An Efficient Cloud Based Data Sharing With Homomorphic Encryption Approach

V.Senthil Kumar[1] R.Dhivagar[2] S.Indhurekha[3] Krithikavenkat[4]

[1]Assistant Professor, Department of Computer Science and Engineering, K.S.R.College of Engineering, Anna University,Tiruchengode-637215, Tamilnadu, India.
Email: senthilkmr777@gmail.com .

[2,3,4]Student, Department of Computer Science and Engineering, K.S.R. College of Engineering, Anna University, Tiruchengode-637215,Tamilnadu, India.
Email: dhivagarvagar@gmail.com, rekhaindhu55@gmail.com, krithikavenkat34@gmail.com .

Abstract - Cloud computing suggestively theatres a part in the feature of realsourceoperation and service consumption. Irrespective of the type of clouds (ex. Private, public, hybrid or inter-cloud), every service providers focuses on the data exist in cloud servers. Each and every moment, the researchers and scholars are proposing multiplicity of security algorithms to secure cloud data during the transactions. Most of the cloud data secure algorithms are focusing on the way to secure to cloud data in a single direction by using cryptographic algorithms. In this study paper emphases on a new direction to combine the features of data compression with the cloud data in order to protect the cloud data storage.

Keywords - Cloud, data, storage and compression.

I. INTRODUCTION

Either to consider a Profitable or non-profitable organizations, dedicated resource utilization brings more elevation in the economic impact and makes huge loss. In order to overcome this defect, every clients hunt for a new technology to solve their demand with minimum effort. In this aspect, the cloud computing provides an excellent environment for the resource seekers over the network. Most of the categories, the cloud service providers are not consider the secure way of data transaction under the public cloud. But, in the same time, private cloud give more attention to secure the data resides in its cloud servers as well as to maintain the security for confidential cloud data. The general cloud storage mechanism is comprised with two major components such as data and its applications. Both data and applications are always handling with the help of cloud data owner and cloud service provider [5].

The most challenging task in the cloud data servers are focusing on the handling of residing data under the category of private and confidential sector classification. In order to ensure the secure data in cloud storage by using a cryptographic mechanism apply encryption on the storage sector and decryption on the authenticated receiving sector. In the aspect of applying cryptographic algorithms such as RSA (Rivest, Shamir and Aldimer) and AES(Advanced Encryption Standard) regarding to secure cloud data using its own way of number theoretical information's and key exchange values [2]. The data orinformation is required to store in cloud service provider to keep as it is not guarantee for secure data. Because of this reason, the researchers concentrate on cryptographic mechanism.

It provides only the crypto mechanism as the solution provider in order to secure the cloud data residing in Cloud Service providers. Most of the cloud security algorithms are focusing on the above said algorithms with performance analysis and comparisons of its required amount of storage [3]. The following block diagram (figure 1) depicted in the existing security mechanism in cloud data. In this research paper to frame, a challenging architectural framework as well as to implement the secure cloud data with data compression mechanism.

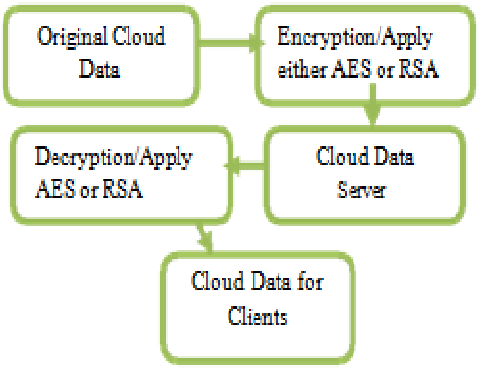

Figure 1: Existing Cloud data secure mechanism

Data compression is used in most of the cases to reduce the original size of the data or information without affect its originality as well as the number of bits. Most of the cases, the compression algorithms are classified in to two major categories: Lossless and Lossy. The Lossy algorithm is indicates few data bits are loss during the retrieval or reproducing environment. At the same time, the Lossless never makes such kind of bits or data loss during the retrieval environment [4].

II. LITERATURE REVIEW

A. Huffman Coding

Huffman coding [7] is an entropy encoding algorithm used for lossless data compression. It practices a exact method for picking the representation for each symbol, consequential in a prefix-free code that couriers the most mutual characters using quicker strings of bits than are used for less common source symbols. Huffman coding is ideal when the chance of individual input symbol is a negative power of two. Prefix-free codes incline to have slight inadequacy on small alphabets, where odds often fall among these optimal points. "Blocking", or increasing the alphabet magnitude by mergingseveral symbols into "words" of static or variable-length earlier Huffman coding, typically helps, particularly when together symbols are linked. Prediction by Partial Matching (PPM) [8,9] is an adaptive numerical data compression system based on setting modeling and prediction. In overall, PPM predicts the probability of a given character built on a given number of characters that closely head it.

