International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 9, September - 2013

Efficient resource allocation and data processing in a secured cloud environment

Prarthana T.V

Abstract

Current data processing frameworks rather expect the cloud to imitate the static nature of the cluster environment. As a result, rented resources may be inadequate for big parts of the processing job, which may lower the overall processing performance and increase the cost.
The main objective of this project is to implement a new processing framework explicitly designed for cloud environments. This framework will include the possibility of dynamically allocating/de-allocating different compute resources from a cloud in its scheduling and during job execution.
The framework is designed for Cloud Computing and can adapt to the dynamic nature of a Cloud. 'Dynamic' means that the number and kind of the machines in the Cloud is neither fixed nor uniform. The system adapts to this changes automatically. The user just submits the task(s) to the system not knowing the hardware or the configuration of it. With this the entire data processing becomes very simple, fast and their by the IaaS service is economical.

1. Introduction

Cloud computing

Cloud computing is a subscription-based service where you can obtain networked storage space and computer resources. One way to think of cloud computing is to consider an experience with the email. Email clients, such as Yahoo!, Gmail, Hotmail, and so on, takes care of housing all of the hardware and software necessary to support personal email accounts. Steps to access the emails are, opening web browsers, email clients, and logging in. The most important part of the equation is having internet access. Emails are not housed on the physical computer; it can be accessed through internet connection and can be accessed from anywhere. Emails are different than software installed on the computer, such as a word processing program. Documents created using word processing software stays on the device, unless physically moved. An email client is similar to how cloud computing works. Except instead of accessing the email, one can choose what information to access within the cloud.

The cloud makes it possible to access information from anywhere at any time. While a traditional computer setup requires being in the same location as the data storage device, the cloud takes away that step. The cloud removes the need to be in the same physical location as the hardware that stores the data. Cloud provider can both own and house the hardware and software necessary to run home or business applications. This is especially helpful for businesses that cannot afford the same amount of hardware and storage space as a bigger company. Small companies can store their information in the cloud, removing the cost of purchasing and storing memory devices. Additionally, the amount of storage space to be purchased is determined based on once usage and business requirement. Based on the business growth, the storage space subscription can be increased or decreased.

One mandatory requirement to access the cloud is to have an internet connection. To access a specific document housed in the cloud, an internet connection either wireless or wired, or a mobile broadband connection need to be established. The benefit is, the same document can be accessed from any place, any device that can access the internet. These devices could be a desktop, laptop, tablet, or phone. This can also help the business to function more smoothly because anyone who can connect to the internet and thereby to the cloud can work on documents, access software, and store data. This is the freedom that the cloud can provide for an organization

There are different types of clouds depending on the needs.

1. Public Cloud - A public cloud can be accessed by any subscriber with an internet connection and access to the cloud space.
2. Private Cloud - A private cloud is established for a specific group or organization and limits access to just that group.
3. Community Cloud - A community cloud is shared among two or more organizations that have similar cloud requirements.

4. Hybrid Cloud - A hybrid cloud is essentially a combination of at least two clouds, where the clouds included are a mixture of public, private, or community [5]

Resource allocation

Current data processing frameworks rather expect the cloud to imitate the static nature of the cluster environments they were originally designed, for example, at the moment the types and number of VMs allocated at the beginning of a compute job cannot be changed in the course of processing, although the tasks the job consists of might have completely different demands on the environment. As a result, rented resources may be inadequate for big parts of the processing job, which may lower the overall processing performance and increase the cost. The optimal solution would therefore be dynamic resource allocating framework. [1]

In this paper we want to discuss the way resource allocation can be handled dynamically and data processing among the cloud resources can be done in a parallel manner.

Information Security

Since the application and data residing in the cloud are becoming critical day by day, it is important that clouds should be secure. The major security challenge with clouds is that the owner of the data may not have control of where the data is placed. This is because if one wants to exploit the benefits of using cloud computing, one must also utilize the resource allocation and scheduling provided by clouds. Therefore, we need to safeguard the data in the midst of un-trusted processes. [4]

To resolve the security issue, in this paper, we utilize the encryption/decryption technique. The algorithm used for this approach is the DNA algorithm.

