A Way to Secure the Data in Cloud Data Storage by using Cloud Data Compression Mechanism

[1]Komati Thirupathi Rao and [2]Sheik Saidhbi
[1]*Department of Dean-Academics, K L University, AP, India*
[2]*Bharathiar University, Coimbatore, Tamil Nadu, India*

Key words: Cloud, data, storage, compression, features

Corresponding Author:
Komati Thirupathi Rao
Department of Dean-Academics, K L University, AP, India

Page No.: 236-241
Volume: 19, Issue 10, 2020
ISSN: 1682-3915
Asian Journal of Information Technology

Abstract: Cloud computing significantly plays a role in the aspect of effective resource utilization and service consumption. Irrespective of the type of clouds (ex. Private, public, hybrid or inter-cloud), every service providers concentrates on the data residing 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 research paper focuses on a new direction to combine the features of data compression with the cloud data in order to secure the cloud data storage.

INTRODUCTION

Either to consider a profitable or non-profitable organizations, dedicated resource utilization brings more elevation in the economical 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.

The cloud computing creates a motivation among the enterprise service or resource sharing groups. Based on the demand, the required supply with adequate information causes a rift in the technological era. Every moment, the attempt of researchers not produces an effective impact on the fulfilment of organization requirement in effective manner.

The cloud service providers facing difficulties in order to make services among the variety of cloud clients in a secure manner. Because of this concern, the cloud resources are virtualized before it's delivered to the clients or cloud users.

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[1].

The most challenging task in the cloud data servers are focusing on the handling of residing data under the

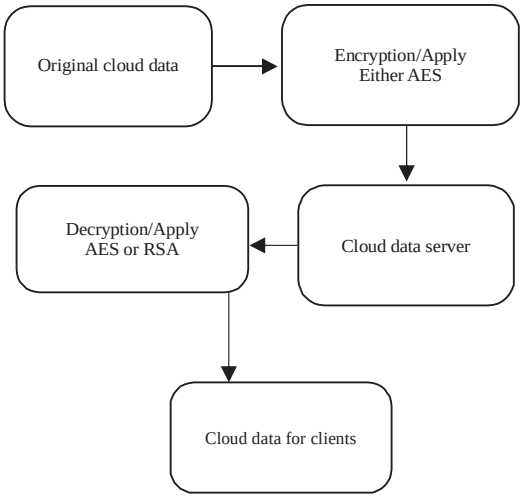

Fig. 1: Existing Cloud data secure mechanism

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[2].

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[3].

The data or information 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[4].

The following block diagram (Fig. 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.

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[5].

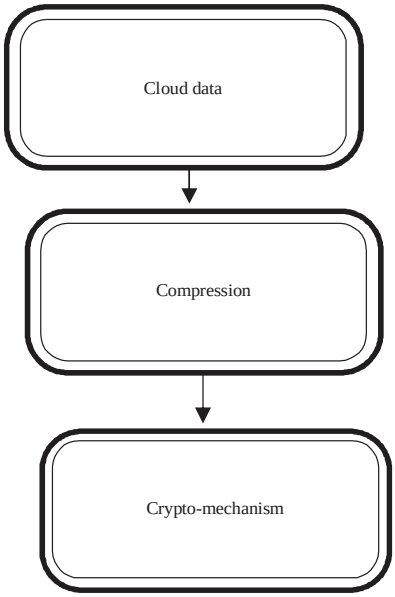

Fig. 2: Existing data security in the combination of compression and cryptography

LITERATURE REVIEW

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 (Fig. 2) is clearly indicates the existing work modules.

Cloud Computing has become an essential characteristic in today's world as technology has grown past all the limitations and there is a need to connect resources and users without having physical connection[6].

The high stipulate for data processing and leads to high computational requirement which is usually not available at the user's end Compression algorithms reduce the redundancy in data representation thus increasing effective data density.

Data compression is a very useful technique that helps in reducing the size of text data and storing the same amount of data in relatively fewer bits resulting in reducing the data storage space, resource usage or transmission capacity[6].

Huffman compressor: Huffman coding, compression and decompression is a short program that needs very little memory. But it doesn't compress particularly well when compared to commonly use other compression techniques programs. The efficiency of Huffman coding also depends heavily on having a good estimate of the true probability of the value of each input symbol.

