www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 2 Issue 4 April, 2013 Page No. 991 -996

Multi Benfit's Through Compression For Large Data Stored In Cloud
(An integrated advantages of data compression in cloud)

[1]M.Ravi Kumar, *[2]S.Manoj Kumar

Dept. of IT (MCA)
G M R Institute of Technology Rajam, Andhra Pradesh, India
Email: ravi.mrk91@gmail.com

Dept.of IT (MCA)
G M R Institute of Technology, Rajam, Andhra Pradesh, India
Email: manojkumar.gmrit@gmail.com

Abstract—*As the sizes of IT infrastructure continue to grow, cloud computing is a natural extension of virtualization technologies that enable scalable management of virtual machines over a plethora of physically connected systems. Cloud computing provides on-demand access to computational resources which together with pay-per use business models, enable application providers seamlessly scaling their services. Cloud computing infrastructures allow creating a variable number of virtual machine instances depending on the application demands. However, even when large-scale applications are deployed over pay-per-use cloud high-performance infrastructures, cost-effective scalability is not achieved because idle processes and resources (CPU, memory) are unused but charged to application providers. Over and under provisioning of cloud resources are still unsolved issues. Here we try to present the data compression techniques to squeezing data that illustrates to reduce the size which is about deploy in cloud servers. These compression techniques are much reliable when we need to manage the large amount of data, especially useful for industries like which maintain huge data warehouses and big educational universities for reducing the cost. This work attempts to establish formal measurements for under and over provisioning of virtualized resources in cloud infrastructures, specifically for SaaS(software as a service) platform deployments and proposes a resource allocation model to deploy SaaS applications over cloud computing platforms by taking into account their multitenancy, thus creating a cost-effective scalable environments. As a result the aim of this paper is two-folded; firstly to evaluate cloud security by compressing data which contains encrypted format to ensure sufficient security, Requirements and secondly to present a viable solution that creates a cost-effective cloud environment for large-scale systems.*

Index terms—Cloud computing, Data compression, arithmetic encoding, the Lempel-Ziv family, Dynamic Markov Compression (DMC), Prediction by Partial Matching (PPM), Huffman, Run Length Encoding (RLE)

I. INTRODUCTION

Cloud computing is for internet computing. The internet is commonly visualized as clouds; hence the term "cloud computing" is for computation done through the Internet. Cloud computing provides the facility to access shared resources and common infrastructure, offering services on demand over the network to perform operations that meet changing business needs.

Several trends are opening up the era of Cloud Computing, which is an Internet-based development and use of computer technology. The ever cheaper and more powerful processors, together with the "software as a service" (SaaS) computing architecture, are transforming data centers into pools of computing service on a huge scale. Meanwhile, the increasing network bandwidth and reliable yet flexible network connections make it even possible that clients can now subscribe high-quality services from data and software that reside solely on remote data centers.

Data storage paradigm in "Cloud" brings about many challenging design issues which have profound influence on the security and performance of the overall system. One of the biggest concerns with cloud data storage is that of data integrity verification at un-trusted servers. As the cloud services have been built over the internet, any issue that is related to internet security will also affect the cloud services. But one question about cloud computing is still in its place- "HOW SECURE IS THE CLOUD??"End user who wants to access the services of cloud must have browser on their system to access the network. We always talk about attacks on clouds which make our data insecure on clouds system but there are so many attacks which can also affect our data. Resources in the cloud are accessed through the internet; consequently even if the cloud provider focuses on security

in the cloud infrastructures, the data is still transmitted to the users through the internet network which may be insecure. As a result, the impact of internet security problems will affect the cloud. Moreover, cloud risks are more dangerous due to valuable resources stored within them and cloud vulnerability. The technology used in the cloud is similar to technology used in the Internet. Encryption techniques and secure protocols are not sufficient to assist data transmission in the cloud. Data intrusion of the cloud through the Internet by hackers and cybercriminals needs to be addressed and the cloud environment needs to be secure and private for clients.

