techniques are calculated for different sized plaintexts. The result shows that compared to the existing obfuscation techniques, the proposed technique has taken low time for obfuscating different size of plaintext.

Table 1. Performance comparison of obfuscation techniques

Size bytes	Hexadecimal Encoding	Base32	Base64	Obfuscation
500	4.51	2.02	3.20	1.08
1000	9.8	4.1	5.91	2.1677
2000	19.20	8.12	9.98	4.33
4000	33.78	16.21	19.01	8.67
6000	45.32	22.10	28.32	12.03
8000	57.8	28.24	35.01	16.35
10000	71.3	32.40	42.12	19.21

Figure 6. represents the graphical performance comparison of obfuscation with existing techniques such as Hexadecimal Encoding, Base 32, and Base 64. The time taken by the existing and proposed obfuscation techniques are calculated and it is shown.

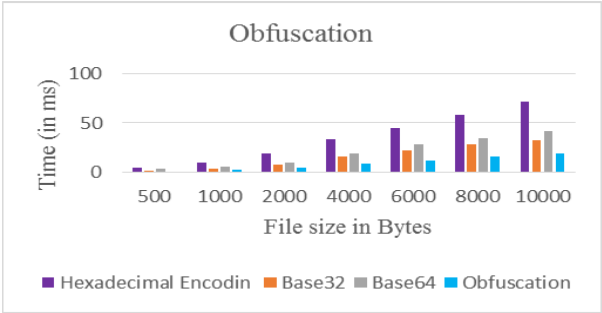

Figure 6. Performance comparison of obfuscation techniques by time

Table 2 represents the graphical performance comparison of de-obfuscation with existing techniques. The time taken by the existing and proposed de-obfuscation technique is calculated for different size of plaintext. The result shows that compared to the existing de-obfuscation techniques, the proposed technique has taken low time for de-obfuscation of different size of plaintext.

Table 2. Performance comparison of de-obfuscation techniques

Size bytes	Hexadecimal Encoding	Base32	Base64	De-Obfuscation
500	4.01	1.98	3.01	0.99
1000	8.4	3.99	5.98	2.01
2000	16.02	7.48	9.50	4.0
4000	31.53	15.26	18.34	8.01
6000	44.82	20.99	26.92	12.98
8000	56.95	27.32	34.82	16.23
10000	69.21	31.01	41.04	19.42

Figure 7. represents the graphical performance comparison of de-obfuscation with existing techniques such as Hexadecimal Encoding, Base 32, and Base 64. The time taken by the existing and proposed de-obfuscation techniques are calculated and it is shown.

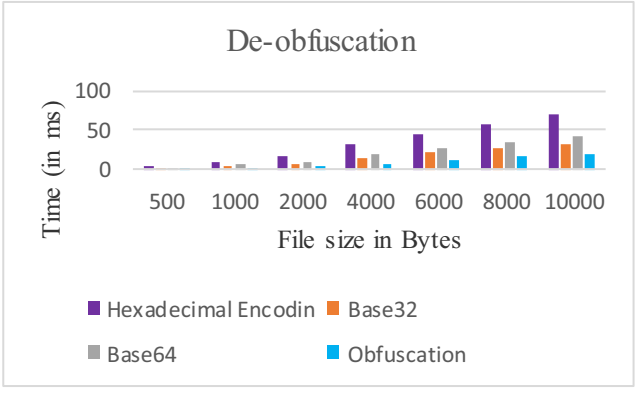

Figure 7. Performance comparison of de-obfuscation techniques

Table 3 represents the comparison of security level with existing techniques. The result shows that compared to the existing obfuscation techniques, the proposed technique has maximum percentage of security.

Table 3. Security level of obfuscation technique compared with existing techniques

S. No.	Obfuscation Technique	Security Level (%)
1.	Hexadecimal Encoding	62
2.	Base64	72
3.	Base32	67
4.	Obfuscation	83

Figure 8. represents the graphical performance comparison of security level with existing techniques such as Hexadecimal Encoding, Base 64, Base 32 and proposed obfuscation technique. The security level of the proposed obfuscation technique is higher compared to the other three techniques.

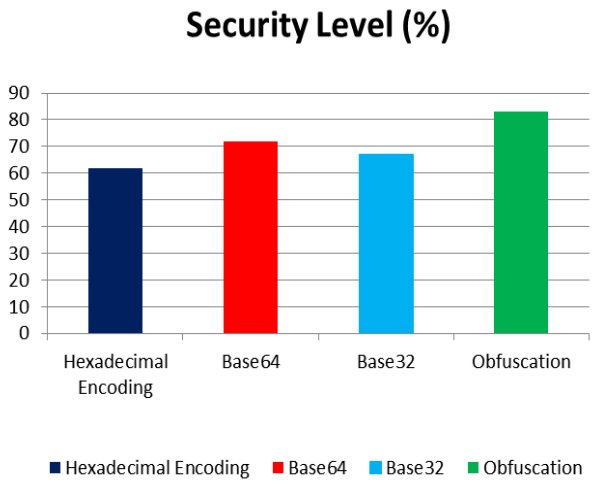

Figure 8. Performance comparison of Security Level

The above performance comparison of security level shows that the proposed obfuscation technique gives maximum security and better performance than existing techniques while storing data into cloud. Hence, the confidentiality of data stored into cloud is achieved.

From the above experimental results it is evident that security level is high in the proposed obfuscation technique. Less time, minimum data size and more efficiency are obtained in this technique compared to other techniques while using security analysis tool called Hackman. The debugging process is more difficult and complicated without the proper, reliable and well-defined de-obfuscation tool.

8 Conclusion

In this paper, a new obfuscation technique is proposed and implemented to secure data confidentiality. This novel method produced high percentage in security level and has taken minimum time for obfuscation and de-obfuscation process when compared with the existing techniques while encrypting and uploading data into public cloud storage. In this way the data confidentiality is protected in cloud storage. Thus the security is enhanced through this proposed obfuscation method.

References

1. P. Mell, T. Grance, Recommendations of the National Institute of Standards and Technology, (2011)
2. C. Colberg, C. Thomborson, IEEE Trans. on Soft. Engg. **28**, 737 (2002)
3. F. Cohen, 1992, https://all.net/books/tech/evolve.pdf.
4. Atiq ur Rehman, M. Hussain, Intl. Jour. of Adv. Sc.Tech. **35**, 1-10 (2011)
5. R. Anitha, P. Pradeepan, P. Yogesh, and S. Mukherjee, *2nd International Conference on Machine Learning and Computer Science (ICMLCS' 2013)*, Kuala Lumpur (Malaysia), 26-30 (2013)
6. S. Pearson, Y. Shen, M. Mowbray, CloudCom '09 Proceedings of the *1st International Conference on Cloud Computing*, Springer-Verlag Berlin, Heidelberg, 90 – 106 (2009)
7. M. Mowbray, S. Pearson, Y. Shen, *J. of Super Comp.*267–291 (2012)
8. P. V. G. D. Prasadreddy, T. S. Rao, S. P. Venkat, *IEEE World Congress on Services*, (2011)
9. M. Hataba, A. El-Mahdy, *Seventh International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, IEEE Computer Society, (2012)
10. K. Govinda, E. Sathiyamoorthy, *Elsevier, Procedia Engineering, ICMOC*, 125-129 (2012)
11. S. Duggins, F. Tsui, O. Karam, Z. Kubanyi, *Intl. Conf. Software Engg. Research and Practice*, (2013)