IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 4, April 2015.
www.ijiset.com
ISSN 2348 – 7968

Proposition for Complete Homomorphic Encryption using Grids in Cloud Computing

Arjunsingh Yadav, Priya Tonde, Priyanka Yewale, Prof. Mrs. Smita Chavan

[1] Computer Engineering, Savitribai Phule Pune University, Pune, Maharashtra, India

[2] Computer Engineering, Savitribai Phule Pune University, Pune, Maharashtra, India

[3] Computer Engineering, Savitribai Phule Pune University, Pune, Maharashtra, India

Abstract

Homomorphic Encryption enables us to perform operations on encrypted data without decrypting it. This method is well suited for enhancing the performance and security where the cloud service providers wouldn't have access to the user's data. A Fully homomorphic encryption is still challenged and is proposed with improved methods for its improved efficiency maintaining its security consistent.

We propose a new method to improve the homomorphic encryption performance along with security by using grid computation in clouds thereby reducing the workload on each processor and thus reducing the overall processing of the encrypted data and sending the encrypted result back to the user

.***Keywords:*** *Encryption, Decryption, Homomorphic Encryption, Grid Computing, Virtualization.*

1. Introduction

The development of cloud storage and computing platforms allows users to outsource storage and computations on their data, and allows businesses to offload the task of maintaining data-centers. However, concerns over loss of privacy and business value of private data is an overwhelming barrier to the adoption of cloud services by consumers and businesses alike. An excellent way to assuage these privacy concerns is to store all data in the cloud encrypted, and perform computations on encrypted data. To this end, we need an encryption scheme that allows meaningful computation on encrypted data, namely a homomorphic encryption scheme.

Although homomorphic encryption has been developed a long ago, its implementation suffers due to the processing involved in processing the operations over the huge encrypted data present in the cloud system which is the reason why full homomorphic encryption isn't possible. We can overcome this barrier if we reduce the workload supplied to a particular processor by distributing it to others and thereby executing the operations in grid and then sending the encrypted result to the user thus reducing the time to decrypt the data to perform operations and then encrypt them again to send it to the user

2. Literature Survey

2.1 A Fully Homomorphic Encryption Scheme

They propose the first fully homomorphic encryption scheme, solving a central open problem in cryptography. Such a scheme allows one to compute arbitrary functions over encrypted data without the decryption key one can efficiently compute a compact cipher text that encrypts f (mt) for any efficiently computable function f. The fully homomorphic encryption has numerous applications. For example, it enables private queries to a search engine the user submits an encrypted query and the search engine computes a succinct encrypted answer without ever looking at the query in the clear. It also enables searching on encrypted data-a user stores encrypted files on a remote file server and can later have the server retrieve only files that (when decrypted) satisfy some Boolean constraint, even though the server cannot decrypt the files on its own. More broadly, fully homomorphic encryption improves the

ISSN 2348 – 7968

efficiency of secure multiparty computation. Their construction begins with a somewhat homomorphic "boostrappable" encryption scheme that works when the function f is the scheme's own decryption function. They then show how, through recursive self-embedding, bootstrappable encryption gives fully homomorphic encryption. The construction makes use of hard problems on ideal lattices. [1].

2.2 High Performance Parallel Computing with Cloud and Cloud Technologies

They present their experiences in applying, developing, and evaluating cloud and cloud technologies. First, they present Hadoop and DryadLINQ to a series of data/compute intensive applications and then compare them with a novel MapReduce runtime developed by them, named CGL-MapReduce, and MPI. They identify the basic execution units of the MapReduce programming model and categorize the runtimes according to their characteristics. MPI versions of the applications are used where the contrast in performance needs to be highlighted. They discuss the application structure and their mapping to parallel architectures of different types, and look at the performance of these applications. Next, they present a performance analysis of MPI parallel applications on virtualized resources. [2]