In addition, data storage or data retrieval cost is high especially for small companies. Economic computing resources and advanced network technology is referred to as cloud computing .The use of cloud computing has increased rapidly in many organizations. For this we present a new approach that facilitates that to reduce the cost than normal usage and providing much security also. To achieve this we follow the most well-known area in the IT for effective data management i.e..The process of reducing the size of a data file is popularly referred to as" data compression". Data compression squeezes data so it requires less disk space for storage and less bandwidth on a data transmission channel. By using this technique we can reduce the size of data stored in the cloud and along with this we can provide sufficient security also. This paper mainly focuses the cost-effective secured data in cloud platforms.

II. EXISTING SYSTEM

Cloud security is one of the major issues. In general Security means, focus will be giving attention on confidentiality, Integrity, Availability. But will that be sufficient? Cloud Computing is providing services Such as Infrastructure as a Service, Platform as a Service, Software as a Service, or Anything as a Service through internet based as pay per usage model like utility computing.

Cloud Security Alliance (CSA): CSA gave a list of top threats to cloud computing such as Abuse and Nefarious Use of Cloud Computing, Insecure Interfaces and APIs, Malicious Insiders, Shared Technology Issues, Data Loss or Leakage, Account or Service Hijacking, Unknown Risk Profile and they suggested tackling methods such as tools to monitor the IP , APIs, encryption, firewalls along with strong authentication .

Since outsourcing is the main theme of cloud computing, there are two main concerns in this area:

1. External attacker (any unauthorized person) can get to the Critical data, as the control is not in the hands of the owner.
2. Cloud service provider himself can breach the owner, as Data is to be kept in his premises.

i.e.. We don't have required security in cloud servers. Even when large-scale applications are deployed over pay-per-use cloud high-performance infrastructures, cost-effective scalability is not achieved because idle processes and resources (CPU, memory) are unused but charged to application providers. Over and under provisioning of cloud resources are still unsolved issues.

III. PROPOSED METHOD

Here our approach is looking simple but it sounds somewhat new. i.e., at first compress the data by using some of the popular compression techniques based on that particular scenarios and then deploy this compressed data into the cloud. As a result we can provide security because the intruder cannot understand the compressed text (which is in encrypted format) mostly. At the same time we can get the low-cost cloud services because it reduces data in a broader way. Retrieval and re-use the compressed data is also easy by following reverse process means that as a decompression (decryption) mechanism.

Two important compression concepts are loss and lossless compression: Lossless algorithms are typically used for text or executable codes, while losses are used for images and audio where a little bit of loss in resolutions often undetectable, or at least acceptable.

Lossless compression researchers have developed highly sophisticated approaches, such as Huffman encoding, arithmetic encoding, the Lempel-Ziv family, Dynamic Markov Compression (DMC), Prediction by Partial Matching (PPM), and Burrows-Wheeler Transform (BWT) based algorithms.

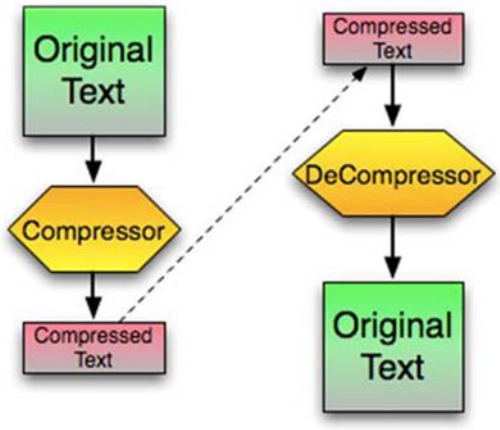

Figure 1: Compression process

Among the available lossless compression algorithms the following are considered for this study.

