International Journal of Enhanced Research in Science, Technology & Engineering
ISSN: 2319-7463, Vol. 6 Issue 5, May-2017

Secure Data Storage and Virtualization At Cloud Data Center

Priya Rathore[1], Mahesh Kumar[2]

[1]M.Tech Scholar, Department of Computer Science and Engineering, MRKIET, Rewari, Haryana, India
[2]Associate Professor, Department of Computer Science and Engineering, MRKIET, Rewari, Haryana, India

ABSTRACT

Now a Days when people are becoming more dependent on Internet, cloud computing is getting increasingly popular. Cloud computing is an important paradigm, with the potential to significantly reduce costs through optimization and increased operating and economic efficiencies. Also, Cloud computing significantly enhances collaboration, agility, and scale, thus enabling a truly global computing model over the Internet infrastructure. Virtualization is one of the concepts that provide flexibility to host multiple operating system stacks on a single hardware. In this paper, we will introduce an efficient cryptographic method which is cost and time efficient both so that the encrypted data is more secure at cloud data center and also maintain integrity of virtual machine of VM's disks. This method support coming soon for-Nettle : for use in TLS, Nettle already supports the method for non-TLS use; this will bring its support to TLS, Googlebot - Google's web , mbed TLS crawler, tink - a small crypto library that provides a safe, simple, agile and fast way to accomplish some common crypto tasks, Zcash - a decentralized and open source crypto-currency using groundbreaking cryptography. A cryptographic network system has to guarantee confidentiality and integrity, and also it has to support random access. To attain desired purpose, presently existing designs mainly rely (on ad-hoc manner) on combination of merkle hash tree with a block cipher mode of encryption.

Keywords – data integrity, virtualization, data security, cloud computing.

1. INTRODUCTION

Now a Days, Resources are being shared around the globe via Internet and to attain it cloud computing is used. In the service of cloud computing, third party cloud service provider offers user to store their data to cloud & cloud data center also offers best quality service to enjoy data on demand.

Cloud services categories:-

1. Infrastructure as a service - It is the most extensible delivery model and provides few, if any, application-like features
2. Platform as a service - The goal is to enable developers to build their own applications on top of the platforms provided.
3. Software as a service - In this service, providers typically enable services with a large number of integrated features, resulting in less extensibility for customers

Cloud infrastructure also introduced a service called DAAS whose solutions provide advantages like agility, cost-effectiveness and data quality. Now, few issues in cloud computing related to security are: Multi-tenancy, Issues in verifying and authenticating the data, insecurity in virtualization, Load Balancing, etc.

Although Cloud offers tremendous number of advantages, still lots of security concerns are there like the major one i.e. storage security.

The integrity of the data has to be looked upon seriously in order to gain trust and satisfaction of the user. However, maintaining security is a challenging task. Cloud computing have already introduces authorize virtualization to remove load balancing issue. Also, Virtualization also have large number of security related issues:- virtual machine escape , utilization of resources etc .In this Paper, we proposed an alternative approach to protect important data stored at cloud in virtual machine disks. This research will make the following contributions:- First we describe an efficient

cryptographic method to provide security. Further we describe Merkle Hash Tree to maintain integrity by using SHA - 256 Hash Function and Dynamic Merkle Hash Tree to maintain large hash values.

.

2. RELATED WORK

Large amount of work for data security at cloud has been done. Pearson [1] discussed in deep the privacy challenges that software engineers face when targeting the cloud as their production environment to offer services and then a deep literature discussed on cloud security by- Gu and Cheng [2] also Siebenlist [3].

Hassan Takabi [4] presented unique issues of cloud computing that exacerbate security and privacy challenges in clouds and a transparent, backward-compatible approach that protects the privacy and integrity of customers' virtual machines on commodity virtualized infrastructures, even facing a total compromise of the virtual machine monitor (VMM) and the management VM by Zhang in cloud visor concept [5].

