An Analysis the Big Data Security Based on Hadoop Framework using Hybrid Encryption Technique

Vijay Kumar Yadav[1]*, Anoop Kumar Chaturvedi[2]

[1] Research Scholar, LNCT

[2] Professor, LNCT

Abstract - Cloud integration helps in distributing data over different cloud storage and helps application programs to access and process data using different computation mechanisms. Hadoop is based on two main modules: Mapreduce for processing and generating large data sets and Hadoop Distributed File System (HDFS) for storing data on distributed clusters. Hadoop has been commonly accepted in the field of cloud computing where resource utilization and system performance require an excellent task scheduling mechanism. AES- 128 in ECM mode will be used to encrypt the OTP algorithm's key, which will then be used to encrypt HDFS data blocks utilizing the suggested technique. Using the OTP algorithm, we can ensure that the plaintext & the asymmetric key are the same length (i.e., 128 bits). Using encryption & decryption with a variety of keys, cryptographic methods are largely used to protect data on the cloud. Hybrid-Key Stream Cipher Mechanism is a proposed method (HKSCM). Python will be used for the actual implementation of the proposed algorithm. For practical implementation we use Linux OS, Containerization, various Clouds platforms.

Keywords - Cloud, Hadoop, Hybrid-Key Stream Cipher Mechanism, Encryption, Decryption

- x -

INTRODUCTION

Cloud computing, which offers users on-demand access to a shared pool of shared software, data storage, & processing power, is receiving a lot of attention from a variety of sectors at the moment. Because of the constant influx of new data into the digital sphere, which necessitates vast amounts of space, processing power, & system throughput, a plethora of cloud computing frameworks have recently emerged to handle such massive amounts of information. One well-known cloud computing framework is Google's (GFS and MapReduce). The predictive capacity analysis & Big Data traits were both improved by the use of data analytics to identify potential threats. Limitations in customizability, security, cost, performance, interoperability, availability, vendor lock-in, &compliance are just a few of the cloud problems that users must face. In recent years, the quantity of digital data has ballooned from a few terabytes to several beta bits. Therefore, it requires a toolkit of methods for extracting meaning from large, heterogeneous data sets.

Hadoop, one of the newest technological trends, is a framework for cloud storage. It is an open-source distributed computing system built on Java, & it consists of two modules: the MapReduce data processing framework & HDFS. HDFS is for storing data on distributed clusters of machines, while MapReduce is for processing on enormous data sets; it enables users to employ parallel thousands of commodity machines efficiently; and by simply creating map & reduce functions, the user can processing massive data. Hadoop is typically implemented in a massively parallel computing environment or in a public cloud service, such as those provided by Yahoo!, Facebook, Twitter, & Amazon. Hadoop's popularity attests to its scalability, yet it lacks built-in protections for user data.

Security in the Hadoop project is implemented using lightweight systems for managing who can access which files. As a result, HDFS files stored in Data nodes, as well as data transferred between Data nodes, during the execution of MapReduce operations, are best protected by encryption.

RESEARCH OBJECTIVE

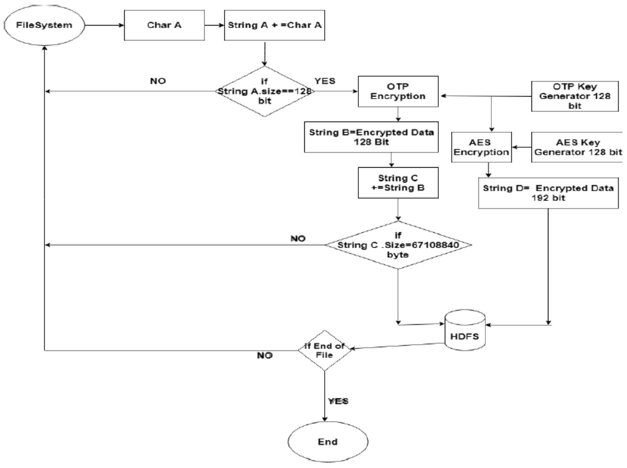
- To study Hadoop's security vulnerability & protect important documents secure.
- To proposed hybrid encryption system which combines HDFS files with two security encryption algorithms (namely, AES & OTP) to speed up the upload/download of encrypted data.

- To proposed algorithm will be implemented by using Python programming. Hybrid-Key Stream Cipher Mechanism (HKSCM) is a proposed method.