Predictions are usually reduced to symbol rankings. The number of preceding symbols, n, regulates the order of the PPM model which is meant as PPM(n). Unbounded variants where the setting has no length limitations also exist and are denoted as PPM*. If no prediction can be finished based on all n context symbols a prediction is struggled with just n-1 symbols. This process is constant until a match is originate or no more symbols remain in context. At that point aimmovable prediction is made. PPM is conceptually simple, but often computationally expensive. Much of the work in optimizing a PPM model is handling inputs that have not already occurred in the input stream[7]. The noticeable way to switch them is to make a "neverseen" symbol which activates the escape order. But what likelihood should be allocated to a symbol that has not ever been seen. This is called the zero-frequency problem. PPMdensityexecutions vary significantly in additional details. The actual symbol selection is typically recorded using arithmetic coding, however it is also possible to use Huffman encoding or smootharound type of dictionary coding practice. The underlying model used in maximum PPM procedures can also be lengthy to predict numerous symbols. The symbol size is regularlystationary, naturallyaonly byte, which kind of generic conduct of any file format easy. The central controller can take advantage of the complete network view supported by SDN [6] to analyze and correlate this feedback from the network.

B. LZ77

LZ77 algorithms attain compression by trading repeated rates of data with positions to aonly copy of that present data past in the input (uncompressed) data stream. A match is fixed by a couple of numbers called a length distance pair, which is equivalent to the statement "each of the next length characters is types behind it in the uncompressed stream". To advert matches, the encoder must keep track of abouttotal of the maximum recent data, such as the last 2 kB, 4 kB, or 32 kB. The construction in which this data is detained is called a sliding window, which is why LZ77 is occasionally called sliding window compression. The encoder wants to have this data to aspect for matches, and the decoder desires to keep this data to read the matches the encoder denotes to. The bigger the sliding window is, the extended back the encoder may search for generating references. It is not individual acceptable but regularly useful to permit length-distance couples to stipulate a length that essentially exceeds the distance. As a copy command, this is confusing: "Go back four characters and copy e10characters from that position into the current position"[7]. How can ten characters be imitative over when only four of them are really in the buffer? Undertaking one byte at a time, there is no problematic serving this request, because as a byte is imitative over, it may be fed again as input to the copy command. When the copy-from placebrands it to the initial destination position, it is therefore fed data that was fixed from the commencement of the copy-from position. The operation is thus corresponding to the statement "copy the data you were given and repeatedly paste it until it fits".

C. LZ78

LZ78algorithms accomplish compression by substituting constant occurrences of data with situations to a dictionary that is constructed based on the input data stream. Each dictionary access is of the form dictionary[...] = {index, character}, where index is the directory to a former dictionary entry, and character is attached to the string represented by dictionary[index]. For example, "abc" would be stored (in reverse order) as follows: dictionary[k] = {j, 'c'}, dictionary[j] = {i, 'b'} , dictionary[i] = {0, 'a'}, where an index of 0 implies the end of a string. The algorithm prepares last matching index = 0 and next available index = 1. For each character of the input stream, the dictionary is examined for a match: {last matching index, character}. If a match is created, then last matching index is usual to the index of the matching entry, and nothing is output. If a match is not found, then a new dictionary entry is created: dictionary[next available index] = {last matching index, character}, and the algorithm outputs last matching index, followed by character, then reorganizes last matching index = 0 and increases next available index. Once the dictionary is full, no more passes are added. When the end in of the input stream is touched, the algorithm outputs last matching index. It is very significant to know that the strings deposited in the dictionary is in the reversed order[11-14]. LZW is an LZ78 based algorithm that uses a dictionary pre-initialized with all thinkable symbols.

The main improvement of LZW is that when a match is not found, the current input stream character is assumed that it will be the first character of an existing string in the dictionary (since the dictionary is initialized with all possible characters), so only the last matching index is output (which may be the pre-initialized dictionary index corresponding to the previous symbol.

