NJBAS

Available online at http://www.ajol.info/index.php/njbas/index
Nigerian Journal of Basic and Applied Science (June, 2019), 27(1): 81-88
DOI: http://dx.doi.org/10.4314/njbas.v27i1.11

ISSN 0794-5698

Comparative Analysis of Some Efficient Data Security Methods among Cryptographic Techniques for Cloud Data Security

***[1]M. A. T. Abubakar, [2]A. Aloysius, [1]Z. Umar and [3]M. Dauda**

[1]Department of Computer Science, College of Science and Technology, Waziri Umaru Federal Polytechnic Birnin Kebbi, Nigeria
[2]Department of Computer Science, Faculty of Physical Sciences, Akwa Ibom State University, Nigeria
[3]Department of Computer Science, Faculty of Computing, Federal University, Dutse
[*Corresponding Author: E-mail: musaibnabubakar@gmail.com; ☎:+2348038482438]

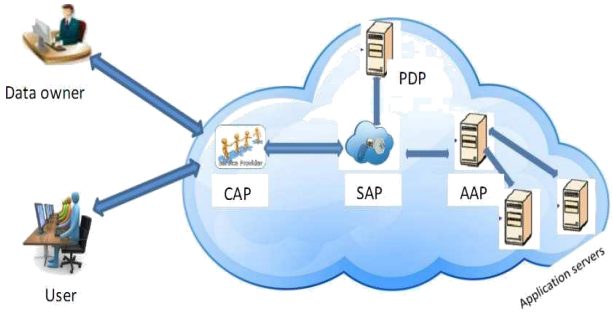
ABSTRACT: The concept of cloud computing model is to grant users access to outsource data from the cloud server without them having to worry about aspects of the hardware and software management. The owner of the data encrypts it before outsourcing to a Cloud Service Provider (CSP) server for effective deployment of sensitive data. Data confidentiality is a demanding task of cloud data protection. Thus, to solve this problem, lots of techniques are needed to defend the shared data. We focus on cryptography to secure the data while transmitting in the network. We deployed Advanced Encryption Standard (AES) used as encryption method for cloud data security, to encrypt the sensitive data which is to be transmitted from sender to receiver in the network and to decrypt so that the receiver can view the original data. Arrays of encryption systems are being deployed in the world of Information Systems by various organizations. In this paper, comparative analysis of some various encryption algorithms in cryptography have been implemented by comparing their performance in terms of stimulated time during Encryption and decryption in the network.

Keywords*:* AES, Data Control, Data Privacy, Data Storage, Encryption Algorithms, Verification.

INTRODUCTION

According to Armbrust *et al.* (2010), the cloud could be data centre resources that comprise hardware and software which provide access to network services on request to a shared pool of needed computing resources. The common characteristics of cloud computing include; (i) pay-per-use (ii) elastic capacity (iii) self-service interface and (iv) resources that are abstracted or virtualized.

A comprehensive functional component of the framework for security in cloud computing as given in Figure 1 depicts three service models in cloud computing namely; Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). Additional models highlighted by Mandal *et al.* (2012) include: Authorized Users, Data Provider, Communication Access Point (CAP), Security Access Point (SAP), Application Access Point (AAP), and Application Servers, as functional components.



CAP-- Communication Access Point , SAP--Security Access Point
AAP--Applications Access Point, PDP--Policy Decision Point

Figure 1: Functional Components of Cloud Computing Security Architecture (Prakash *et al.,* 2014)

In summary, all the described security actions are initiated by the SAP server, and then forwarded to the PDP server. The feedback is sent back to the SAP server, which then grants access to the application server for the service requested. Therefore, the requested services are achieved promptly and clearly to the user (Alanazi *et al.,* 2010). The user does not need to know these actions as they occur, except if there is an attempt of illegal access.