PROPOSED METHODOLOGY

According to the proposed algorithm, the OTP key will be encrypted using the AES algorithm and included in the encrypted file. This will improve the OTP algorithm, and then the security of the file is become robust. In the same time the size of the encrypted file is not increased instead of using block cipher algorithms. The encryption and decryption processes according to our proposed mechanism will be discussed in the following sections.

The proposed mechanism will encrypt HDFS data blocks using the OTP algorithm and the Key of the OTP algorithm will be encrypted using the AES- 128 with ECM mode algorithm. Th OTP algorithm is used to keep the plaintext in the same length as the asymmetrically key length (i.e., 128 bits). Then, the user keeps the private key of the AES to decrypt the OTP key in order to decrypt the whole file.


FileSystem
Char A
String A + =Char A
NO
if String A.size==128 bit
YES
OTP Encryption
OTP Key Generator 128 bit
String B=Encrypted Data 128 Bit
AES Encryption
AES Key Generator 128 bit
String C +=String B
String D= Encrypted Data 192 bit
NO
if String C .Size=67108840 byte
HDFS
if End of File
NO
YES
End

According to Fig. 1, the client requests to upload data file to HDFS. The application server will generate the random key with 128bit for the OTP algorithm which will be encrypted by AES algorithm with ECM mode to be 192 bits. Then, a stream of 67,108,840 bytes (64MB – 192 bit) will be encrypted with OTP algorithm and added to the encrypted key then will be sent to the HDFS. This will be continuing for all data needs to be stored in the HDFS

RESULTS AND DISCUSSION

The emergence of big data and the popularity of big data analytics necessitates the need for cloud based storage and processing frameworks. The massive amount of data is stored in cloud and is also effectively shared among authorized users. Once data is moved to cloud the data owner do not have direct control over data and so there must be security and privacy mechanisms to ensure integrity of data stored in cloud. The big dataset in cloud storage is prone to intentional or accidental attacks and hardware failures. Thus securing data in cloud storage is inevitable and is the most considerable issue to solve. The sensitive or private data on cloud must be protected at any cost because the leakage of it leads to heavy damage to the data owner. Cloud storage security is a major concern in all the deployment models of cloud such as public, private, and hybrid cloud.

ECC & AES—A Combined Hybrid Proposed Approach

ECC & AES, we now have the most cutting-edge and effective cloud-based cryptographic method available. Due to the greater key size required by single AES, the hybrid (ECC-AES) approach is preferable for data encryption due to its faster encryption mechanism and less key size requirements. When AES encrypts with ECC, the key size is decreased due to ECC's primary property of small key size, and the encryption speed is boosted. Standards for both encryption and decryption keys are used in ECC to accomplish these goals of key reduction and safe key generation. To further strengthen the security of your data, you should utilize ECC in conjunction with AES. Data encryption and decryption are automatically generated by ciphertext once the key size has been determined. AES makes use of the key generated by ECC. The proposed method at cloud storage is compatible with the combined effect of both ECC and AES to get the protected system. This aids in the reduction of the size of secure data storage. The proposed algorithm's block diagram is shown in Figure 1 below.

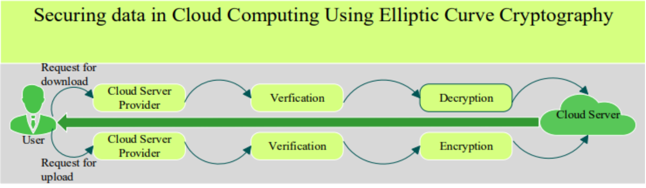

Securing data in Cloud Computing Using Elliptic Curve Cryptography

Figure 1: Hybrid Approach of Elliptic Curve Cryptography - Advanced Encryption Standard

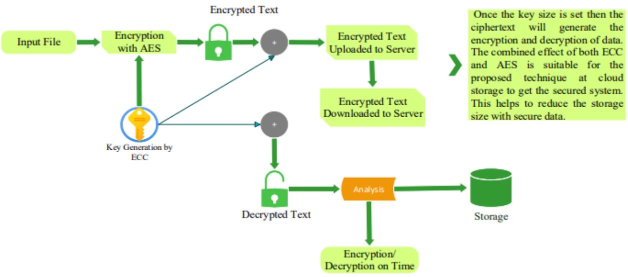

Encrypted Text
Input File
Encryption with AES
Encrypted Text Uploaded to Server
Encrypted Text Downloaded to Server
Key Generation by ECC
Decrypted Text
Analysis
Storage
Encryption/ Decryption on Time
Once the key size is set then the ciphertext will generate the encryption and decryption of data. The combined effect of both ECC and AES is suitable for the proposed technique at cloud storage to get the secured system. This helps to reduce the storage size with secure data.