Improvements: Here to surpass the compression ratio of these programs, we need to start adding enhancements to the modelling code. That is Arithmetic coding produces slight gains over Huffman coding. So, after constructing the Huffman code, once again apply the arithmetic coding in Huffman code, might be we can get the better result.

Arithmetic compressor: This algorithm wills results in both faster updates as well as better compression. This scheme works well for incrementally encoding a message. There is enough accuracy retained during the double precision integer calculations to ensure that the message is accurately encoded.

In order to use arithmetic coding to compress data, a model for the data is needed. The model needs to be able to do two things to effectively compress data:

- The model needs to accurately predict the frequency/probability of symbols in the input data stream
- The symbol probabilities generated by the model need to deviate from a uniform distribution

The process of encoding and decoding a stream of symbols using arithmetic coding is not too complicated. But at first glance, it seems completely impractical. Most computers support floating point numbers of up to 80 bits or so.

Run length encoding: The first step in this technique is read file then it scans the file and find the repeating string of characters[6]. When repeating characters found it will store those characters with the help of escape character followed by that character and count the binary number of items it is repeated. This method is useful for image having solid black pixels. This algorithm is also effective for repeating of characters. But it is not effective if data file has less repeating of characters. We can compress the run-length symbols using Huffman coding, arithmetic coding, or dictionary based methods[6].

Data compression mechanism is almost used in variety of multimedia applications in order to avoid the wastage of storage locations as well as to save the memory locations.

The audio or video files and Images consume more storage space and it may cause more delay in response at the time retrieval over the network. Especially the security concern is only focus on the stored content based on their copy right violation assessment and makes it a protection from unauthorized copy sources.

But the same mechanism is combined with the cloud data servers, it may create an effective communication and secure data sharing path between the cloud clients and cloud service providers. Each and every data compression algorithms such as Arithmetic coding.

Table 1: Performance evaluation survey for data compression with cryptography

Compression algorithm	Compression ratio (KB)	Cryptographic algorithm	Run-time efficiency (sec)
Arithmetic coding	13:2	RSA	0.76
Run-length coding	23:5	AES	0.98
Huffman coding	34:6	DES	1.23

Run length coding LZW and Huffman coding are rating based on the run time efficiency as well as storage statistics. Cryptographic algorithms such as RSA, Julies Ceaser Cipher, AES and DES are rating based on the run-time decoding efficiency as well as the number of iterations required to get original data or message.

The performance evaluations survey[5] based on the combination of cryptographic algorithm along with compressions are listed in Table 1. The compression ratio for the different data compression algorithms are evaluated by using the simulation software mental modeller 5.0 and the run-time efficiency is calculated based on the execution time in order to re-produce the original data at the receiving end.

One of the major disadvantages of this existing work is not suitable for any cloud service data storage. 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[7].

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 Fig. 3.

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.

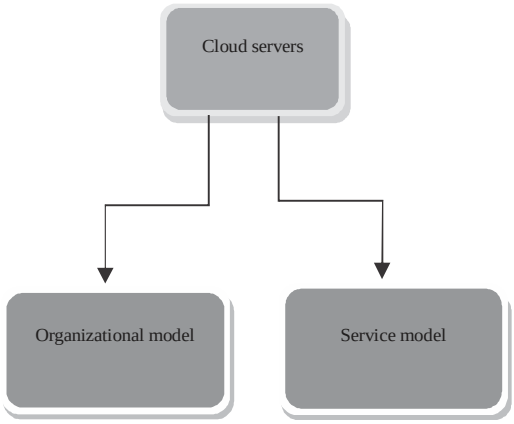

Fig. 3: Cloud service model classifications

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 content address mapping mechanism for 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 Ssoftware 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.

PROPOSED WORK

The data for the transmission is using encryption with the help of any one of the crypto algorithm is combined with data compression algorithm before it will store into cloud storage.

The encryption is a possible way to ful?ll the confidentiality requirement of data that is being stored in an untreated cloud database. The request data are used to retrieve data from a cloud database.

After the completion of the framework, we proceeded to tests in order to validate experimentally the suitable operation of the framework and to evaluate different parameters associated with the compression, decompression, encryption and decryption of multimedia contents.

