Research Journal of Mathematics and Computer Science (ISSN:2576-3989)

An Implementation of a One-Time Pad Encryption Algorithm for Data Security in Cloud Computing Environment

Omotunde Ayokunle A[*a], Faith Adekogbe[b], Onuiri Ernest[c], Precious Uchendu[d]

[a]Computer Science Department, Babcock University, Ilishan-Remo, Ogun State, Nigeria.
[b]Computer Science Department, Babcock University, Ilishan-Remo, Ogun State, Nigeria
[c]Computer Science Department, Babcock University, Ilishan-Remo, Ogun State, Nigeria
[d]Computer Science Department, Babcock University, Ilishan-Remo, Ogun State, Nigeria

ABSTRACT

The cloud is a computing model used by many consumers which include individuals and organizations for data storage which hitherto demands that adequate security measures be put in place to protect the confidentiality and integrity of that information. In cases where these security measures are inefficient or in some cases non-existent, client data is prone to a number of unauthorized violations which include breach of privacy, loss of data, compromised data integrity, and data manipulation among others. This therefore necessitates the demand for efficient security measures. Encryption is a security technique adopted by many for data protection which entails concealing the information content of a message in a way that only the intended recipient can make use of it. This research paper discusses the concept of encryption, a review of different encryption schemes that have been discussed over the years, and proposes a one-time pad encryption (OTP) algorithm (FAPE's OTP). FAPE's OTP implements the one time pad using a key expansion process that transforms a 512 bit key to the length of the plaintext. This research was carried out through a comprehensive study of encryption and cloud processes to understand both concepts independently and determine how they can be interleaved while sustaining optimum delivery. Furthermore, our findings indicate that FAPE's OTP has a faster speed of operation in comparison to the Advanced Encryption Standard.

Keywords: Data Storage, Security, Encryption, One-Time Pad, Cloud Computing, Information

***Correspondence to Author:**
Omotunde Ayokunle A
Computer Science Department, Babcock University, Ilishan-Remo, Ogun State, Nigeria.

How to cite this article:
Omotunde Ayokunle A et al., An Implementation of a One-Time Pad Encryption Algorithm for Data Security in Cloud Computing Environment. Research Journal of Mathematics and Computer Science, 2017; 1:6 .

eSciencePublisher
eSciPub LLC, Houston, TX USA.
Website: http://escipub.com/

INTRODUCTION

The heightened demand for centralized information processing heralded the birth of cloud computing in our world today. Cloud computing enables client access to computing resources on a pay-per-use basis which eliminates the financial intensity of traditional computing technologies. Clients hereby relinquish part or in some cases full responsibility of securing their data to the cloud provider which necessitates that some basic measures be put in place. Infrastructure-as-a-Service (IaaS) offers numerous benefits to cloud users of which data storage is of high prominence. Furthermore, different problems faced in the infrastructure layer with respect to data storage include data privacy, data breach or loss, data integrity, data location, data availability and incomplete data deletion. Storing important information on the cloud in most cases can render it accessible to unauthorized parties in the cases of a cloud breach, especially when this information is stored in its plain form in the cloud. This then necessitates that the cloud service provider finds some means to protect the content of this information [1].

Numerous solutions have been proposed to effectively secure client information of which cryptography has stood out. Cryptography is a technique used to protect information from unwanted persons by converting it to an unrecognizable format both during transit and at rest, the field of cryptography is closely related to cryptology and cryptanalysis [2]. Encryption is a method of protecting data that has been proposed by many security experts with a view to combining multiple standard algorithms to create a very secure system. However, a major drawback of such hybrid systems is the large computational overhead it demands in terms of speed. In the past, encryption was predominantly used in military organizations as it played a major role in information transmission during World war I and World war II, but of recent the risk involved in a potential information leakage has greatly increased because of the heavy reliance on information quality in many business dealings, and so a technique that effectively mitigates these risks is required. In any modern cryptosystem it is expected that the following security criteria are met, namely: Secrecy, Reliability, Non-Repudiation and Certification, a cryptosystem is secure only when it can meet all four criterions [2]. This research work proposes a single layer symmetric key algorithm (FAPE’s OTP) that protects text files (.txt, .docx) stored in the infrastructure layer of the cloud. In this paper current implementations of cryptography in the cloud are reviewed, followed by a brief exposition of FAPE’s OTP as well a comparative analysis with the Advanced Encryption Standard (AES).