Figure 2: Representation of ECC and AES algorithm

As can be seen in the preceding diagram, using both AES & ECC to encrypt data in the cloud is an excellent way to prevent unauthorized access. An innovative new diagram is presented, illustrating how the suggested solution ensures the safety of user data during transmission to the server and during storage as well, all thanks to the usage of encrypted data. Novelty can also be measured in terms of the time & effort it takes to run a computer program.

1. First level encryption

The 128bits sequence of blocks is divided into 4 x 4 bytes-arrays for matrix operation. AES is secure and fast. It can extend a basic block cipher into a stream cipher to encrypt arbitrary length data. AES algorithm supports 128, 192, or 256-bit data and 10-, 12-, or 14-key rounds.

Alice and Bob share the private key to decrypt the data. -key derivation generates the key (KDF). KDF creates cryptographic keys from a password, secret question, or random number.

Created keys protect electronic data in transit or storage. KDF key generation is pseudorandom, which prevents eavesdroppers from using suggestions to access electronic data. Given an input string, such as a password, it will confine a pointer to the true string source, either KDF or a random string of the same length.

KDF generates a private key for encryption K e to convert m to Cipher text c and a private key for decryption K d to convert c back to m where K e =K d.

2. Next-level encryption

ECC algorithm takes over AES encryption for cloud applications. ECC protects cloud data from hackers and snoopers. ECC uses discrete logarithms to generate a public encryption key and a private decryption key. Any non-vertical line intersects ECC in at most three points. An elliptic curve over a field k is a nonsingular cubic curve in two variables, f(x, y) =0. k can be complex, real, an algebraic extension of rational numbers, or a finite set.

3. Hybrid Encryption Level

Unifying the two algorithms ensures security and reduces hacker data loss. Figure shows how the ECC algorithm takes over the AES encryption by encrypting the AES key in the cloud.

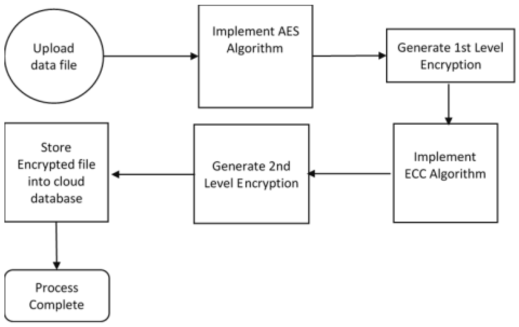

Upload data file
Implement AES Algorithm
Generate 1st Level Encryption
Implement ECC Algorithm
Generate 2nd Level Encryption
Store Encrypted file into cloud database
Process Complete

Figure 3: Block Diagram Multilevel Encryption

During file download, inverse ECC decrypts the AES Key and inverse AES decrypts the cipher text. Figure shows multilevel decryption.

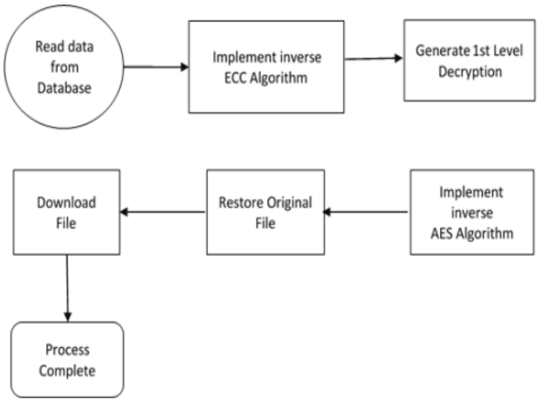

Read data from Database
Implement inverse ECC Algorithm
Generate 1st Level Decryption
Implement inverse AES Algorithm
Restore Original File
Download File
Process Complete

Figure 4: Decryption Block Diagram

The figure shows the proposed model for HDFS cloud applications with hybrid encryption, where AES encrypts data first and ECC encrypts the AES key to enhance cloud data security. ECC encrypts only the AES key in a bit to speed up multi-level encryption.