3. Existing System Design

3.1 Overview and Working

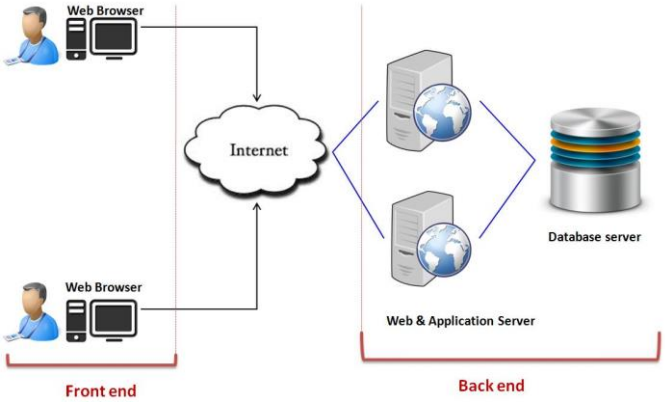

Fig 1: Existing Cloud System

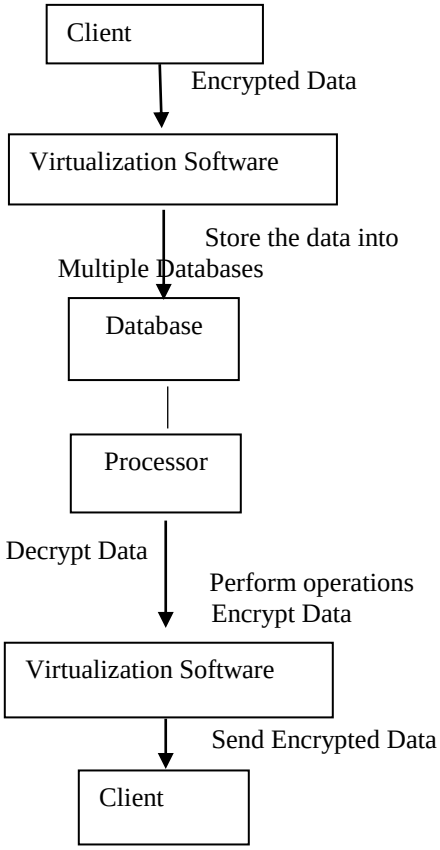

Fig 2: Flowchart of Existing System

As we can see in the above Figure 1, the normal processing of data takes place in the following way-

The Clients ask the Cloud Service Provider to grant the cloud storage which will be required.
The client then shares its data over the cloud.
The Cloud then stores the data in various databases along with processing operations in the given data.
The Cloud then sends the results back to the Server.

The Actual working of the whole system can be understood with a flow graph from Figure 2

The client encrypts the data and sends it to the cloud over the internet.
This data is received by the cloud and the virtualization software (Xen, HyperV) stores the data in multiple databases in order to obtain redundancy so that there is high fault tolerance.
Now in order to perform any operation to this data, the virtualization software instructs the processor to first decrypt the data, perform operations on it where only a

ISSN 2348 – 7968

single processor is used for a single user and then encrypt it again.
The encrypted data result is sent back to the client.

The disadvantages include:

Security
Privacy
Dependency
Downtime

4. Proposed System

4.1 Homomorphic Encryption

We propose a system design which will remove the barrier of security and privacy in cloud computing.
The vulnerability in existing cloud system design is that the client & the cloud service provider both have the encryption, decryption key known to each other. In other words, the cloud service provider has access to the client's data.
Now in order to remove this barrier, we must make sure that the cloud service provider doesn't know the encryption, decryption key except the clients who want to access the data using cloud. Thus we would like if the cloud server processes the operations on its data without decrypting it, which is done via Homomorphic Algorithm which has been long recognized.

The problem in implementing Complete Homomorphic Encryption Algorithm is the processing time and power required to process operations on the encrypted data without decrypting it.

4.2 Proposed Method to Implement Complete Homomorphic Encryption

The proposed method for implementing Complete Homomorphic Encryption is shown in Figure 3.

The problem of processing a huge encrypted data is solved by considering the processors connected in a grid such that a processor solves a given part of encrypted data assigned to it by the virtualization software and then submits it to the virtualization software.