2. Related work

Today's processing frameworks typically assume the resources they manage consist of a static set of homogeneous compute nodes. Although designed to deal with individual nodes failures, they consider the number of available machines to be constant, especially when scheduling the processing job's execution. While IaaS clouds can certainly be used to create such cluster-like setups, much of their flexibility remains unused. One of an IaaS cloud's key features is the provisioning of compute resources on demand. New VMs can be allocated at any time through a well-defined interface and become available in a matter of seconds. Machines which are no longer used can be terminated instantly and the cloud customer will be charged for them no more. Moreover, cloud operators like Amazon let their customers rent VMs of different types, i.e. with different computational power, different sizes of main memory, and storage. Hence, the compute resources available in a cloud are highly dynamic and possibly heterogeneous.

With respect to parallel data processing, this flexibility leads to a variety of new possibilities, particularly for scheduling data processing jobs. The question a scheduler has to answer is no longer "Given a set of compute resources, how to distribute the particular tasks of a job among them?", but rather "Given a job, what compute resources match the tasks the job consists of best?". This new paradigm allows allocating compute resources dynamically and just for the time they are required in the processing workflow. E.g., a framework exploiting the possibilities of a cloud could start with a single VM which analyzes an incoming job and then advises the cloud to directly start the required VMs according to the job's processing phases. After each phase, the machines could be released and no longer contribute to the overall cost for the processing job. Facilitating such use cases imposes some requirements on the design of a processing framework and the way its jobs are described. First, the scheduler of such a framework must become aware of the cloud environment a job should be executed in. It must know about the different types of available VMs as well as their cost and be able to allocate or destroy them on behalf of the cloud customer. Second, the paradigm used to describe jobs must be powerful enough to express dependencies between the different tasks the jobs consist of. The system must be aware of which task's output is required as another task's input. Otherwise the scheduler of the processing framework cannot decide at what point in time a particular VM is no longer needed and de-allocate it. The MapReduce pattern is a good example of an unsuitable paradigm here: Although at the end of a job only few reducer tasks may still be running, it is not possible to shut down the idle VMs, since it is unclear if they contain intermediate results which are still required. Finally, the scheduler of such a processing framework must be able to determine which task of a job should be executed on which type of VM and, possibly, how many of those. This information could be either provided externally, e.g. as an annotation to the job description, or deduced internally, e.g. from collected

statistics, similarly to the way database systems try to optimize their execution schedule over time. [6]

The cloud's virtualized nature helps to enable promising new use cases for efficient parallel data processing. However, it also imposes new challenges compared to classic cluster setups. The major challenge we see is the cloud's opaqueness with prospect to exploiting data locality: In a cluster the compute nodes are typically interconnected through a physical high-performance network. The topology of the network, i.e. the way the compute nodes are physically wired to each other, is usually well known and, what is more important, does not change over time. Current data processing frameworks offer to leverage this knowledge about the network hierarchy and attempt to schedule tasks on compute nodes so that data sent from one node to the other has to traverse as few network switches as possible. [7] That way network bottlenecks can be avoided and the overall throughput of the cluster can be improved. In a cloud this topology information is typically not exposed to the customer. [3] Since the nodes involved in processing a data intensive job often have to transfer tremendous amounts of data through the network, this drawback is particularly severe; parts of the network may become congested while others are essentially unutilized. Although there has been research on inferring likely network topologies solely from end-to-end measurements, it is unclear if these techniques are applicable to IaaS clouds. For security reasons clouds often incorporate network virtualization technique which can hamper the inference process, in particular when based on latency measurements. Even if it was possible to determine the underlying network hierarchy in a cloud and use it for topology aware scheduling, the obtained information would not necessarily remain valid for the entire processing time. VMs may be migrated for administrative purposes between different locations inside the data center without any notification, rendering any previous knowledge of the relevant network infrastructure obsolete.