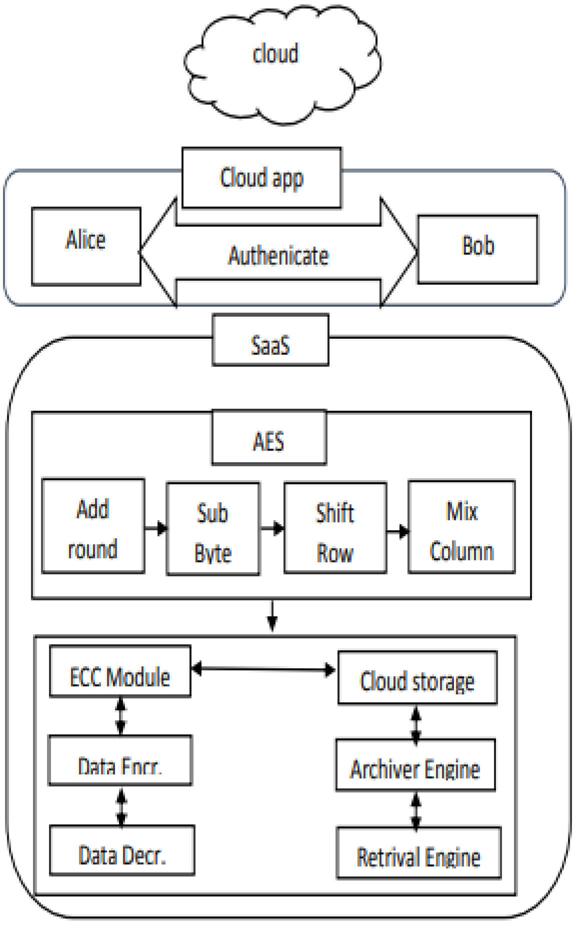


Figure 5: Proposed model

Web-based cloud computing stores data on external servers. A performance evaluation of the privacy model is computed to determine the effectiveness of the proposed scheme, the hybrid encryption of AES symmetric and ECC asymmetric cryptographic algorithms, using these parameters: encryption and decryption time, throughput that calculates algorithm efficiency, and cipher-text to plain-text ratio. Using a symmetric and an asymmetric algorithm to improve cloud data security is best practice. The AES algorithm was chosen because it's fast, cost-effective, and has a medium memory size. ECC was chosen alongside AES because it's better than other asymmetric algorithms in speed, key length, and computational time. In the experiment, some virtual machines were dedicated to ECC services to reduce interference. Elliptic Curve Admission (ECAs) is needed to handle more user applications. Load balancers send ECAs authentication requests and tokens. The ECA load balancers only authenticate and generate authentication keys for their own application users.

Comparing 192-bit AES and 384-bit ECC ciphers. The hybrid scheme's lightweight feature uses the least likely key numbers. 288 128-bit AES and 160-bit ECC keys were used to accomplish the same task as 192-bit AES and 384-bit ECC.

Table1: Encryption & Decryption Time for multilevel

Size (kb) 20kb 0.25 0.17	Encryption time (seconds)	Decryption Time (seconds)
20 kb	0.26	0.16
65 kb	0.89	0.74
40 kb	0.38	0.29
45 kb	0.51	0.35
30 kb	0.3	0.29

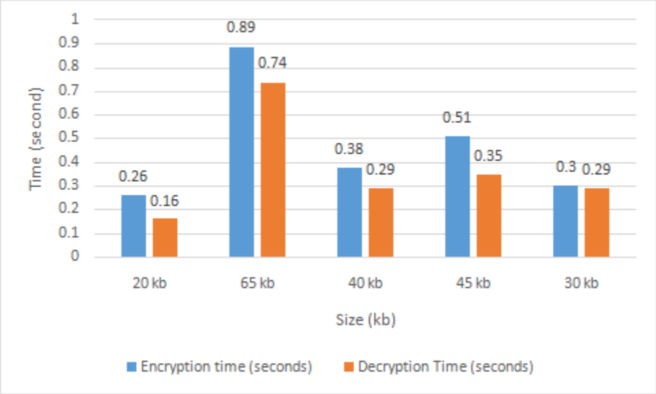


Figure 6: Size Vs. Time Comparison

Size is represented on the horizontal axis, while the various time of encryption & decryption are indicated on the vertical axis. Time required for encryption & decryption is proportional to the data's size.

A hybrid task requires 192 AES bits and 384 ECC bits, or 128 AES, 160 ECC, and 288 bits. Table 2 compares the hybrid model encryption time to using AES and ECC separately in a cloud application. Even though the AES algorithm alone takes less time, it is not as secure as the proposed system because if a hacker decrypts one level, it will be difficult to decrypt the second level using the same algorithm.