Cloud Computing moves the application software and databases to the large data centers, where the management of the data

and services may not be fully trustworthy. This unique attribute, however, poses many new security challenges which have not been well understood. There are issues of trust amongst all the parties involved, namely, data owner, cloud service provider, and the authorized users. To solve this problem, a repository is built to facilitate the data integration and sharing across cloud along with preservation of data confidentiality. To achieve this, an encryption technique is used to provide data security on data storage. Trusted third party becomes a helping block to ensure shared trust between the authorized user and data owner with cloud service provider.

Qiang *et al.* (2009) proposed an RSA signature authenticator for verification with data dynamic support. Their work broadens the bilinear aggregate signature technique to adequately deal with numerous auditing tasks, while using a third party as auditor to execute the multiple auditing tasks at the same time. Their approach makes possible public audibility without having to retrieve the data blocks from the server.

The job of defining access policies and dynamic updates is the most difficult in data sharing systems such as cloud. Hur (2013) clarifies the cryptographic-based solution for data sharing using cipher text policy attribute-based encryption (CP-ABE) to improve data security. Here, the access policies on the data to be shared are defined by the data owner. The major disadvantage of this approach is that the key can be unauthorized users to decrypt encrypted data.

Applications and data in cloud computing are managed by the service provider and data owner. In order to gain secure access to these data and applications, Mohamed (2012) proposed a model for data security. A single default gateway is provided as a platform in the model to secure user data over public cloud software. Sensitive data is encrypted by the gateway by means of an encryption algorithm before pushing to the cloud server. Only authenticated users can have access to the data, though the service provider can circumvent the data owner and grant access to unauthorized users. The access key is not properly managed in this method; hence, security is poorly handled.

Dubey *et al.* (2012) developed a system using RSA and MD5 algorithms to prevent unauthorized users from gaining access to data from the cloud server, thus increasing revenue and the level of connectivity to cloud users from cloud computing model. The major problem of this method is that both the cloud service provider and the data owner have the same level of control over data with the load computation for the service provider being directly proportional to the level of connectivity. This gives room for degrading in system performance

Besides, for the real use of complex data from CSP, the data owner encrypts before subcontracting to the cloud server. Meanwhile, to defend data in cloud, data privacy is the stimulating duty. In order to address this problem, Prakash *et al.* (2014), proposed an efficient data security method using cryptographic techniques. Thus, the proposed method not only encrypts the sensitive data, but also detects the dishonest party to access the data using combined hash functions which analyzed in terms of storage, communication and computational overheads. Therefore, the main stumbling block of the hash functions method in network security is a one way method, which leads to the degrading of the sensitive encrypted data in a cloud.

This work demonstrates the use of AES as the efficient algorithm for use in Cloud Storage System and compared its performance of encrypts technique based on the analysis of stimulated time at the time of encryption and decryption on the network.

MATERIAL AND METHOD

The cloud computing system architecture is made up of functional blocks, namely, cloud service provider (CSP), data owner, authorized users, and trusted third party as indicated in Figure 2 (Dinh *et al.* (2013).

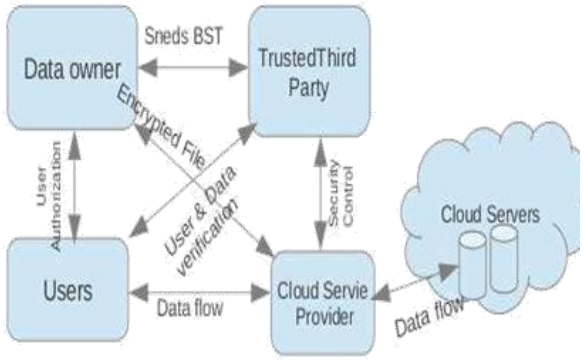


Figure 2: Block diagram of Cloud Computing System Architecture (Prakash *et al.* 2014)

Users: these are all unique clients of the data owner who have been authorized to remotely access cloud data through services offered by trusted third parties.

Data owner: authorized institution that produces data to store at the public cloud model data center for use by any authorized-praying user as the need for such data arises.