As a result, the only way to ensure locality between tasks of a processing job is currently to execute these tasks on the same VM in the cloud. This may involve allocating fewer, but more powerful VMs with multiple CPU cores. E.g., consider an aggregation task receiving data from seven generator tasks. Data locality can be ensured by scheduling these tasks to run on a VM with eight cores instead of eight distinct single-core machines. However, currently no data processing framework includes such strategies in its scheduling algorithms. [1]

In order to protect data through the unsecure networks like the Internet, using various types of data protection is necessary. With advent of Cloud Computing idea, a common problem, confidentiality of data, was emerged. [8, 9, 10, 11 and 12] Thus, to solve the raised difficulty, combining different ideas can help to achieve an acceptable level of confidentiality in Cloud Computing environments.

One of the famous ways to protect data through the Internet is data hiding. Because of the increasing number of Internet users, utilizing data hiding or Steganographic techniques is inevitable. Eliminating the role of the intruder and authorizing the clients are eventual goals of these techniques. Therefore, the role of data hiding has become more eminent nowadays. Before employing biological properties of DNA sequences, the common way of embedding a secret data into the host images was the traditional way of data hiding. [13, 14, 15, 16 and 17]

It unfortunately leads to some liabilities. The most important ones was the detection of the distortions of the image when the host image changed to some degrees. That was the best spot to start the wholly detection of the secret data through the image. By advent of biological aspects of DNA sequences to the computing areas, new data hiding methods have been proposed by researchers, based on DNA sequences.[18, 19, 20, 21 and 22] The key portion of their work is, utilizing biological characteristics of DNA sequences.

3. Proposed design

The framework takes up many ideas of previous processing frameworks but refines them to better match the dynamic and opaque nature of a cloud.

3.1 Architecture

The framework's architecture follows a client-server pattern as illustrated in the below diagram-

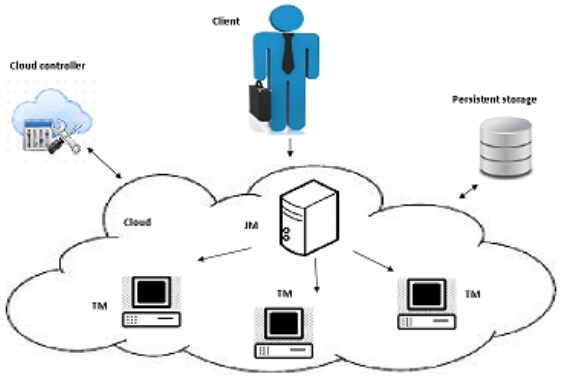

Fig. 1. Structural overview of the framework running in an Infrastructure-as-a-Service (IaaS) cloud

Before submitting the compute job, a user must start a virtual machine (VM) in the cloud which runs the so called Job Manager (JM). The Job Manager receives the user's jobs, is responsible for scheduling them, and coordinates their execution. It communicates with the interface the cloud operator provides to control the number of VMs. We call this interface the 'Cloud Controller'. The Job Manager, with the help of Cloud controller, can allocate or de-allocate VMs according to the current job execution phase.

Such jobs are divided into independent tasks by the job manager. The actual execution of these tasks is carried out by a set of instances (VM's). Each instance runs a Task Manager (TM). A Task Manager receives one or more tasks from the Job Manager at a time, executes them, and after that informs the Job Manager about their completion or possible errors. Unless a job is submitted to the Job Manager, we expect the set of instances (and there by the Task Managers) to be empty. Upon receiving the job, the Job Manager then decides, depending on the job's particular tasks, how many and what type of instances the job should be executed on, and when the respective instances must be allocated/de-allocated to ensure a continuous but cost-efficient processing.

The newly allocated instances boot up with a previously compiled VM image. The image is configured to automatically start a Task Manager and register it with the Job Manager. Once all the 3necessary Task Managers have successfully contacted the Job Manager, it triggers the execution of the scheduled job.