AES is the most widely used in the World Encryption Data Standard approved by NIST (National Institute of Standards and Technology). The Rijndeal algorithm was selected as a standard encryption algorithm in 1997 by NIST, this was done to replace the Data Encryption Standard (DES) which was the previously used standard, the Rijndael algorithm after the standardization process by NIST was officially adopted as the Advanced Encryption Standard [3]. Furthermore AES is a block cipher that implements permutation and substitution, AES is divided into three based on their key sizes namely

- AES-128
- AES-192
- AES-256

The electrical One-Time Pad was created by Gilbert Vernam in 1917 for telegraph encryption. It was referred to as the Vernam cipher and each character in the massage was combined with each character on a paper tape using the XOR operation for encryption [4], [5].

BACKGROUND

The implementation of cryptography for cloud storage is not a new concept as it has been discussed in detail by many researchers, this section contains a few of these research works arranged in chronological order.

Table I: Review of Related Works

	Title	Work Done	Advantages	Disadvantages
1	Elliptical curve cryptography (ECC)	Proposed Elliptical Curve Cryptography (ECC) - a public key encryption technique based on elliptic curve theory used to create faster, smaller, and more efficient cryptographic keys [6].	The parameter which is the order of the point on the curve determines the strength and level of security of the system [6].	No implementation details
2	A fully Homomorphic encryption implementation on cloud computing	The application of a fully Homomorphic system is a major milestone in cloud security; furthermore, outsourcing calculations on client data to the Cloud server is possible, while sustaining data privacy [7].	Data Privacy is kept at an optimum level since the content of the message is only available to the sender [7].	Implementation details are not discussed so there is no consideration on working capacity.
3	Security System for Healthcare Data in Cloud Computing	Proposed AES-256/SHA (Secure Hash Algorithm) for the purpose of encryption-decryption. Re-encryption data is used to tag and mark data that is to be saved in cloud storage. Any untagged person will be unable to access the data since no authorization is given [8].	This encryption system takes into consideration authorization so as to avoid repudiation of information [8].	This is basically a theoretical framework with no details of the implementation of the system
4	Triple Security of Data in Cloud Computing	Proposed the adoption of three different layers of security which includes DSA (Digital Signature Algorithm), DES (Data Encryption Standard) and Steganography [9].	The adoption of steganography hides the existence of the message from intruders which improves security [9].	A major disadvantage is the computational overhead involved in terms of speed
5	Security in Cloud Computing using Cryptographic Algorithms	Proposed a combination of DES (Data Encryption Standard-Symmetric key algorithm) and RSA (Rivest Shamir Adleman-Asymmetric key algorithm) [10].	The combination of a symmetric key and an asymmetric key algorithm adds another layer of security [10].	No implementation details of this algorithm and hence does not discuss its feasibility
6	A New Technique for One Time Pad Security Scheme with Complement Method	Proposed the adoption of a One Time Pad (OTP) cryptosystem through which the OTP can be practically used for secure communication between client and server using the complement method [2].	Its simplicity aids fast communication between both parties involved [2].	The use of an 8 bit key for encryption is a major weakness of this system that could lead to information compromise

FAPE's OTP ALGORITHM

The encryption algorithm proposed in this research work combines various strengths of existing cryptosystems to optimize security in cloud with considerations to computing overhead. Major features of this algorithm include;

1. *512 bit key constant key size, with variable length appended offsets*: The key used in the encryption/decryption process is generated as a random sequence of 512 bits, this large key size re-enforces this algorithm against brute force attack which entails trying a combination of all possible key sequences to break or decode a cipher, brute force mandates that the attacker knows the distribution of the key bit size. The use of a 512 bit key reduces the likelihood of occurrence of brute force in the nearest future.