Cloud Service Provider: an entity offering cloud infrastructure and management of such to the data owner for storage of outsourced data on the basis of the amount paid. It also takes charge of authenticating third parties before making stored cloud data accessible to them based on demand.

Trusted Third Party (TTP): TTP is an entity that is trusted by all other stakeholders of the system. These include the Cloud Service Provider, data owner, and users. TTP is to verify whether the user requested is being authorized or not, also updating the block status table of the file and calculating the hash value for the file are all parts of the function of TTP (Armbrust *et al.* 2010).

AES ALGORITHM

1. Setup for Key and File Pre-Processing Algorithm Setup (File, Key)

Input: Source file and 256 bits key

Output: number of data blocks (m) and key rotation setup

Step1: Divide the source file in to block of equal size.

Number of blocks (m) = file size/block size

Step2: create array list of 16 nodes for key rotation and store one character in each node.

Step3: Create a simple data encoding map table for all the characters

The AES encryption and decryption algorithm is summarized in Figure 3 and 4 below.

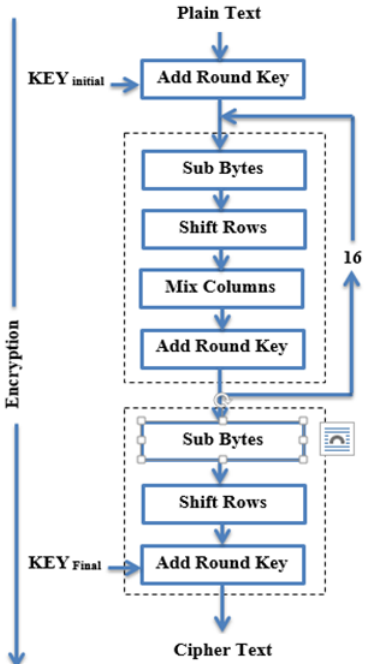


Figure 3: AES Encryption Process

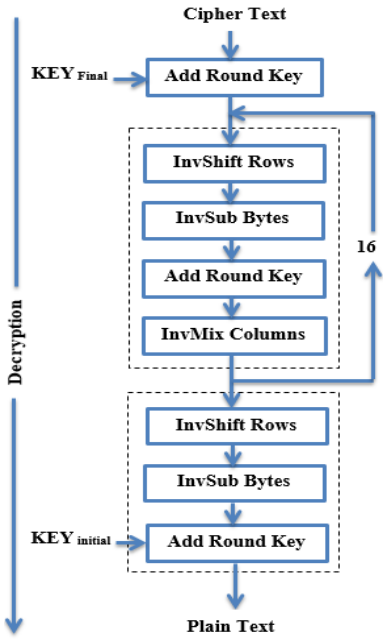


Figure 4: AES Decryption Process

The AES Algorithm is capable of supporting data combinations of 128 bits and key lengths of 128, 192, and 256 respectively. The number of rounds that the algorithm can do depends on the key length; 10 rounds for 128 bit length, 12 for 192, and 14 for the 256 bit key length.

The decryption process simply uses inverse function to reverse all the steps performed during encryption as indicated in Figure 4 above.

2. Data Encryption Algorithm

Algorithm Data Encryption (Source file(f), key, Decrypted file)

Input: Source file and key

Output: Decrypted file

Step 1: Split the characters of the file/string into chunks 128 characters.

Step 2: Get the binary equivalent of the current Character (process character by character of chunks)

Step 3: Remove the first character from the binary value and store it into first character variable and consider the rest of binary value for shift operations.

Step 4: Store binary value into an array list for bit operations.

Step 5: Select a key character from i^{th} position, such a way that if 1^{st} chunk character is selected for encryption, then selects the first character of the key, so i = 1.

Step 6: If selected chunk character greater size of key (16), get the modulus of chunk character position.

Step 7: Add the stripped off first character to the resultant binary value of circular array after bit operations.

Step 8: Add the selected i^{th} key character and $(i + 2)^{th}$ key character.