Initially, the VM's used to boot up the blank Task Managers which do not contain any of the data the job is supposed to operate on. As a result, we expect the cloud to offer persistent storage. This persistent storage is supposed to store the job's input data and eventually receive its output data. It must be accessible for both the Job Manager as well as for the set of Task Managers, even if they are connected by a private or virtual network. [1]

In this paper we exemplify the above concept using File upload or download process. Multiple users log into the system and can upload or download files of their choice. Cloud is a platform that can be utilized by many users across locations and platforms. So, security is always an issue. To address the issue, the file being upload by the user has to be in turn encrypted and stored in the cloud storage. While the user downloads the file, it has to be decrypted and presented to the user. DNA algorithm, which is an apt for cloud computing environments, has been implemented for encryption and decryption.

DNA algorithm implements data hiding in DNA sequences to increase the confidentiality and complexity in cloud computing environments. The algorithm is based on binary coding and complementary pair rules. Therefore, DNA reference sequence is chosen and a secret data M is hidden into it. As result of applying the algorithm, M´´´ is come out to upload to cloud environments.

The algorithm is divided into two phases. The first one is, embedding data and the second one is, extracting the original data.

A. Phase1: Embedding Secret Data

Embedding phase is separated into some successive and sub-phases. The sub-phases have been shown below, respectively.

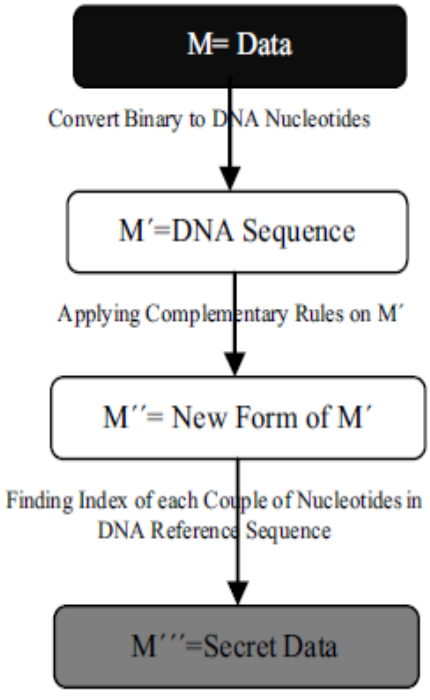

Fig.2. Phase 1: embedding secret data

'M' is the original data, which the client decides to upload via a network to cloud computing environments. There are three sub-phases to provide the final form, M´´´ (the encrypted data) and upload it to cloud.

Consider the following DNA Sequence in order to have a better understanding of the algorithm.
DNA Reference Sequence:
AT1CG2AA3TT4CG5CG6CT7GA8GT9CA10CA11A
T12TC13
GC14GC15TG16AG17TG18AA19CC20

Let the message be, M=100111000011

- Sub-phase 1: Converting by DNA base pairing rules.
  The product is M´. M´ contains nucleotides sequences (A,T,G,C). By applying DNA base pairing rules, the data can convert from binary to DNA sequence. Not only DNA base pairing helps to encrypt the data from binary to DNA sequence but also it is applied to decrypt the secret data to original one, truly.

Eg: Considering the base pair rules "A= 00, T= 01, C= 10, G= 11", Sub-phase1 - M´= CTGAAG

- Sub-phase 2: Applying complementary rules.

Increasing the complexity is the real and exact purpose of this step. By applying the complementary rules, the new form of the M´ which is M´´ emerges. As mentioned before, both of clients have a DNA reference sequence from a large number of possibilities based on the database. It means that, they have selected the same DNA reference sequence, exactly.

Eg: Apply complementary rules, "((AC) (CG) (GT) (TA))" - M´´= GATCCT

- Sub-phase 3: Extracting the index of each couple nucleotides in DNA reference sequence.
  At the end of this phase, the resultant, M´´´ is a series of numbers. Each of the number specifies the position of the couple of nucleotides in M´´ with respect to the DNA reference sequence. M´´´ is precisely the secret data with some changes through the embedding phase. Now, the encrypted data (M´´´) can be uploaded to cloud.
  Eg : Considering the indexes - M´´´=8137

B. Phase2: Extracting Original data

This phase involves a few sub-phases which takes M´´´ as input, process it and results in the original message M. There are three sub-phases discussed below.