A. Run Length Encoding Algorithm

Run Length Encoding or simply RLE is the simplest of the data compression algorithms. The consecutive

Sequences of symbols are identified as runs and the others are identified as non runs in this algorithm. This algorithm deals with some sort of redundancy. It checks whether there are any repeating symbols or not, and is based on those redundancies and their lengths. For an example, the text "ABABBBBC" is considered as a source to compress, then the first 3 letters are considered as a non-run with length 3, and the next 4 letters are considered as a run with length 4 since there is a repetition of symbol B. The major task of this algorithm is to identify the runs of the source file, and to record the symbol and the length of each run. The Run Length Encoding algorithm uses those runs to compress the

original source file while keeping all the non-runs without using for the compression process.

B. Huffman Encoding

Huffman Encoding Algorithms use the probability distribution of the alphabet of the source to develop the code words for symbols. The frequency distribution of all the characters of the source is calculated in order to calculate the probability distribution. According to the probabilities, the code words are assigned. Shorter code words for higher probabilities and longer code words for smaller probabilities are assigned. For this task a binary tree is created using the symbols as leaves according to their probabilities and paths of those are taken as the code words. Two families of Huffman Encoding have been proposed: Static Huffman Algorithms and Adaptive Huffman Algorithms. Static Huffman Algorithms calculate the frequencies first and then generate a common tree for both the compression and decompression processes. Details of this tree should be saved or transferred with the compressed file. The Adaptive Huffman algorithms develop the tree while calculating the frequencies and there will be two trees in both the processes. In this approach, a tree is generated with the flag symbol in the beginning and is updated as the next symbol is read.

C. The Shannon Fano Algorithm

This is another variant of Static Huffman Coding algorithm. The only difference is in the creation of the code word. All the other processes are equivalent to the above mentioned Huffman Encoding Algorithm.

D. Arithmetic Encoding

In this method, a code word is not used to represent a symbol of the text. Instead it uses a fraction to represent the entire source message. The occurrence probabilities and the cumulative probabilities of a set of symbols in the source message are taken into account. The cumulative probability range is used in both compression and decompression processes. In the encoding process, the cumulative probabilities are calculated and the range is created in the beginning. While reading the source character by character, the corresponding range of the character within the cumulative probability range is selected. Then the selected range is divided into sub parts according to the probabilities of the alphabet. Then the next character is read and the corresponding sub range is selected. In this way, characters are read repeatedly until the end of the message is encountered. Finally a number should be taken from the final sub range as the output of the encoding process. This will be a fraction in that sub range. Therefore, the entire source message can be represented using a fraction. To decode the encoded message, the number of characters of the source message and the probability/frequency distribution are needed.

E. Measuring Compression Performances

Depending on the nature of the application there are various criteria to measure the performance of a

Compression algorithm When measuring the performance the main concern would be the space efficiency. The time efficiency is another factor. Since the compression behavior depends on the redundancy of symbols in the source file, it is difficult to measure performance of a compression algorithm in general. The performance depends on the type and the structure of the input source. Additionally the compression behavior depends on the category of the compression algorithm: lossy or lossless. If a lossy compression algorithm is used to compress a particular source file, the space efficiency and time efficiency would be higher than that of the lossless compression algorithm. Thus measuring a general performance is difficult and there should be different measurements to evaluate the performances of those compression families. Following are some measurements used to evaluate the performances of lossless algorithms.

Compression Ratio is the ratio between the size of the compressed file and the size of the source file.

$$\text{Compression Ratio} = \frac{\text{size after compression}}{\text{size before compression}}$$

Compression Factor is the inverse of the compression ratio. That is the ratio between the size of the source file and the size of the compressed file.

$$\text{Compression Factor} = \frac{\text{size before compression}}{\text{size after compression}}$$

Saving Percentage calculates the shrinkage of the source file as a percentage.

$$\text{Saving percentage} = \frac{\text{size before compression} - \text{size after compression}}{\text{size before compression}} \%$$

All the above methods evaluate the effectiveness of compression algorithms using file sizes. There are some other methods to evaluate the performance of compression algorithms. Compression time, computational complexity and probability distribution are also used to measure the effectiveness.

Compression Time

Time taken for the compression and decompression should be considered separately. Some applications like transferring compressed video data, the decompression time is more important, while some other applications both compression and decompression time are equally important. If the compression and decompression times of an algorithm are less or in an acceptable level it implies that the algorithm is acceptable with respective to the time factor. With the development of high speed computer accessories this factor may give very small values and those may depend on the performance of computers.