Deswarte in [6], uses hash function which is RSA based, for verifying the file stored at the remote server. The limitation of this scheme lies in the computational complexity at the server which must exponent all the blocks in the files.

Schwarz and Miller [7] proposed a technique in which they used erasure-correcting coding to safeguard the stored data and use algebraic signatures hash functions with algebraic properties for verification. In this technique, a function is used to store the fingerprint of the file block and then verifies if the signature of the parity block is same as the signature of block. The Drawback of this method is that the computation complexity at client side and server side takes place at the cost of linear combination of file blocks.

Poonam Pardeshi and Deepali Borade[8] make use of Public-Auditing & proposed a way in which the Merkle Hash Tree used in a method called RSASS, is made dynamic by using the concept of relative index to compute the index of leaf node quickly and a dynamic operation scheme based on this tree structure for cloud storage. Also, AES algorithm is used instead of RSA algorithm because AES requires less encryption/decryption time as well as less buffer space as compared to RSA algorithm. But, this scheme has disadvantages of differential cryptanalysis of SHA -1 hash function and cryptanalysis of AES 128 bit scheme by side channel attacks.

Ateniese [9] introduced a highly efficient and provably secure PDP technique based entirely on symmetric key cryptography, while not requiring any bulk encryption. His PDP technique also allows outsourcing of dynamic data. It is Provably-secure scheme for remote data checking. Few limitations of this scheme are: No support provided for dynamic auditing, requires more than 1 kilo-byte of data for a single verification, etc.

A scheme called, "Proofs of Retrievability" proposed by Juels and Kalisiki [10] focuses on static archival of large files. To ensure data Possession and retrievability, it makes use of spot checking and error correcting codes.POR scheme cannot be used for public databases; it is suitable only for confidential data. But this scheme has several limitations like Dynamic updation which is prevented due to the introduction of sentinel nodes, Pre-processing of each file is needed prior to storage at the server, etc.

The scheme proposed by Erway [11] is a auditing protocol that can support the dynamic operations of the data on the cloud servers. It requires the linear combination of data blocks to be sent for verification to the auditor. The scheme makes use of a TPA for integrity verification. It also supports data dynamics via the most general forms of data operation, such as block modification, insertion and deletion. The main drawback is that the scheme may leak data content to the auditor

Yunchun [12] make a comparative research analysis of existing research work regarding the data security and privacy protection techniques used in the cloud computing.

Shacham and Waters [13] design an improved POR scheme with full proofs of security in the security model defined in. They use publicly verifiable homo-morphic authenticators built from BLS signatures, based on which the proofs can be aggregated into a small authenticator value, and public retrievability is achieved. Still, the authors only consider static Data files.

Wang [14] updated the previous methods by introducing revised version of BLS scheme to enhance security aspects with classical version of MHT to attain integrity. But in this model, classical MHT with revised BLS scheme is used to manage hash values of large files blocks at their leaf level

3. PROPOSED METHOD

In our proposed Scheme, we described a member of Salsa 20 family to achieve privacy for sensitive data stored at

cloud along with Merkle Hash Tree to achieve integrity and protection to data stored on virtual machine's disk

In this Paper, we describe a method called chacha20 for encryption. It is considerably faster than AES in software-only implementations, making it around three times as fast on platforms that lack specialized AES hardware and also not sensitive to timing attacks.

Basics of proposed method [15] [16] [17] [18] has input state matrix and round functions. It has relevant characters to "column round" and "diagonal round", input matrix of 16 words pattern, 20 rounds (4 sub rounds to each so total 80 rounds)) are executed on original input state matrix by altering column and diagonal matrix on every 16 words of new plaintext data. After that execute the sum operation on input state matrix with output of 20 rounds where this output is arranged and XORed with 16 words of plaintext to generate encrypted message.