As may be seen in this table, the obtained results show that compression followed by encryption may be effective for efficient and secure storage of multimedia content. On the other hand, if we apply first encryption followed by compression, the compression rate is very small due to the higher degree of randomness introduced by encryption. These results confirm the unsuitability of encryption followed by compression for efficient multimedia storage.

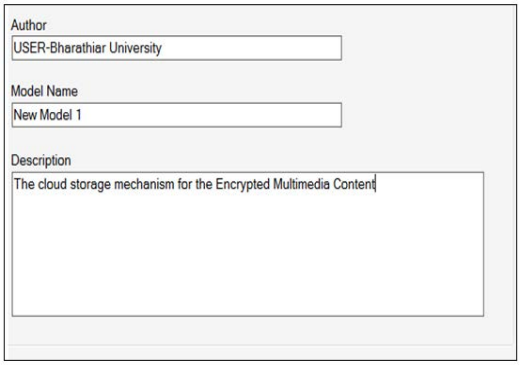

Fig. 4: Interface

The properties of an encryption function determine which kinds of method can be applied on the encrypted data without decrypting them and, consequently, how it will be stored in to the cloud service provider. The interface for the cloud data access is illustrated by Fig. 4.

The data for the transmission is used for the encryption by using any one of the crypto algorithm. A deterministic encryption scheme (as opposed to a probabilistic encryption scheme) is a crypto system which always produces the same cipher text for a given plaintext and key. The property of in distinguish ability under chosen plaintext attack is considered a basic requirement for most provably secure public key crypto systems.

The cloud encryption component is shown by Fig. 4. Among the existing crypto algorithms, most of the security algorithms focus on public key encryption mechanism.

In public key crypto algorithms provide an efficient way to make a coding statement in order to secure the original data from unauthorized third party over the network.

In most of the occurrences, the cloud data resides in sever is undergone for unauthorized way by breaking the cipher text or Encrypted information.

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.

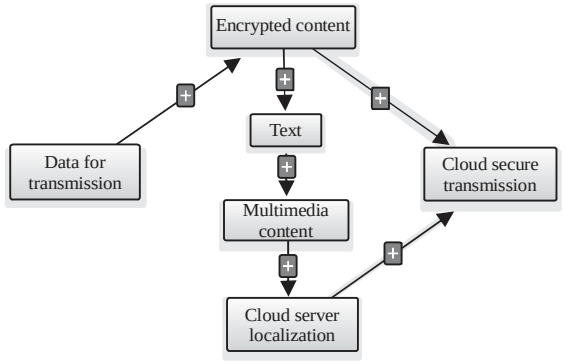

Fig. 5: Cloud encryption components

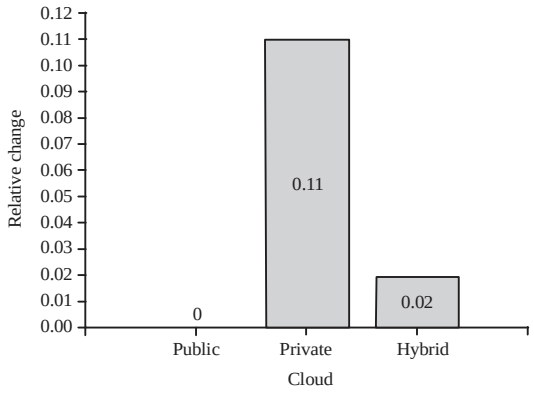

Fig. 6: Level of security assignments

Among the existing data compression algorithms, arithmetic coding is selected and RSA is selected among the cryptographic algorithms based on their compression ratio efficiency and run-time efficiency.

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 level of security assignment in the cloud category is depicted in Fig. 5 and 6. In most of the category the private cloud take a place of high priority in the security level assignment.

Every cloud service providers give more attention towards the private cloud data transaction rather than the public and hybrid clouds.

The multimedia content especially data distributed in different cloud servers located in different geographical location is usually managed by block link tables for linked access.

The data (original) is get compressed by using arithmetic code compression algorithms as per the justification and it will encrypt by using an RSA public key encryption algorithm.

The way to secure the cloud data by using compression and cryptographic algorithm is implemented in the combination of Arithmetic Coding along with RSA algorithm. The sample source code is specified as follows Algorithm 1.