III. RELATED WORK

The related work regarding to secure the data is specified in a general frame work not a specific boundary to describe the combination of data compression along with cryptographic mechanism.The following diagram (figure 2) is clearly indicates the existing work modules,

Figure 2: Existing Data security in the combination of compression and encryption

In our research work, we have future a novel model that associations compression algorithm and crypto mechanism. After analyzing the available compression algorithms like Arithmetic coding, Huffman coding, LZ78 and LZW, we have suggested a new compression algorithm, named as Parallelized Sparse Data Compression Algorithm (PSDCA).

It offers significant advantages in saving more storage space by avoiding the wastage of storage locations by eliminating the memory entries with values of “0”. Here, as a unique provision, if "1" exceeds the threshold limit, then "0" will be stored.

Measuring Compression Performances

Classically, the presentation measure proves the effectiveness of a compression technique or otherwise i.e. whether it is companionable with any specific criteria. Based on the nature of the application, the criteria is selected to measure the performance of compression algorithms.

In fact, time complexity and space complexity are usually regarded as the most significant criteria, used interchangeably.

The compression behavior largely is determined by the category of compression algorithm chosen, whether lossy or lossless.

A major advantage of the proposed framework is that it has the potential to reduce the accessing time between cloud clients via cloud servers, apart from less cloud storage on account of wastage by utilizing “NULL” values.

However, a likely drawback could be seen in the unusually longer execution time while matching the missing storage values in the reproduction section due to slow connections in the communication channel. The comparative performance analysis is factored on the cloud storage efficiency, quality of data in retrieval efficiency and time of execution efficiency. The resulting chart describes the relative analysis between various compression algorithms such as Arithmetic Coding, Huffman Coding, LZ78, LZW and PSDCA.

Table 1: Performance Comparisons with PSDCA

Original Data			Compressed Data			
File Name	File Size	File Type	Parallel Arithmetic	Parallel Huffman	Parallel LZW	Purposed

			Coding	Coding	Coding	PSDCA
ERR003 495	1899995 323	PDF	4582469 88	321186 439	279540 234	210822 457
ERR227 988	5697846 47	PNG	5677043 01	567704 301	569606 384	567704 301
ERR009 295	4708114 30	BMP	2176946 10	217694 610	224274 069	199908 837
ERR092 485	6113547 45	MP3	5854670 42	585467 042	588234 117	585467 042
SRR628 456	1075151 912	WA V	6333178 08	633317 843	692954 495	633317 776
SRR624 457	6032274 92	DLL	1531137 67	156648 643	157349 618	133078 188
SRR624 558	2175143 21	C++	2306296 0	260599 23	208534 61	192592 73
SRR624 659	1608147 955	Strea m	7748705 08	766382 271	686669 775	679072 189
SRR624 850	4985746	TXT	1052673	121654 7	110090 0	101460 0
SRR624 951	5045745	TXT	1231402	143366 0	119687 1	118597 5

s

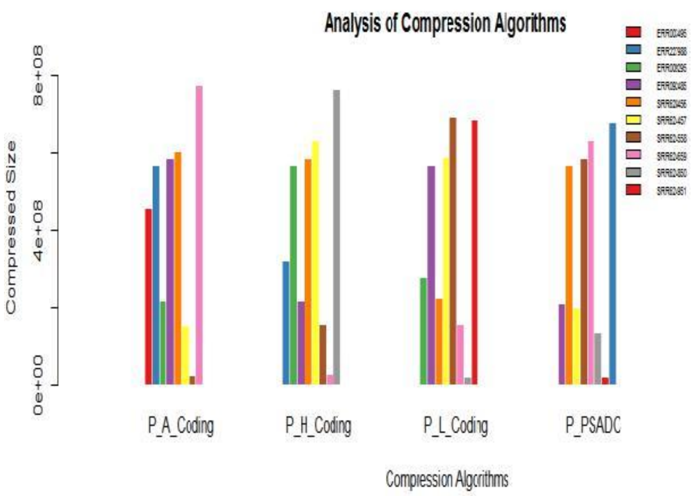

Figure 3: Analysis of compression Algorithms

The security in cloud data sector is not only the responsible for service provider alone; the owner of the data is also participating in order to store the data. Each and every IT infrastructure for cloud service provider maintains their own infrastructure in order to protect the data from unauthorized users. The data storage in the cloud servers as static category may bring a high level of security risk at this time and to create a questioner window for security treat towards its storage. According to the National Institute of Standards and Technology (NIST), the cloud models are classified into a standard way such as: Organizational Model and Service Model [6]. Most of the cases the organizational models are concentrating on the security issues related its storage infrastructures and the service model focus on the security issues related with the service utilization among the different users or clients. It will clearly depict in the following figure 4.