Step 9: Right shift the binary bits in circular array by (added value mod 5), if mod 5 is 0 then do by 2.

Step 10: Add the selected i^{th} key character value and its previous $(i - 1)^{th}$ character value for instance: if 1^{st} key character is selected then the previous character would the 16^{th} character (key is stored into circular array list for this operations).

Step 11: Get the character equivalent of the binary value.

Step 12: Get the mapped value from encoding Map, which is the decrypted value of the character.

Step 13: Repeat step 2 through 12 for all the chunks with different key (by shifting the key character right using Circular array list).

3. Data Decryption Algorithm

Algorithm Data Decryption((Decrypted file), key, Source file(f))

Input: Encrypted file and KEY for encryption 16 characters

Output: Source file

Step 1: Split the characters of the file/string into chunks 128 characters.

Step 2: Get the binary equivalent of the current Character (process character by character of chunks)

Step 3: Remove the first character from the binary value and store it into first character variable and consider the rest of binary value for shift operations.

Step 4: Store binary value into a circular array list for bit operations.

Step 5: Select a key character from i^{th} position, such a way that if 1^{st} chunk character is selected for encryption, then selects the first character of the key, so i = 1. If elected chunk character greater than size of key (16) get the modulus of chunk character position.

Step 6: Add the stripped off first character to the resultant binary value of array list after bit operations.

Step 7: Add the selected i^{th} key character and $(i + 2)^{th}$ key character.

Step 8: Right shift the binary bits in circular array list by (added value mod 5), if mod 5 is 0 then do by 2.

Step 9: Add the selected i^{th} key character value and its previous $(i - 1)^{th}$ character value for instance: if 1^{st} key character is selected then the previous character would the 16^{th} character (key is stored into circular array list for this operations).

Step 10: Get the character equivalent of the binary value.

Step 11: Get the mapped value from encoding Map, which is the decrypted value of the character.
Step 12: Repeat step 2 through 11 for all the chunks with different key (by shifting the key character right using Circular array list).

RESULT AND DISCUSSION

The NetBeans IDE 8.1, which allows the runs of the AES algorithms; demonstrate the use of AES as the efficient algorithm for use in Cloud Storage System was used in evaluation experiment. The evaluation of AES has been carried out by using the Array Data Structure. The objective of this paper is to produce secure environment for the users of Cloud Storage System, which involves reducing the time of Encryption and Decryption at the time of storage on the network.
The experiment was run 15 times, that produced identical results as expected from both the Encryption and Decryption algorithms. The results obtained are presented in Figure 5.

It can be seen from Figure 5, the average time is 1555621560394 ms for both encryption and decryption execution, and there is no change on memory usage during 15 times executions. Meanwhile, the total Memory was 15, Free Memory was 12, Used Memory was 2 and Maximum Memory was 247.
To implement the AES performance of encrypt technique based on the analysis of stimulated time at the time of Encryption and Decryption on the network with the performance of Rivest-Shamir Adleman (RSA) and Data Encryption Standard (DES).

AES is chosen based on the comparison made from various cryptographic techniques including asymmetric and symmetric methods. RSA was chosen in asymmetric method, while in symmetric DES, and AES were selected.