Algorithm sample source code

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include “coder.h”
#include “model.h”
#include “bitio.h”
FILE *text_file
FILE *compressed_file
/*Declarations for local procedures.*/
void initialize_options (int argc, char **argv)
int check_compression (void)
void print_compression (void)
void main (int argc, char **argv)
{
    SYMBOL s
    int c
    int escaped
    int flush = 0
    long int text_count = 0
    initialize_options (--argc, ++argv)
    initialize_model()
    initialize_output_bitstream()
    initialize_arithmetic_encoder()
    for ( ; ; )
   {
          if ( ( ++text_count & 0x0ff ) = = 0 )
        flush = check_compression()
     if ( !flush )
        c = getc( text_file )
     else
       c = FLUSH
     if ( c == EOF )
        c = DONE
     do {
       escaped = convert_int_to_symbol (c,
&s );
       encode_symbol( compressed_file, &s )
     } while ( escaped )
     if ( c = = DONE )
          break
     if ( c = = FLUSH )
        {
           flush_model()
       flush = 0
        }
     update_model (c)
     add_character_to_model (c)
  }
```

The outputs obtain in the coding are present in this research work.

Huffman coding: Run the program as: Huffcomp aaa.txt huff.in. Huffman code compression program. Getting frequency counts.

- Building initial heap
- Building the code tree

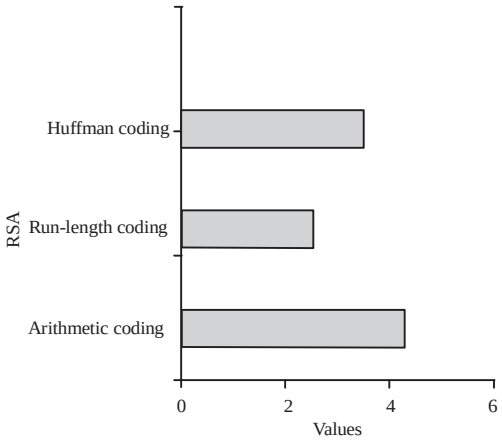

Fig. 7: Performance analyses with RSA

- Generating the code table
- Compressing and Creating the Output File
- Input characters (Bytes): 589312
- Compressed characters (Bytes): 340021
- Percentage Saving (ratio): 42.30%
- Compression time: 0.08 sec
- Huffman code decompression program
- Building the tree
- Decompressing and creating the output file
- Expansion time: 0.09 sec

Arithmetic coding:

- Run the program as: Arithcomp aaa.txt text.cmp
- Input characters (Bytes): 589312
- Compressed characters (Bytes): 327837
- Percentage Saving (ratio): 44.37%
- Compression time: 0.09 sec
- Decoding test.cmp to test.out
- Expansion time: 0.08 sec

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 (Fig. 7) clearly depicted the detailed performance evaluation.

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 cryptographic mechanism.

REFERENCES

01. Prakash, G.L., D.M. Prateek and D.I. Singh, 2014. Data security algorithms for cloud storage system using cryptographic method. Intl. J. Sci. Eng. Res., 5: 54-61.
02. Sandeep, B. Deepthi and M.S. Muneshwara, 2015. Securing and managing data for cloud storage applications using High Throughput Compression (HTC). Intl. J. Adv. Sci. Tech. Res., 3: 556-561.
03. Jayant, D.B., A.U. Swapnaja, V.P. Subhash, J.K. Kailash and S.A. Sulabha, 2015. Developing secure cloud storage system by applying AES and RSA cryptography algorithms with role based access control model. Intl. J. Comput. Appl., 118: 46-52.
04. Arora, R., A. Parashar and C.C.I. Transforming, 2013. Secure user data in cloud computing using encryption algorithms. Intl. J. Eng. Res. Appl., 3: 1922-1926.
05. Sharma, R. and S. Bollavarapu, 2015. Data security using compression and cryptography techniques. Intl. J. Comput. Appl., 117: 15-18.
06. Soni, M. and N. Shukla, 2016. Data compression techniques in cloud computing. Intl. J. Adv. Res. Comput. Commun. Eng., 5: 203-204.
07. Mrabti, A.A.E., N. Ammari, A.A.E. Kalam and A.A. Ouahman, 2016. New mechanism for cloud computing storage security. Intl. J. Adv. Comput. Sci. Appl., 7: 526-539.