   Using a 512 bit key, it takes 2^512 trials to analyze all possible combinations of the key.

   2^512 = 1.3407807929942597099574024998206e+154

2. *One time pad:* this is an encryption technique developed in the early 90's, it is known as the only perfect cipher but a factor that has impeded its implementation in modern day cryptography is its key size. The one time pad cipher mandates that the key size be as long as the size of the plain text, this criterion has been a major limiting factor to its implementation as the computational overhead in terms of size is great. This research work implements the one time pad using a key expansion and reduction process, during the encryption process 512 bit key is expanded by performing a NOT operation and shifting they key by a random integer, this is done continually, until the key size is as long as the plain text, the random numbers used for shifting are stored alongside the key, this process is repeated during the decryption process to generate the same key.

3. *Speed using multithreading:* a major factor to take into consideration in cryptosystems is the speed with which it works, this algorithm implements multithreading techniques by dividing the main execution thread into four threads that work simultaneously to hitherto speed up the process.

A. The Encryption Algorithm

```
Step 1: Load file
Step 2: Read the content of the file and store in an arraylist Plain
Step 3: Calculate the number of blocks in the file and the offset (Plain / 64 = blocks; Plain % 64 = offset)
Step 4: Generate 512 bit random key
Step 5: Generate a random integer for each file block and store in an array
Step 6: Shift the key four times by four of the above generated random numbers and store the shifted keys as s1, s2, s3, s4
Step 7: Generate a one-time pad (OTP) with the key and the random integers
    a: Get the total size of the plain text
    b: Shift the key with a random integer
    c: Inverse the shifted key NOT(shiftkey)
    d: Repeat steps b : c for each block, append it to the one time pad array
    e: For the offset append one (1) to the pad and append zeros till the pad is complete
    f: Return one time pad = OTP
Step 8: Break the plain text into blocks of 512 bit each perform a Cipher Block Chaining (CBC) with each of the previously shifted keys
    a: Exclusive Or the block and key, block XOR key = text1
    b: Exclusive Or text1 and s1, text1 XOR s1 = text2
    c: Exclusive Or text2 and s2, text2 XOR s2 = text3
    d: Exclusive Or text3 and s3, text3 XOR s3 = text4
    e: Exclusive Or text4 and s4, text4 XOR s4 = text5
    f: Append text5 to the array text
    g: Repeat steps a : f for each block
Step 9: Exclusive Or each bit in OTP and the bit in text, OTP XOR text = cipher
Step 10: Convert the binary values in cipher to character and store in file
Step 12: Store the random key into the key file and append it with all random numbers used for shifting
Step 13: End
```

B. *The Decryption Algorithm*

```
Step 1: Load file
Step 2: Read the content of the file and store in an arraylist
Step 3: Calculate the number of blocks in the file and the offset (Plain / 64 = blocks; Plain % 64 = offset)
Step 4: Read the content of the key file, store the key into an array
Step 5: Store the shift integers into another array
Step 6: Shift the key four times by 4 of the shift integers and store as s1, s2, s3, s4
Step 7: Generate a onetime pad with the key and the random integers
    a: Calculate the total size of the plain text
    b: Shift the key with the shift integers gotten from the key file
    c: Inverse the shifted key NOT (shiftkey)
    d: Repeat steps b : c for each block, append it to the one time pad array
    e: For the offset append one (1) to the pad and append zeros till the pad is complete
    f: Return one time pad = OTP
Step 8: Convert the cipher text to binary values = text
Step 9: Exclusive Or each bit in OTP and the bit in text, OTP XOR text = text1
Step 10: Break the cipher text into blocks of 512 bit each perform a Cipher Block Chaining (CBC) with each of the previously shifted keys
    a: Exclusive Or the tex1 and s4, text1 XOR s4 = text2
    b: Exclusive Or text2 and s3, text2 XOR s3 = text3
    c: Exclusive Or text3 and s2, text3 XOR s2 = text4
    d: Exclusive Or text4 and s1, text4 XOR s1 = text5
    e: Exclusive Or text5 and key, text5 XOR key = text6
    f: Append text6 to the array plain
    g: Repeat steps a : f for each block
```