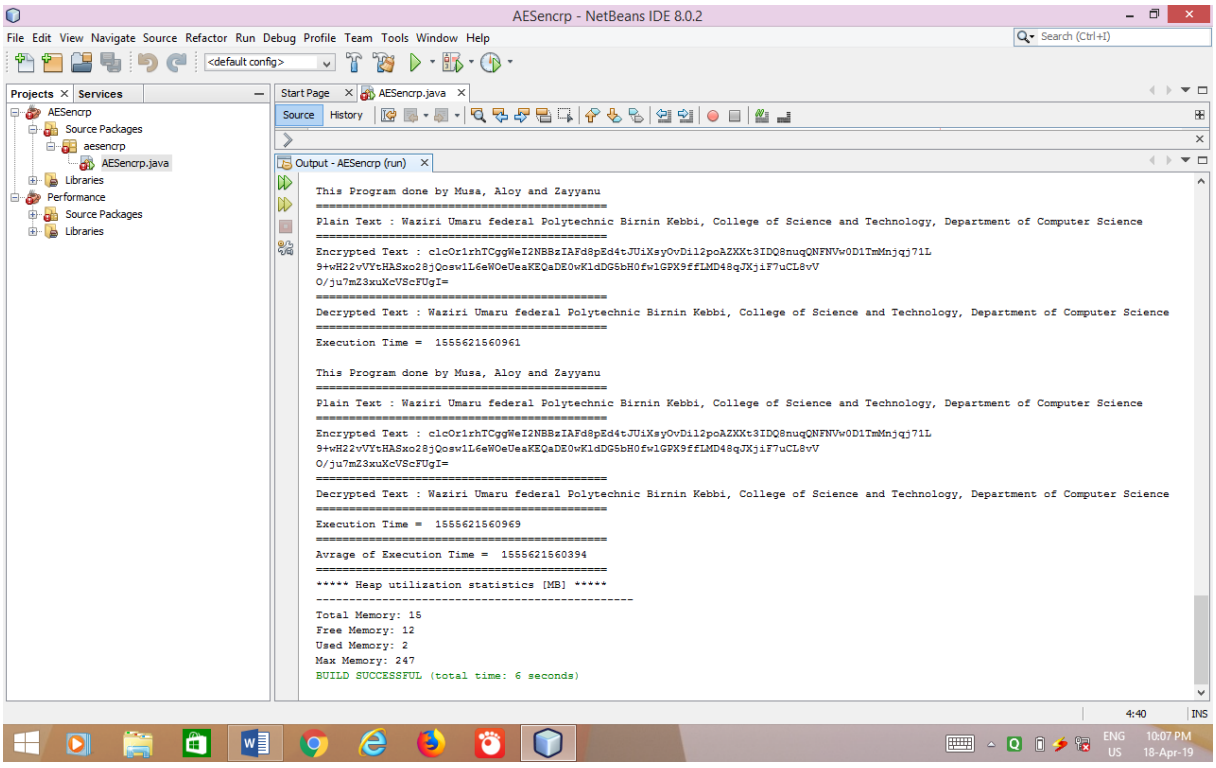


Figure 5: AES Algorithm Result

The parameters used in comparison are as follows:

a. **Key Length:** AES has the longest key length of maximum 256 bits, compared to DES and 3DES which have maximum key lengths of 56 and 168 bits respectively, while that of RSA is dependent on the number of bits in the modulus n, where n = p*q.

b. **Rounds:** AES has the highest number of possible rounds compared to the others.

c. **Block size:** AES compared to the others has a block size of 128 bits, while DES have 64 bits each, and that of RSA is of variable size, depending on the modulus.

d. **Ciphertext:** in terms of ciphertext, AES, and DES all are symmetric family, with RSA only belonging to the asymmetric family.

e. **Speed:** AES performs fastest compared to the others.

f. **Security:** on security rating, AES has excellent security, while RSA is the least secure.

Four experiments were carried out with four text files of different sizes, and comparisons were evaluated based on performance of three algorithms, namely, AES, DES and RSA. They were analyzed based on such parameters as Encryption Time and Decryption Time.

The encryption time is considered as the time that an encryption algorithm utilized to produce a cipher text from a plain text. Earliest Encryption time is used to calculate the throughput of an encryption scheme is calculated as the total plaintext in bytes encrypted divided by the encryption time vice-visa.

Thus, comparisons analyses of the results of the selected different encryption scheme are performed (Najeet and Gaurav, 2012).

Experimental result of the comparison of three Encryption algorithm AES, DES and RSA are shown in Table 1, using same text file for four experiments.