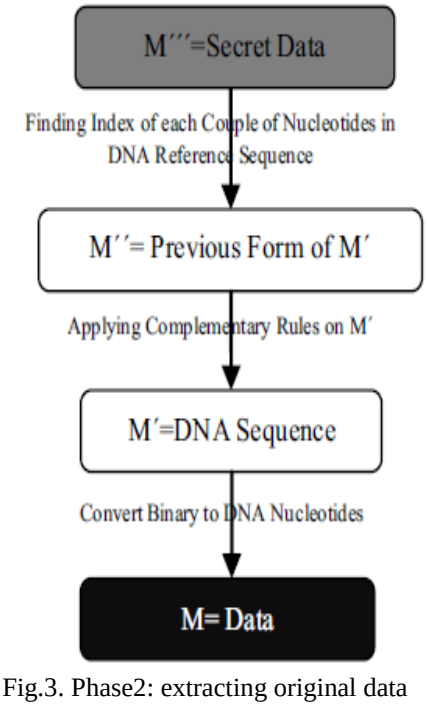

Fig.3. Phase2: extracting original data

This is the reverse process of encryption. The first sub-phase manipulates the M´´´. Extracting the data starts

by finding the indexes on DNA reference sequence one by one according to the numbers in the current secret data. M´´ is the exact product of the first sub-phase. The second sub-phase applies complementary rules on M´´ in order to extracting M´, correctly. The importance of the M´ is the form of it. M´ is the last form of data, based on DNA nucleotides. Converting the M´ to the M is the third sub-phase. Transforming from DNA nucleotides to the binary is the responsibility of the last sub-phase. This phase truly extracts the original data M. [2]

4. Implementation

The framework discussed above is currently under implementation. We follow the modular approach for the design and implementation of the framework.
The framework has been divided into three modules –

- Admin – The management part of the framework is taken care by the admin module. The Admin manages all the existing users. The DNA encryption/decryption algorithm requires a unique key for all the users. Such key distribution is also handled by the admin. Number of servers' available, total number of servers, tasks being performed at each server is kept track of in this module. Transaction management is also one of the functions.

- User – Users or clients can upload or download files to the cloud environment. Each user will be assigned credentials to maintain the confidentiality of their data. Users will not be aware of the complexity of the encryption and decryption process taking place, they can be rest assured about the data, its security and minimal response time.

- Cloud – Cloud is a virtual module here, which deals with the resources. It includes the servers that receive the request from the users, process them and store them correspondingly. Different servers used here are - the process servers, which take care of encryption and decryption processes, and the storage servers that are meant for upload and download of the files to the cloud environment. Since tasks performed by each server are kept track of, the servers are dynamically allocated according to the load.

The practical implementation of this concept involves three applications. First one being the main application with the user interface. Two other applications running on different servers in the background –

- Process: This process handles the encryption and decryption.
- Storage: Once the files are processed and ready, storing them into an appropriate place is the job of this app.

5. Future work

Currently the server details like number of servers, their IP addresses and so on are read from the configuration file. Thus addition of new servers and removal of existing servers cannot be handled dynamically by the present framework. This will addressed by the further enhancement.

6. Conclusion

In this paper we have discussed the challenges and opportunities for efficient parallel data processing in cloud environments and presented a new data processing framework, that exploits the dynamic resource allocation offered by IaaS clouds. We have described the basic architecture and design of the framework. The benefits of utilizing the framework in terms of security and reducing the usage cost are also discussed. With a framework like this at hand, there are a variety of open research issues, which we plan to address for future work. In particular, we are interested in improving the ability of the framework to adapt to resource overload or underutilization during the job execution automatically. In general, we think our work represents an important contribution to the growing field of Cloud computing services and points out exciting new opportunities in the field of parallel data processing.