Entropy

This method can be used, if the compression algorithm is based on statistical information of the source file. Self Information is the amount of one's surprise evoked by an event. In another words, there can be two events: first one is an event which frequently happens and the other one is an event which rarely happens. If a message says that the second event happens, then it will generate more surprise in receivers mind than the first message. Let set of event be S = $\{s_1,s_2,s_3,...,)$ for an alphabet and each S_j is a symbol used in this alphabet. Let the occurrence probability of each event

be PJ for event S_j. Then the self information I(s) is defined as follows.

$I(S_j) = \log \frac{1}{p_j}$ (or) $I(S_j) = -\log \frac{1}{p_j}$

The first order Entropy value H (P) of a compression algorithm can be computed as follows.

$H(P) = \sum_{j=1}^{n} P_j\ I(S_j)$ (or) $H(P) = -\sum_{j=1}^{n} P_j\ I(S_j)$

Code Efficiency

Average code length is the average number of bits required to represent a single code word. If the source and the lengths of the code words are known, the average code length can be calculated using the following equation.

$$l = \sum_{j=1}^{n} P_j\ L_j$$

Where pi is the occurrence probability of J^{th} symbol of the source message ,Li is the particular length of the code word for that symbol and L = {l1, l2, …. Ln}. Code Efficiency is the ratio in percentage between the entropy of the source and the average code length and it is defined as follows.

$E(P, L) = \frac{H(P)}{l(P,L)}$ X 100 %

Where E(P,L) is the code efficiency, H(p) is the entropy and l(P,L) is the average code length.

The above equation is used to calculate the code efficiency as a percentage. It can also be computed as a ratio. The code is said to be optimum, if the code efficiency values is equal to 100% (or 1.0). If the value of the code efficiency is less than 100%, that implies the code words can be optimized than the current situation.

IV. Results and Recommendations

The overall performance in terms of average BPC(bits per character) of the above referred Statistical coding methods are shown in Fig. Table shows the comparative analysis between various Statistical compressions techniques discussed above.

As per the results shown in Table for Run Length Encoding, for most of the files tested, this algorithm generates compressed files larger than the original files. This is due to the fewer amount of runs in the source file. For the other files, the compression rate is less. The average BPC obtained by this algorithm is 7.93. So, it is inferred that this algorithm can reduce on an average of about 4% of the original file. This cannot be considered as a significant improvement.

BPC and amount of compression achieved for Shannon-Fano algorithm is presented in Table, The compression ratio for Shannon-Fano algorithm is in the range of 0.60 to 0.82 and the average BPC is 5.50.

Compression ratio for Huffman coding algorithm falls in the range of 0.57 to 0.81. The compression ratio obtained by this algorithm is better compared to Shannon-Fano algorithm and the average Bits per character is 5.27.

The amount of compression achieved by applying Adaptive Huffman coding is shown in Table . The adaptive version of Huffman coding builds a statistical model of the text being compressed as the file is read. From Table it can be seen that, it differs a little from the Shannon-Fano coding algorithm and Static Huffman coding algorithm in the compression ratio achieved and the range is between 0.57 and 0.79. On an average the number of bits needed to code a character is 5.21. Previous attempts in this line of research make it clear that compression and decompression times are relatively high for this algorithm because the dynamic tree used in this algorithm has to be modified for each and every character in the source file.