Step 11: Convert the binary values to character and store into the plain text file

Step 13: End

IMPLEMEMTATION AND TESTING

The algorithm is implemented using Java language on Netbeans IDE. A comparison is also done in terms of speed with Advanced Encryption Standard (AES) which is implemented under the same conditions. For this test, a laptop with a processor core i3 at 2.10 GHz is used, where performance data is collected. For each run, text files with sizes in a range of 63 Kilo-bytes to 995 Kilo-Bytes are encrypted using both algorithms. The performance metrics of interest are: encryption time, decryption time, throughput of encryption and decryption.

The encryption time is the time that an encryption algorithm takes to produce a cipher text from a plaintext given the key. Encryption time is used to calculate the throughput of an encryption scheme which indicates the speed of encryption [11].

Encryption Throughput = Te (kilobytes/second)

Total size of plaintext = P (kilobytes)

Encryption Time = E (seconds)

Te = P/E

Decryption Throughput = Td (kilobytes/second)

Total size of ciphertext = P (kilobytes)

Decryption Time = D (seconds)

Td = P/D

A. *Encryption*

Advanced Encryption Standard (AES) and the FAPE's OTP are used to encrypt text files of varying size and the result is given below;

Table I: Encryption Test

i	Input size (kb) [Pi]	AES [E1i]	FAPE's OTP [E2i]
1	63	12	10
2	125	18	15
3	249	36	27
4	498	68	55
5	995	135	113
Total	1930	269	220
	Throughput (Kilobyte s/sec)	7.1747	8.7727

NB: AES encrypts 7.1747 kilobytes of data every second

FAPE's OTP encrypts 8.7727 kilobytes every second

Test results are based on speed, for each file size the file is encrypted using AES and another copy of it also encrypted using FAPE's OTP algorithm, and the speed of execution is measured in seconds. For example, given a file of size 63 kilobytes, AES encrypts it in 12 seconds while FAPE's OTP encrypts it in 10 seconds. The test is carried out repeatedly for increasing file sizes after which the encryption throughput is calculated;

$$Te(AES) = \frac{\sum_{i=1,..,5} Pi}{\sum_{i=1,..,5} E1i} = 7.1747$$

$$Te(FAPE) = \frac{\sum_{i=1,..,5} Pi}{\sum_{i=1,..,5} E2i} = 8.7727$$

B. *Decryption*

Advanced Encryption Standard (AES) and the FAPE's OTP are used to decrypt text files of varying size and the result is given below

Table II: Decryption Test

i	Input size (kb)[Pi]	AES [D1i]	FAPE's OTP [D2i]
1	63	12	7
2	125	22	14
3	249	38	25
4	498	86	52
5	995	195	103
Total	1930	353	201
	Throughput (Megabyte s/sec)	5.4674	9.6019

NB: AES decrypts 5.4674 kilobytes of data every second FAPE's OTP decrypts 9.6019 kilobytes every second

Test results are based on speed, for each file size the file is decrypted using AES and another copy of it also decrypted using FAPE's OTP algorithm, and the speed of execution is measured in seconds. For example, given a file of size 63 kilobytes, AES decrypts it in 12 seconds while FAPE's OTP decrypts it in 7 seconds. The test is carried out repeatedly for increasing file sizes after which the decryption throughput is calculated;

$$Td(AES) = \frac{\sum_{i=1,..,5} Pi}{\sum_{i=1,..,5} D1i} = 5.4674$$

$$Td(FAPE) = \frac{\sum_{i=1,..,5} Pi}{\sum_{i=1,..,5} D2i} = 9.6019$$

The result of the encryption and decryption tests carried out above indicate that the proposed algorithm has a higher speed of operation in comparison to the advanced encryption standard, a major factor that improves the performance is the use of multiple threads of execution that work simultaneously.

ENCRYPTION IN THE CLOUD

Implementing the cloud for data storage enables clients to store their files at remote locations, where these files can be retrieved at any time on any device with Internet connection, this section discusses the framework for integrating FAPE's OTP with the cloud.