Table 2: Comparing the hybrid model encryption time to using AES and ECC

			Model time
20 kb	0.24	0.5	0.41
65 kb	0.85	1.5	1.54
40 kb	0.37	0.74	0.69
45 kb	0.49	1.01	0.82
30 kb	0.32	0.73	0.59
AVG time	0.4	0.1039	0.821
AVG size	45	45	45
Throughput	79	39.3	42.30

Findings are significant in that the hybrid ECC-AES strategy requires less time to decode data than the existing alternatives due to its smaller key size, making it the preferred model for suggested encryption utilizing AES & ECC. In addition, the hybrid ECCAES method combines features from both techniques, resulting in increased security thanks to the system's complexity and resistance to attacks. It's also easy to notice that the suggested hybrid method requires substantially less time to decrypt than previous algorithms do. Decryption now takes less time, which means less money spent on computations on our end. Our method is therefore superior to others.

Encryption Time

For additional validation, we compared our suggested hybrid method to both previously-existing techniques (AES) & time required to encrypt & decrypt using a variety of key sizes. Different keys, including 192-bit AES and 384-bit ECC ciphers, were used in the experiments. The lightweight component of the hybrid system makes use of the least likely key numbers. This was accomplished with 288 128-bit AES keys & 160-bit ECC keys instead of the more conventional 192-bit AES & 384-bit ECC keys.

DISCUSSION

The proposed approach for HDFS cloud applications with hybrid encryption is discussed in this chapter. Unlike other multi-level encryption methods, ECC only encodes the AES key in a single bit. Using the following metrics (encryption and decryption times, throughput which measures algorithm efficiency, and cipher-text to plain-text ratio), we compute a performance evaluation of the privacy model to ascertain the efficacy of the proposed scheme, a hybrid encryption of the AES symmetric and ECC asymmetric cryptographic algorithms. The best way to ensure the safety of data stored in the cloud is to use both symmetric & asymmetric algorithms. This can be broken down into 128 AES bits, 160 ECC bits, and 288 bits for a hybrid task. Although the AES method alone is faster, it is not as safe as the suggested approach since a hacker who breaks the first level of encryption will have a hard time breaking the second level using the same algorithm. The consumer is afforded a wide variety of conveniences thanks to cloud services. Different types of cloud services cater to different types of users. The ability to access one's data from anywhere is a huge convenience for many users, and the services supplied are offered at a minimal cost. Since you don't need to carry about your personal device, cloud services could be made available on any system. The weakness of cloud services is their lack of data security, but this can be compensated for by employing additional precautions. In order to simplify the key generation process, we use error-correcting code (ECC). ECC's enhancement is superior to those of other cryptographic methods because of the small size of its keys. Using both AES & ECC together can greatly improve data optimization & security. But more cryptographic safety measures will be required in the future to fully realize the cloud computing concept's potential. More secure implementations of the hybrid method can be used to advance this study in the future. Incorporating additional levels of protection into the system is a great way to increase its effectiveness & efficiency.

CONCLUSION

Hadoop is a free and open-source software framework for handling large data volumes on a dependable and scalable network of computers. The low-priced, fast-processing, error-resistant, adaptable Hadoop is typically utilized in big clusters or public cloud services.In order to simplify the key generation process, we use error-correcting code (ECC). ECC's enhancement is superior to those of other cryptographic methods because of the small size of its keys. Using both AES and ECC together can greatly improve data optimization and security. But more cryptographic safety measures will be required in the future to fully realize the cloud computing concept's potential.The study also compares and analyzes the security, efficiency, performance, and resistance to the aforementioned assaults of the asymmetric key cryptography RSA, the symmetric key cryptographic AES, and Hybrid Encryption-RSA. Despite cloud's numerous benefits, it does have drawbacks, and one of those is the potential lack of security.