Table 1: Comparisons of AES, DES and RSA of Encryption and Decryption Time

S/NO	Algorithm	Packet Size (KB)	Encryption time (Sec)	Decrypting time (Sec)
	AES		1.6	1
1	DES	153	3	1.1
	RSA		7.3	4.9
	AES		1.7	1.4
2	DES	196	2	1.24
	RSA		8.5	5.9
	AES		1.8	1.6
3	DES	312	3	1.3
	RSA		7.8	5.1
	AES		2	1.8
4	DES	868	4	1.2
	RSA		8.2	5.1

As shown in Table 1, the time taken by RSA algorithm is much higher for both encryption and decryption process compared to the time taken by AES and DES algorithm.

The graphs in Figures 6 and 7 indicate the time taken for encryption and decryption of file sizes of variable length by the three algorithms. As indicated in the graphs, RSA algorithm utilized much longer time compare to time utilised by AES and DES algorithm. AES and DES algorithm show very minor differences in time utilising for encryption and decryption process. This is in agreement with Das. and Misra (2011) who reported that, AES algorithm can be utilized not only for security but also for great speed on the network. This implies that both hardware and software available on the network also respond faster. However, AES is a new encryption standard recommended by NIST to replace DES and 3DES. More so, it can be implemented on many platforms such as small devices and carefully tested for many applications on the network.

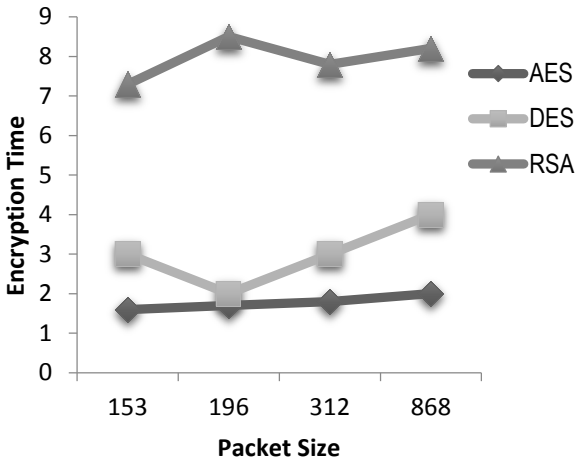


Figure 6: Performance of AES over DES, and RSA based on Encryption Time

Figures 6 and 7 indicate that of the three algorithms analysed, the RSA takes the longest time for encryption and decryption of the varied packets. Earliest, AES and DES algorithms show very minor difference in time taken for encryption and decryption processes. This result is in line with (Lu *et al*., 2010) whose findings reveals that in any cloud computing, the service provider is responsible for maintaining the confidentiality, integrity and availability (CIA) of the instances on a hardware level while the user is responsible for protecting the CIA on a higher level, such as the content of files and the operating system.

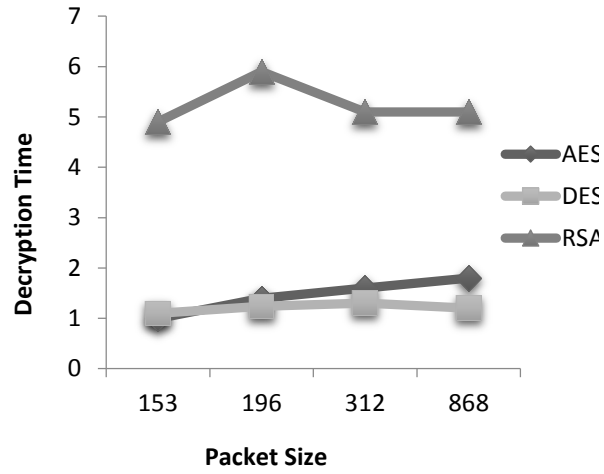


Figure 7: Performance of AES over DES, and RSA based on Decryption Time.