As the file to be stored is uploaded by the client, the file is transmitted using the Transport Layer Security (TLS) which is a protocol that handles data transmission between clients and servers, it authenticates the severs and the clients and then encrypts the messages transmitted between the authenticated parties using public key cryptography [12]. After transmission is carried out the file is encrypted with FAPE's OTP algorithm, after the encryption process is executed the encrypted file is stored on the database while the file containing the encryption/decryption key is stored on a separate database. The database where the key is stored must be an offline database, so that in the event of a cloud breach the attacker will only have access to the encrypted files and not the key. When the user requests for the document, the encrypted file and key are retrieved from their respective locations and the decryption process is carried out. After decryption the plaintext is transferred to the client through TLS, when a client deletes a file from the cloud storage the files holding the key and the encrypted text are permanently destroyed to ensure that an encryption key is only used once which is a distinctive feature of the One-time Pad.

CONCLUSION

In conclusion cloud computing offers clients a range of services which include on-demand delivery, broad network access, rapid elasticity and measured service but also at the risk of information breach. Security has always been at the core of safe computing practices and so security within cloud computing is an especially worrisome issue. The notion of a perfectly secure system will always be an illusion because attempts to compromise systems are made every day, and as a result efforts must be continually made to improve security. It is the job of the cloud service provider to secure client data effectively because “attack only get better not worse”. The application of encryption to combat data storage issues is a strongly advised one but regular improvements need to be implemented. A 512 bit key might seem sufficient as of today but processing speed improves continually which necessitates that continuous revisions be implemented as updates.

References

[1] Kuyoro, S. O., Omotunde, A. A., Ajaegbu, C. A., & Ibikunle, F. (2012, July). Towards Building a Secure Cloud Computing Environment. *International Journal of Advanced Research in Computer Science, 3*(4).

[2] Devipriya, M., & Sasikala, G. (2015, June). A New Technique for One Time Pad Security Scheme with Complement Method. *International Journal of Advanced Research in Computer Science and Software Engineering, 5*(6), 220-223.

[3] Widiasari, I. R. (2012, November). Combining Advanced Encryption Standard (AES) and One Time Pad (OTP) Encryption for Data Security. *International Journal of Computer Applications, 57*(20).

[4] Jayashree, K., Santoshi, P., & Lande, B. (2015, July). Two Level Encryption based on One Time Pad and Koblitz Method of Encoding. *International Journal of Computer Applications, 122*(15).

[5] Menezes, A., PC, O. V., & Vanstone, S. (1996). *Handbook of applied cryptography.* Boca Raton, FL: CRC Press.

[6] Ogundele, O., Adetunmbi, A., Adewale, O., Alowolodu, O., & Alese, B. (2013). Elliptic Curve Cryptography for Securing Cloud Computing Appliations. *International Journal of Computer Applications*, 10-18.

[7] Bajpai, S., & Srivastava, P. (2014). A Fully Homomorphic Encryption Implementation on Cloud Computing. *International Journal of Information & Computation Technology*, 811-816.

[8] Louk, M., Lim, H., & Lee, H. J. (2014). Security System for Healthcare Data in Cloud Computing. *International Journal of Security and Its Applications, 8*(3), 241-248.

[9] Saini, G., & Sharma, N. (2014, March). Triple Security of Data in Cloud Computing. *International Journal of Computer Science and Information Technologies, 5*(4), 5825-5827.

[10] Khan, S. S., & Tuteja, P. (2015, January). Security in Cloud Computing using Cryptographic Algorithms. *International Journal of Innovative Research in Computer and Communication Engineering, 3*(1), 148-155.

[11] Elminaam, D. S., Kader, H. M., & Hadhoud, M. M. (2009). Performnce Evaluation of Symmetric Encryption Algorithms. *Communications of the IBIMA, 8*, 58-65.

[12] Microsoft Technet. (2003, March 28). *What is TLS/SSL?* Retrieved from Microsoft:https://technet.microsoft.com/enus/library/cc784450(v=ws.10).aspx