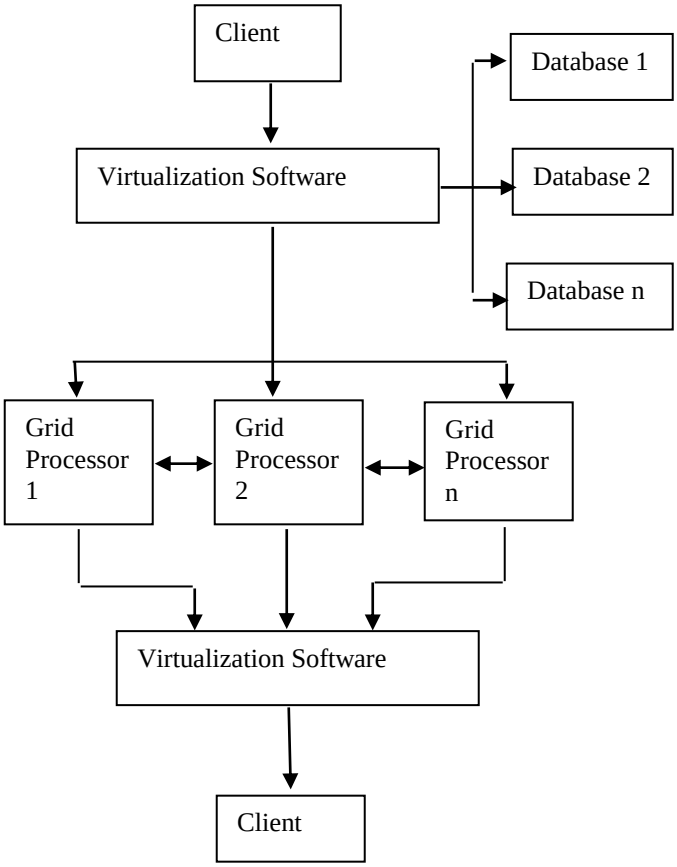

Fig 3: Proposed System

The virtualization software then sums up the assigned encrypted data to every processor into one forming the whole encrypted data on which operation has been performed and then sends it to the client thereby attaining Security- as the encrypted data ensures the prevention of Man in the Middle attacks as well as Denial of Service Attacks, Privacy- as the Cloud Service Provider doesn't know the decryption key of the data it has thereby providing restriction to unauthorized users and Performance- as the time required by the Cloud Service Provider to decrypt the data in order to perform operations as well as encrypting it again in order to send it to the client has been overcome by the Complete Homomorphic Encryption.
The following example illustrates the working of the Proposed System-

ISSN 2348 – 7968

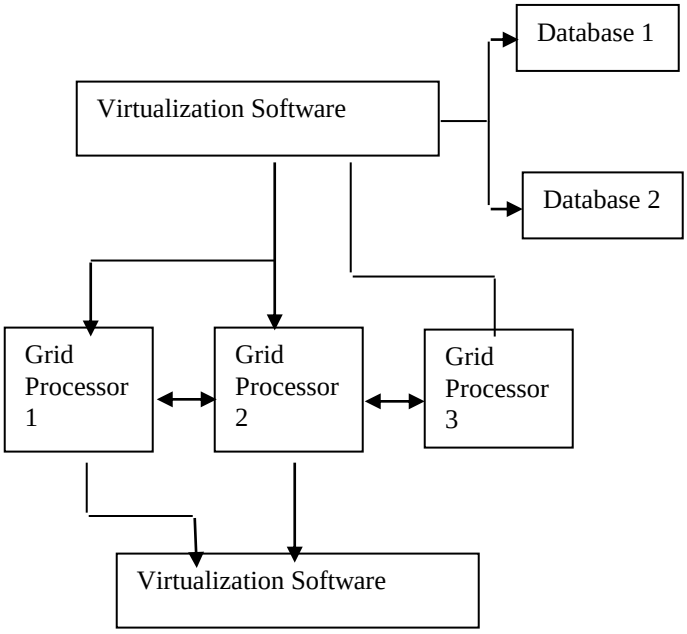

Fig 4: Internal Working of the Proposed System

Consider that there are 2 databases that store the same data in order to perform redundancy and there are 3 processors connected in a grid. Now, the following functioning takes place in the whole system-

The Client wants to send ((10+20) + (20+30))
The client uses the encryption key as "Divide by 2" there by sending the encrypted data as-
((5+10) + (10+15))
This operation is to be performed on the data residing in the databases 1 & 2.
Now, the virtualization software looks at the processors which are idle or having minimal workload, which is concluded as the Processors 1 & 2 are idle whereas the Processor 3 isn't free.
The Virtualization software thus divides the task to be performed into 2 Processors namely 1 & 2.
Here nobody except the client knows the decryption key which can be used to perform operations on the data, hence the processors have to deal with the encrypted data itself.
Thus the tasks performed are-

Processor 1- (5+10) => 15
Processor 2- (10+15) => 25

They send the results-15 & 25 to the Virtualization software which sums it up to 40.
Now, this value-40 is sent to the client without encrypting as it is already in the encrypted format
The client now receives the value 40 and then thus applying the decryption key as "multiply by 2" i.e. 40*2 => 80 which gives the same output it would have given by performing ((10+20) + (20+30)) with a single processor.
Thus the objective of achieving security, privacy & performance has been achieved by implementing Complete Homomorphic Encryption using grids.

5. Conclusion

Thus we have studied how to how to enable the complete processing of homomorphic encryption in cloud by using grids which performs operations on encrypted data simultaneously thereby utilizing the processors for maintaining efficient performance, privacy & security. This proposed system can be used where there are a lot of cloud resources being used.
Advantages include privacy, security, performance, efficiency whereas disadvantages include internet traffic, cloud database limit, number of processors available.

References

[1] Craig Gentry "A Fully Homomorphic Encryption Scheme", Stanford University, 2009.
[2] Jaliya Ekanayake, Xiaohong Qiu, Thilina Gunarathne Scott Beason, Geoffrey Fox, Pervasive Technology Institute, School of Informatics and Computing, Indiana University "High Performance Parallel Computing with Cloud and Cloud Technologies".
[3] Jian Li, Danjie Song, Sicong Chen, Xiaofeng Lu, IEEE, "A simple fully homomorphic encryption scheme available in cloud computing", 2012.
[4] Jing-Li-Han, Ming Yang, Cai-Ling Wang, Shan-Shan Xu, IEEE, "The Implemention and Application of Fully Homomorphic Encryption Scheme", 2012.
[5] Fau S., Sirdey R, Fontaine C, Aguilar-Melchor C., IEEE, "Towards Practical Program Execution over Fully Homomorphic Encryption Schemes", 2013.