Quarter Round Function-

Quarter round on 4 words are performed in following manner

a= a + b d=d XOR a d<<<=16
c= c + d b=b XOR c b<<<=12
a= a + b d=d XOR a d<<<=8
c= c + d b= b XOR c b<<<=7

Figure 1: Quarter Round Function

Below Diagram shows working of proposed method:

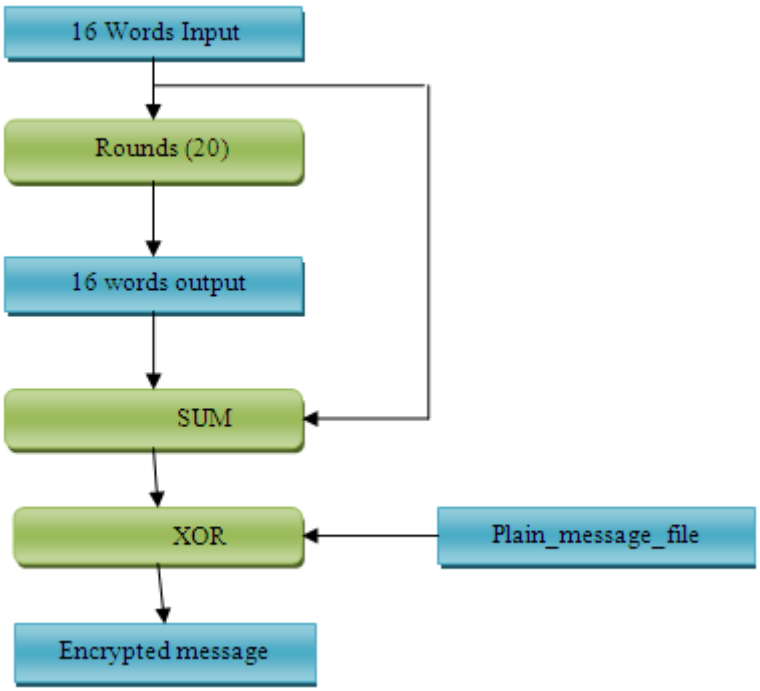


Figure 2: Working of chacha20 method

Algorithm:-

1. ALGORITHM_Encryption(key, counter ,nonce, Plain_message_file)
2. for counter = 1to (cell (len (plain_message_file) / 64))
3. key_stream = ALGORITHM_Block (key, counter, nonce)
4. Block = plain_message_file [(counter*64) …..(counter*64 - 1)]
5. encrypted_message = encrypted_message + (Block $\oplus$ key_stream)
6. end
7. if ((cell (plain_message_file%64)! = 0))
8. key_stream = ALGORITHM_Block (key, counter, nonce)
9. Block = plain_message_file [(counter*64) ….. len (plain_message_file) -1)
10. encrypted_message = encrypted_message + (Block $\oplus$ key_stream)[0…..len (plain_message_file%64)]

11. end
12. return encypted_message
13. end

SHA-256 Hash Function

SHA 256 function operates on 512 bit message block and 256 bit intermediate hash value. It is actually a 256 bit algorit hm which encrypts the intermediate hash value using the message block as a key. The message to be hashed is first:

1) Padded with its length in such a way that result is a multiple of 512 bits long & then
2) Parsed into 512 bit message blocks $M^{(1)}$, $M^{(2)}$,…., $M^{(N)}$

The message blocks are processed one at a time. Starting with initial hash value $H^{(0)}$, sequentially compute:

$$H^{(i)} = H^{(i-1)} + C_M^{(i)}(H^{(i-1)}) ,$$

Where C is SHA 256 Compression Function & + means word-wise mod 2^{32} addition. $H^{(N)}$ is the hash of M.

SHA-512 Hash Function

SHA-512 is identical in structure to SHA-256, but:

- the message is broken into 1024-bit chunks,
- the initial hash values and round constants are extended to 64 bits,
- there are 80 rounds instead of 64,
- the message schedule array w has 80 64-bit words instead of 64 32-bit words,
- to extend the message schedule array w, the loop is from 16 to 79 instead of from 16 to 63,
- the word size used for calculations is 64 bits long,
- the appended length of the message (before pre-processing), in *bits*, is a 128-bit big-endian integer, and
- the shift and rotate amounts used are different.