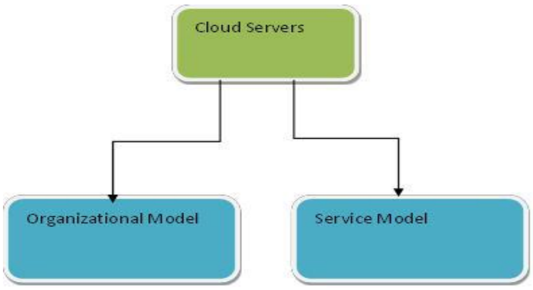

Figure 4: Cloud Service Model Classifications

Whenever to go for the cloud service storage architectural framework, it emphasis on the risk for malicious data as well as the data loss. In order to allocate the storage space for the incoming data or information, in general it assigned a proposer link with the succeeding memory locations. If the data is not a compressed one, it may require more memory space for accommodating within the existing frame and to further encryption mechanism also make a room for excess conflict in the storage infrastructure. In order to avoid such kind of storage space wastage as well as the complication access is eliminate with the help of this proposed work in this research paper. The service model is always linked with the retrieval operation whenever it will require by the cloud service clients or users.

Irrespective of the cloud type, the service related with cloud computing is always considered as software as a service (SaaS), Infrastructure as a Service (IaaS) and Platform as a Service (PaaS). The designed architecture for cloud computing data storage is varying from vendor to vendor based on the applications. In most of the cloud service storage contents are focusing on multimedia data in its environment over the network. This drawback is overcome by this proposed research work publication.

IV. PROPOSED WORK

The data for the transmission is used for the encryption by using Secured Adaptive Block Based Data Encryption is combined with data compression algorithm before it will store into cloud storage. In Adaptive Block Based Data Encryption each block of data is encrypted using a random number. i.e use only one random number at beginning of encryption and that can used recursively to all the data blocks using one-way hash function. This random number must be shared securely between the participants before the encryption by a one pass method.

In order to eliminate the threat for data loss is managed by the implementation of an exact link between the segmented storage element in the cloud server as well as the service providers. In most of the cases, the data retrieval or decoding information face a problem at the situation of inactive intermediate cloud servers. Such demerit or drawback is handled with the help of multiple copy of (replication) compressed as well as encrypted data to make it available in redundancy mechanism.

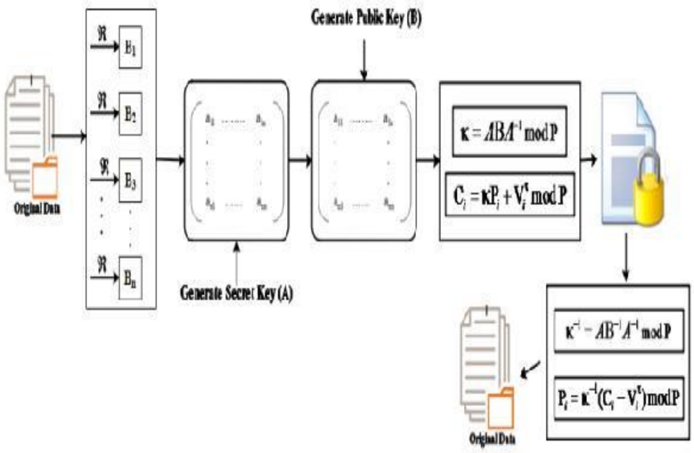

Figure 5: Encryption procedure

The cloud data which one is ready for transmission is encrypted along with compression in default manner and it will transfer towards the cloud server storage location.The number of key attributes and the file sizes are the major factors in encryption and decryption process. The time requirement is more for large file size and even number of attributes increased. Let us discuss the performance analysis of time, security performance and speed for each method.

Table 2: Security performance analysis for maximum number of attributes

No. of Attributes	Security Performance (%)				
	KP-ABE	CP-ABE	CP-ABE-WP	EPPDR	SABBDE
2	59.27	58.97	58.50	62.99	68.99
4	66.75	66.52	67.00	63.74	69.74
6	70.90	73.17	73.25	69.92	78.92
8	78.44	76.62	78.45	76.53	82.53
10	75.19	76.91	76.97	79.14	87.14
12	83.93	83.19	83.41	83.42	92.42
14	83.96	83.26	83.18	85.91	92.91