7. References

[1] Daniel Warneke,Odej Kao, Exploiting Dynamic Resource Allocation for Efficient Parallel Data Processing in the Cloud, IEEE Transactions On Parallel And Distributed Systems, January 2011
[2] Mohammad Reza Abbasy, Bharanidharan Shanmugam, Enabling Data Hiding for Resource Sharing in Cloud Computing Environments Based on DNA Sequences, 2011 IEEE World Congress on Services
[3] T. White. Hadoop: The Definitive Guide. O'Reilly Media, 2009.
[4] Kevin Hamlen, Murat Kantarcioglu, Latifur Khan, Bhavani Thuraisingham, Security Issues for Cloud Computing, International Journal of Information Security and Privacy, 4(2), 39-51, April-June 2010
[5] Alexa Huth, James Cebula, The Basics of Cloud Computing, Carnegie Mellon University 2011

[6] M. Stillger, G. M. Lohman, V. Markl, and M. Kandil. LEO -DB2's LEarning Optimizer. In LDB '01: Proceedings of the 27th International Conference on Very Large Data Bases, pages 19–28, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[7] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. In OSDI'04: Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation, pages 10–10, Berkeley, CA, USA, 2004. USENIX Association.

[8] Minqi Zhou; Rong Zhang; Wei Xie; Weining Qian; Aoying Zhou; ,"Security and Privacy in Cloud Computing: A Survey," Semantics Knowledge and Grid (SKG), 2010 Sixth International Conference on , vol., no., pp.105-112, 1-3 Nov. 2010.

[9] Jian Wang; Yan Zhao; Shuo Jiang; Jiajin Le; , "Providing privacy preserving in cloud computing," Test and Measurement, 2009. ICTM '09. International conference on , vol.2, no., pp.213-216, 5-6 Dec. 2009

[10] Itani, W.; Kayssi, A.; Chehab, A.; , "Privacy as a Service: Privacy-Aware Data Storage and Processing in Cloud Computing Architectures," Dependable, Autonomic and Secure Computing, 2009. DASC '09. Eighth IEEE International Conference on , vol., no., pp.711-716, 12-14 Dec. 2009.

[11] Doelitzscher, F.; Reich, C.; Sulistio, A.; , "Designing Cloud Services Adhering to Government Privacy Laws," Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on , vol., no., pp.930-935, June 29 2010-July 1 2010.

[12] Pearson, S.;, "Taking account of privacy when designing cloud computing services," Software Engineering Challenges of Cloud Computing, 2009. CLOUD '09. ICSE Workshop on, vol., no., pp.44-52, 23-23 May 2009.

[13] C.C. Chang, C.C. Lin, C.S. Tseng, W.L. Tai, Reversible hiding in DCTbased compressed images, Information Sciences 177 (2007).

[14] C.C. Chang, W.C. Wu, Y.H. Chen, Joint coding and embedding techniques for multimedia images, Information Sciences 178 (2008).

[15] C.H. Huang, J.L. Wu, Fidelity-guaranteed robustness enhancement of blind-detection watermarking schemes, Information Sciences 179 (2009).

[16] H.H. Tsai, D.W. Sun, Color image watermark extraction based on support vector machines, Information Sciences 177 (2007).

[17] H.W. Tseng, C.P. Hsieh, Prediction-based reversible data hiding, Information Sciences 179 (2009).

[18] C.C. Chang, T.C. Lu, Y.F. Chang, R.C.T. Lee, Reversible data hiding schemes for deoxyribonucleic acid (DNA) medium, International Journal of Innovative Computing, Information and Control 3 (2007).

[19] C.T. Clelland, V. Risca, C. Bancroft, Hiding messages in DNA microdots, Nature 399 (1999).

[20] A. Leier, C. Richter, W. Banzhaf, H. Rauhe, Cryptography with DNA binary strands, BioSystems 57 (2000).

[21] I. Peterson, Hiding in DNA, Muse (2001)

[22] B. Shimanovsky, J. Feng, M. Potkonjak, Hiding data in DNA, in: Revised Papers from the 5th International Workshop on Information Hiding, Lecture Notes in Computer Science 2578 (2002).