Arithmetic coding has been shown to compress files down to the theoretical limits as described by Information theory. Indeed, this algorithm proved to be one of the best performers among these methods based on compression ratio. It is clear that the amount of compression achieved by Arithmetic coding lies within the range of 0.57 to 0.76 and the average bits per character is 5.15. The overall performance in terms of average BPC of the above referred Statistical coding methods are shown in below Table

S.No	File names	File Size	RLE	Shannon Fano coding	Huffman coding	Adaptive Huffman coding	Arithmetic coding
			BPC	BPC	BPC	BPC	BPC
1	bib	111261	8.16	5.56	5.26	5.24	5.23
2	book1	768771	8.17	4.83	4.57	4.56	4.55
3	book2	610856	8.16	5.08	4.83	4.83	4.78
4	news	377109	7.98	5.41	5.24	5.23	5.19
5	obj1	21504	7.21	6.57	6.45	6.11	5.97
6	obj2	246814	8.05	6.5	6.33	6.31	6.07
7	paper1	53161	8.12	5.34	5.09	5.04	4.98
8	paper2	82199	8.14	4.94	4.68	4.65	4.63
9	progc	39611	8.1	5.47	5.33	5.26	5.23
10	progl	71646	7.73	5.11	4.85	4.81	4.76
11	progp	49379	7.47	5.28	4.97	4.92	4.89
12	trans	93695	7.9	5.88	5.61	5.58	5.49
	Average BPC		**7.93**	**5.5**	**5.27**	**5.21**	**5.15**

Figure 2: Comparison of BPC for different Statistical Compression techniques

Below graph presents the overall results of the given statistics in the above table:

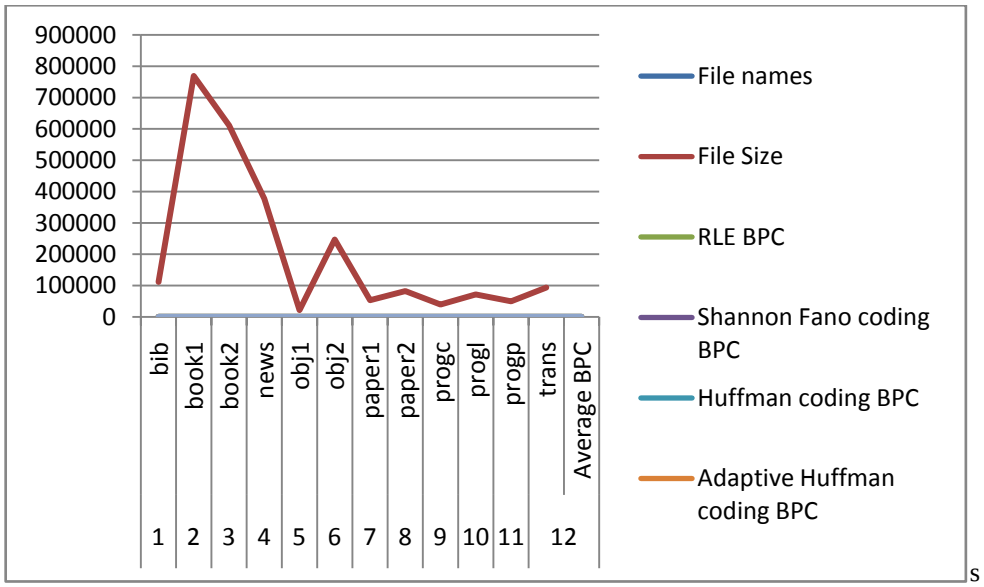

Figure 3: Chart showing Compression rates for various Statistical Compression techniques

After performing the entire compression process based upon the requirements whether it may be the text, image, audio, or videos anything But here in this paper we mostly concentrate on the textual data only. We can compress then deploy that into the cloud as under its specific services.

For retrieve and reuse also we perform the same task in an opposite way. The below diagram illustrates the clear understanding of our approach.

Of course it needs some additional resources like technical staff and consumes some extra time also. But if we did like this we can get some secured storage and also get lower cost cloud services.