CONCLUSIONS

This paper examined data security issues in cloud storage environment and proposed AES to be the cryptographic technique. A comparison result indicates that AES is a better security technique for encrypted and decrypted data to store in the CSP. Encryption algorithm plays very significant part in communication security. The performance of these encryption techniques based on the text files used and the results showed that AES algorithm significantly consumes least compared to RSA algorithm that consume longest in terms of encryption time. On the other hand, Decryption time AES algorithm was found to be better than other algorithms. Furthermore, simulation result reveals that AES algorithm is much better than that DES and RSA algorithms in terms of the stated criteria.

ACKNOWLEDGEMENT

The authors are grateful to Kareem Abbas Dawood in the Department of Software Engineering Universiti Putra Malaysia for the role he played during the research work carried out in the Faculty Laboratory, Master of Computer Science, Faculty of Computer Science and Information Technology.

REFERENCES

Alanazi, H. O., Zaidan, B. B., Jalab, H. A., Shabbir, M., Al-Nabhani, Y. (2010). New Comparative Study between DES, 3DES and AES within Nine Factors. *Journal of Computing*, **2**(3): 152-157.

Armbrust, M., Stoica, I., Zaharia, M., Fox, A., Griffith, R., Joseph, A. D., Rabkin, A. (2010). A view of cloud computing. *Communications of the ACM*, **53**(4), 50.

Barsoum, A., and Hasan, A. (2013). Enabling Dynamic Data and Indirect Mutual Trust for Cloud Computing Storage Systems. *IEEE Transactions on Parallel and Distributed Systems,* **24**(12), 2375-2385.

Das D. and Misra R. (2011) Programmable Cellular Automata Based Efficient Parallel AES Encryption Algorithm. *International Journal of Network Security & Its Applications (IJNSA)*, 3(6): 204

Dinh, H. T., Lee, C., Niyato, D., Wang, P. (2011). A survey of mobile cloud computing: Architecture,

Applications, and approaches. Wireless Communications and Mobile Computing, 13(18), 1587-1611.

Dubey, A. K., Dubey, A. K., Namdev, M., Shrivastava, S. S. (2012). Cloud-user security based on RSA and MD5 algorithm for resource attestation and sharing in java environment, *CSI Sixth International Conference on International Journal of Advanced Computer Research (IJACR) 2(1), 10.*

Hur, J. (2013). Improving Security and Efficiency in Attribute-Based Data Sharing. *IEEE Transactions on Knowledge and Data Engineering*, **25**(10), 2271-2282.

Lu, R., Lin, X., Liangand, X., Shen, X. (2010). Secure provenance: the essential of bread and butter of data forensics in cloud computing, In: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, ASIACCS pp. 282–292. ACM, New York.

Mandal, A. K., Parakash, C., Tiwari, A. (2012). Performance evaluation of cryptographic algorithms: DES and AES. In: Proceeding of 2012 IEEE Students' Conference on Electrical, Electronics and Computer Science, 1-5.

Mohamed, E. M., Abdelkader, H. S., El-Etriby, S. (2012). Enhanced data security model for cloud computing. In: Proceedings of the 8th International Conference on Informatics and Systems (INFOS), 12-17.

Prakash G L, Dr. Manish Prateek, Dr. Inder Singh (2014). Data Security Algorithms for Cloud Storage System using Cryptographic Method, *International Journal of Scientific & Engineering Research*, 5(3): 54 – 61.

Raj, J. (2016):
http://www.cse.wustl.edu/~jain/cse473-16/
(accessed 2017).

Seth, S. M., and Mishra, R. (2011, June). Comparative Analysis of Encryption Algorithms for Data Communication. *International Journal of Computer Science and Technology*, **2**(2), 292-294.

Wang, C., Ren, K., Lou, W., Li, J. (2010). Toward publicly auditable secure cloud data storage services, *IEEE Network*, **24**(4), 19-24.

Wang, Q., Wang, C., Li, J., Ren, K., Lou, W. (2009). Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing.In: Proceeding of 14th European Symposium, Research in Computer Security (ESORICS 09), 355-370.