Merkle Hash Tree

An authentication tree which provides a method for making public data available by using a tree structure in conjunction with a suitable hash-function. Merkle organize and then maintain all hash values of disk data/ virtual machine images to shield the originality and ordering of disk data

.

Dynamc Merkle Hash Tree:

When users want to modify their data stored at cloud then it requires dynamic change in cloud storage to update the same. For this Reason, we introduced dynamic version of MHT [19] named merkle b+ hash tree. This combination has worst case complexity as O(log n) instead of O(n) for dynamic modification or updation.

It stores hash values H(M1),H(M2),…..,H(Mn) at leaf level [20]. At the leaf level each node have parts as left, right, middle and rank i.e. number of dependent descents. Our scheme has:

Generation of Key: Function of key_generation takes a security token as input and produces public and private keys respectively.

Preparation: This function create blocks of data files in encoded form at leaf level of MBHT and produce sets of signatures as outputs.

Generation of Challenge: Function of challenge_generation generates sets of queries and a random value for each index of block set at client side. These quires consists collection of IDs .Finally, Client sends set of IDs to server. After receiving these IDs, server checks integrity of data file's block .

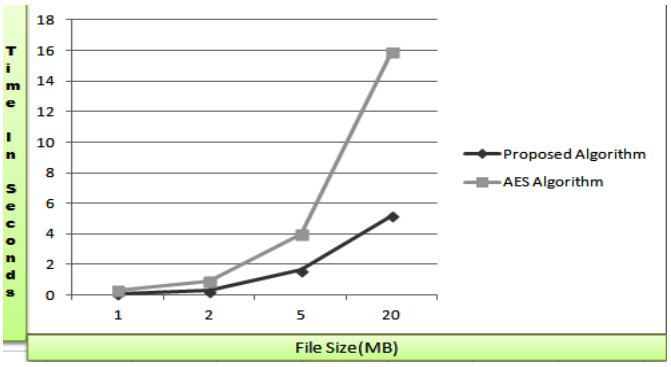
Generation of Proof: Server authenticates integrity of queries by generating proofs.

Verification: User executes Verification function to check integrity of blocks after receiving proof and gives output as true or false.

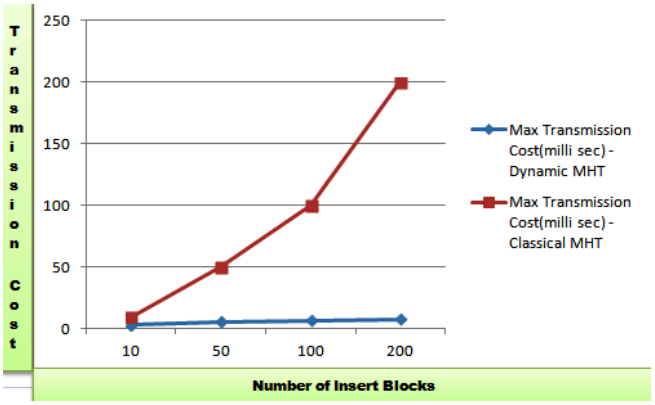
Updation Process: For any dynamic updation, client runs update request function which generates modification request. It takes order and sends this modification request in form of add, delete or modify to server.

4. PERFORMANCE ANALYSIS

1. Below fig. proved that proposed method is less time consuming and beneficial then that of AES 128 bit algorith m as it graphically represents the time required on different file sizes.



2. Below figure describes about the performance of Classical MHT and dynamic MHT on data file/blocks on leaf level of hash tree and message transmission cost.



CONCLUSION

Our proposed method is to resolve problems related to security and integrity issues by using a member of salsa20 which is very secure and fast encryption method and dynamic version of merkle hash tree. It works with hash function which provides proper dynamic modification to users in worst case complexity to remain protected under chosen message attack. In future, our system can be further enhanced by providing more efficient dynamic updation method to attain better rapid modification in constant time.