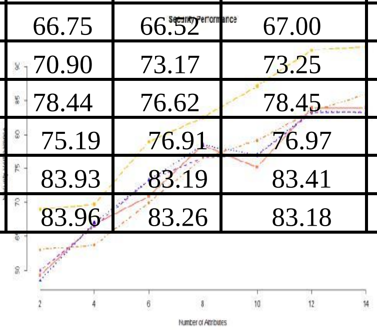

Figure 6: Analysis of different security algorithms

Table 3: Encryption time of different files sizes anddifferent algorithms

File Size	Encryption Time (ms)				
	KP-ABE	CP-ABE	CP-ABE-WP	EPPDR	SABBDE
10MB	93.67	92.45	90.23	89.32	65.89
100MB	189.23	183.38	178.83	172.35	142.44
1GB	484.26	436.45	418.28	389.24	338.49
2GB	784.44	764.87	755.88	684.43	656.29
3GB	1024.56	1113.34	1023.56	968.75	924.56
5GB	2346.58	2289.52	2212.87	1759.35	1649.70

In most cases, the performance of security algorithms are assessed with the help of its run-time efficiency and it will not focus on its storage. In this implementation, it gives an equal heaviness for both run time as well as storage efficiency for the cloud data residing in the cloud data server. The above graph (figure 6) clearly represented the detailed presentation evaluation.

V. CONCLUSION

Security is mostly concerned in each and every data transactions over the internet among the different users. In the same aspect, the researchers are struggling to propose a standard architectural framework in order to save the cloud data residing in cloud service provider with different mechanism. In this research paper provide an excellent roadmap to ensure the cloud data security with the help of compression and encryption mechanism.

REFERENCES

[1] “Securing and Managing Data for Cloud Storage applications using High Throughput Compression (HTC) “, International Journal Of Advanced Scientific and Technical Research , Issue 5 volume 3, May-June ,2015 , ISSN 2249-9954.

[2] “Developing Secure Cloud Storage System by Applying AES and RSA cryptography algorithms with Role based

Access Control Model", International Journal of Computer Applications (ISSN: 0975-8887), Volume 118-No 12, May 2015.

[3] "Secure User Data in Cloud Computing Using Encryption Algorithms" by RachnaArora, AnshuParashar , International Journal of Engineering Research and Applications , ISSN: 2248-9622, Vol-3, Issue 4, Jul-Aug 2013,pp.1922-1926.

[4] "Data Security Using Compression and Cryptography Techniques" by Ruchita Sharma and SwarnalataBollavarapu, International Journal of Computer Applications (ISSN: 0975-8887), Volume 117-No 14, May 2015.

[5] "Data Security Algorithms for Cloud Storage System using Cryptographic Method" byPrakash G L, Dr. Manish Prateek, and Dr. Inder Singh, International Journal of Scientific & Engineering Research, Volume 5, Issue 3, March -2014, ISSN 2229-5518.

[6] Mr.V.Senthil Kumar, "Providing Security against IP crowdsourced spoofing attacks on cloud using TopoguardAlgorthim" , South Asian Journal of Engineering and Technology, pp. 158-165 /158, 2019.

[7] C:\Documents and Settings\DELL\Local Settings\temp\IM\LZSS (LZ77) Discussion and Implementation.mht.

[8] D.A. Huffman, "A Method for the Construction of Minimum Redundancy Codes", Proceedings of the I.R.E., September 1952, pp 1098-1102.

[9] T. Bell, J. Cleary, and I. Witten, "Data compression using adaptive coding and partial string matching," IEEE Transactions onCommunications, Vol. 32 (4), p. 396-402, 1984.

[10] Moffat, Implementing the PPM data compression scheme, IEEE Transactions on Communications, Vol. 38 (11), pp. 1917-1921,November1990**.**

[11] Ziv, J., & Lempel, A. "A Universal Algorithm for Sequential Data Compression," IEEE Transactions on Information Theory, 23(3), pp.337-343, May 1977.

[12] Ziv, J., & Lempel, A. "Compression of individual sequences via variable-rate coding," IEEE Trans. Inform. Theory, 24(5), 530-536, September 1978.

[13] "New mechanism for Cloud Computing Storage Security" by AlmokhtarAit El Mrabti, NajimAmmari, AnasAbou El Kalam, AbdellahAitOuahman OSCARS Laboratory, National School of Applied Sciences, CadiAyyad University ,arrakesh, Morocco,Mina De Montfort ,ARTIMIA, 75 Street Guy Mˆoquet, 92240 Malakoff, France, (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 7, No. 7, 2016.