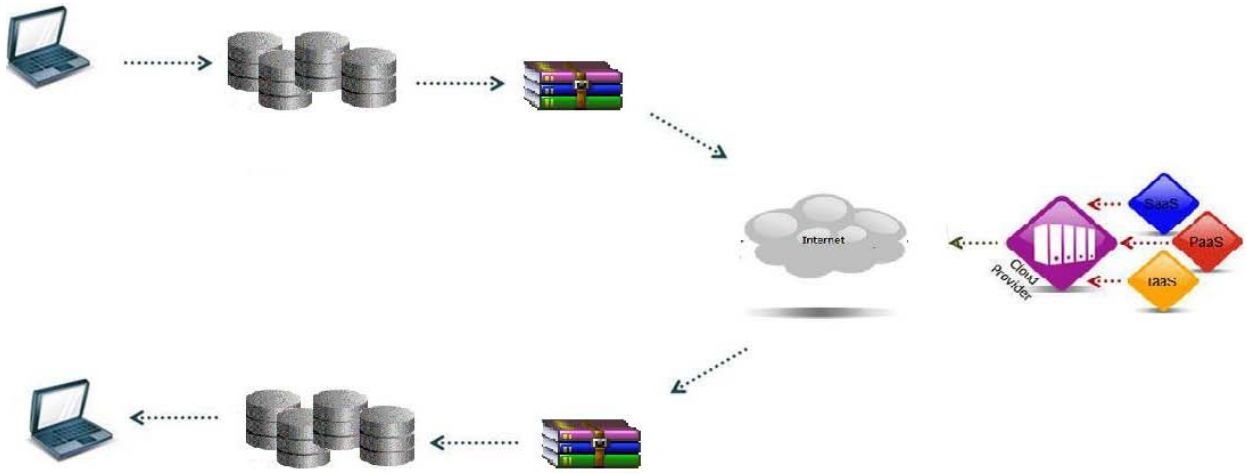

Figure 4: Storing and Retrieval process of compressed text in Cloud

V. CONCLUSION

Modern computing presents not only technical problems, but also major environmental challenges in terms of its high energy consumption. As the sizes of IT infrastructure continue to grow, it is clear that effective green IT solutions must be developed to minimize its impacts on our environment. Here, cloud computing offers a natural extension of virtualization technologies which enable scalable management of virtual machines residing on distributed hosts. By applying the required compression techniques to reduce the size of the data as well as we can produce the encrypted data to resolve the security issues, then the result will be the solution that produces a cost-effective cloud environment for both small and large-scale industries also.

Even though Cloud has some serious issues like security, privacy, social and political issues. Cloud computing is going to be one of the venture technology in future. Cloud user should understand their own network, system, applications and data are moving to an unknown network which poses serious threat to security and privacy. As a cloud user or developer, they have to choose the vendor based on their Service Level Agreements, security service standards and compliances.

REFERENCES

[1] NilayOza, KaarinaKarppinen, ReijoSavola, "User experience and Security in the Cloud– An Empirical Study in the Finnish Cloud Consortium",2nd IEEE International Conference on Cloud Computing Technology and Science, 2010

[2] Loganayagi B, Sujatha.S., "Enhanced Cloud Security by Combining Virtualization and Policy Monitoring Techniques" Procedia Engineering 30 (2012) 654 – 661

[3] http://en.wikipedia.org

[4] S.R. Kodituwakku ,"COMPARISON OF LOSSLESS DATA ALGORITHMS FOR TEXTDATA", Indian Journal of Computer Science and EngineeringVol 1 No 4 416-425

[5] "DataCompressionTheCompleteReference 3/e",David

[6] SalomonSalauddin Mahmud.," An Improved Data Compression Method for General Data "International Journal of Scientific & Engineering Research Volume 3, Issue 3, March -2012

[7] http://www.journalofcloudcomputing.com

M.RaviKumar (ravi.mrk91@gmail.com), is pursuing MCA (Master of Computer Applications) from GMR Institute of Technology, Rajam, A.P, India. Presently in second year and has completed his graduation (Computer Science) in Andhra University. His area of interest includes Network & Information Security, Web designing & developing and development technologies.

S.Manoj Kumar

(manojkumar.gmrit@gmail.com), is pursuing MCA (Master of Computer Applications) from GMR Institute of Technology, Rajam, A.P, India. Presently in second year and has completed his graduation (Computer Science) in Andhra University. His area of interest includes Network & Information Security, Cloud Computing and latest technologies.