REFERENCES

[1]. Pearson, Siani. "Taking account of privacy when designing cloud computing services." Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing. IEEE Computer Society, 2009.

[2]. Gu, Lin, and Shing-Chi Cheung. "Constructing and testing privacy-aware services in a cloud computing environment: challenges and opportunities."Proceedings of the First Asia-Pacific Symposium on Internetware. ACM, 2009.

[3]. Siebenlist, Frank. "Challenges and opportunities for virtualized security in the clouds." Proceedings of the 14th ACM symposium on Access control models and technologies. ACM, 2009.

[4]. Takabi, Hassan, James BD Joshi, and Gail-Joon Ahn. "Security and privacy challenges in cloud computing environments." IEEE Security & Privacy 6 (2010): 24-31.

[5]. Zhang, Fengzhe, Jin Chen, Haibo Chen, and Binyu Zang. "CloudVisor: retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization." In Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, pp. 203-216. ACM, 2011.

[6]. Deswarte, Yves, Jean-Jacques Quisquater, and Ayda Saïdane. "Remote integrity checking." Proceedings of IICIS 140 (2004): 1-11.

[7]. Schwarz, Thomas SJ, and Ethan L. Miller. "Store, forget, and check: Using algebraic signatures to check remotely administered storage." Distributed Computing Systems, 2006. ICDCS 2006. 26th IEEE International Conference on. IEEE, 2006. DOI: 10.1109/ICDCS.2006.80

[8]. Pardeshi, Poonam M., and Deepali R. Borade. "Enhancing Data Dynamics and Storage Security for Cloud Computing using Merkle Hash Tree and AES Algorithms." International Journal of Computer Applications 98.21 (2014). DOI: 10.5120/17304-7734 .

[9]. Ateniese, Giuseppe, et al. "Scalable and efficient provable data possession."Proceedings of the 4th international conference on Security and privacy in communication netowrks. ACM, 2008.

[10]. Juels, Ari, and Burton S. Kaliski Jr. "PORs: Proofs of retrievability for large files." Proceedings of the 14th ACM conference on Computer and communications security. Acm, 2007

[11]. Erway, C. Chris, et al. "Dynamic provable data possession." ACM Transactions on Information and System Security (TISSEC) 17.4 (2015): 15.

[12]. Sun, Yunchuan, et al. "Data security and privacy in cloud computing." International Journal of Distributed Sensor Networks 2014 (2014).

[13]. Lombardi, Flavio, and Roberto Di Pietro. "Secure virtualization for cloud computing." Journal of Network and Computer Applications 34.4 (2011): 1113-1122.

[14]. Wang, Qian, et al. "Enabling public verifiability and data dynamics for storage security in cloud computing." Computer Security–ESORICS 2009. Springer Berlin Heidelberg, 2009. 355-370.

[15]. R.Dharavath, R.Mishra, and Bhukya Shankar Nayak. "Cha-Cha 20: Stream Cipher Based Encryption for Cloud Data Centre." Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies. ACM, 2016.

[16]. Nir, Yoav, and Adam Langley. "ChaCha20 and Poly1305 for IETF protocols."draft-nir-cfrg-chacha20-poly1305-01 (work in progress) (2014).

[17]. Bernstein, Daniel J. "ChaCha, a variant of Salsa20." Workshop Record of SASC. Vol. 8. 2008.

[18]. Procter, Gordon. "A Security Analysis of the Composition of ChaCha20 and Poly1305." (2014).

[19]. Zheng, Qingji, and Shouhuai Xu. "Fair and dynamic proofs of retrievability."Proceedings of the first ACM conference on Data and application security and privacy. ACM, 2011.

[20]. Gilbert, H.; Handschuh, H.: Security analysis of SHA-256 and sisters. In: International Workshop on Selected Areas in Cryptography.Springer, Berlin, Heidelberg, pp. 175–193 (2003).