REFERENCES

1. Alexandre C. B. Delbem, Telma W. de Lima and Guilherme P. Telles, „Efficient Forest Data Structure for Evolutionary Algorithms Applied to Network Design", IEEE Transactions On Evolutionary Computation, vol. 16, No. 6, December 2012, pp. 829-846.
2. Anjali Patel, Nimisha Patel, Dr. Hiren Patel, „Secure Data Sharing Using Cryptography in Cloud Environment", IOSR Journal of Computer Engineering (IOSR-JCE), vol. 18, No. 1, January-February 2016, pp. 58-62.
3. Ayad Ibrahim, Hai Jin, Ali A. Yassin, DeqingZou and PengXu, „Towards Efficient Yet Privacy-Preserving Approximate Search in Cloud Computing", The Computer Journal, vol. 57, No. 2, February 2014 , pp. 1-14.
4. B.Preethi, M.Shahin and K.RamaDevi, „Synonym Query Using Multikeyword Search Using Cloud Computing", International Journal of Advances in Engineering, 2015, vol. 1, no. 3, pp. 192 – 195.
5. BojanSuzic, Andreas Reiter, Florian Reimair, Daniele Venturi, Baldur Kubo, „Secure Data Sharing and Processing in Heterogeneous Clouds", Procedia Computer Science, Elsevier, vol. 68, 2015 , pp. 116 – 126.
6. Boyang Wang, YantianHou, Ming Li, „Maple: Scalable Multi- Dimensional Range Search over Encrypted Cloud Data with Tree-based Index", ASIA CCS '14 Proceedings of the 9th ACM symposium on Information, computer and communications security, June 2014, pp. 111-122.
7. Buyun Sheng, Chenglei Zhang, Xiyan Yin, Qibing Lu, Yuan Cheng, Ting Xiao and Huimin Liu, „The International Journal of Advanced Manufacturing Technology, Springer, April 2016, vol. 84, No. 1–4, pp. 103–118.
8. C.J. Stam, P. Tewarie , E. Van Dellen, E.C.W. van Straaten, A. Hillebrand and P. Van Mieghem, „The trees and the forest: Characterization of complex brain networks with minimum spanning trees", International Journal of Psychophysiology, Elsevier, vol. 92, No. 3, June 2014, pp. 129–138.
9. CengizOrencik, AyseSelcuk, ErkaySavas and Murat Kantarcioglu, „Multi-Keyword search over encrypted data with scoring and search pattern obfuscation", International Journal of Information Security, vol. 15, No. 3, June 2016, pp. 251–269.
10. Chang Liu, Rajiv Ranjan, Chi Yang, Xuyun Zhang, Lizhe Wang and Jinjun Chen, „MuR-DPA: Top-down Levelled Multi-replica Merkle Hash Tree Based Secure Public Auditing for Dynamic Big Data Storage on Cloud", IEEE Transactions on Computers, vol. 64, No. 9, September 2015, pp. 2609 – 2622.
11. Chen Lyu, Shi-Feng Sun, Yuanyuan Zhang, Amit Pande, Haining Lu and DawuGu, „Privacy-Preserving Data Sharing Scheme over Cloud for Social Applications", Journal of Network and Computer Applications, Elsevier, vol. 74, October 2016, pp. 44-55.
12. Chen Yang, WeimingShen, Tingyu Lin and Xianbin Wang, „A hybrid framework for integrating multiple manufacturing clouds", The International Journal of Advanced Manufacturing Technology, Springer, September 2016, vol. 86, No. 1–4, pp. 895–911.
13. Cheng Guo, NingqiLuo, MdZakirulAlamBhuiyan, YingmoJie, Yuanfang Chen, Bin Feng and Muhammad Alam, „Key-Aggregate Authentication

Cryptosystem for Data Sharing in Dynamic Cloud Storage‟ , Future Generation Computer System, Elsevier, August 2017, pp. 1-30.

14. Cheng-Kang Chu., Sherman S. M. Chow., Wen-GueyTzeng, Jianying Zhou., and Robert H. Deng., „Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage,‟ IEEE Transactions on Parallel and Distributed Systems, vol. 25, No. 2, February 2014, pp. 468 –477.
15. Cong Wang, Sherman S.-M. Chow, Qian Wang, KuiRen, and Wenjing Lou, „Privacy-Preserving Public Auditing for Secure Cloud Storage‟, IEEE Transactions on Computers, vol. 62, No. 2, February 2013, pp. 362 – 375.
16. D. Chandramohan , T. Vengattaraman, D. Rajaguru and P. Dhavachelvan, „A new privacy preserving technique for cloud service user endorsement using multi-agents‟, Journal of King Saud University - Computer and Information Sciences, Elsevier, vol. 28, No. 1, January 2016, pp. 37-54.
17. Daniel Díaz-Sánchez, FlorinaAlmenarez, Andrés Marín, DavideProserpio, and Patricia Arias Cabarcos, „Media Cloud: An Open Cloud Computing Middleware for Content Management‟, IEEE Transactions on Consumer Electronics, vol. 57, No. 2, May 2011, pp. 970-978.

Corresponding Author

Vijay Kumar Yadav*

Research Scholar, LNCT