International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 6 | ISSN : 2456-3307

Data Allocation in the Cloud Using Distributed Accountability

T Sharada, Nageswara Rao Jillapalli
Department of CSE, Osmania University, Amberpet, Hyderabad, Telangan, India

ABSTRACT

Now a day's Cloud Computing is the rapid growing technology. Now most of the persons are using Cloud Computing technology .Cloud computing enables highly scalable services to be easily consumed over the Internet on an as-needed basis. A major feature of the cloud services is that users' data are usually processed remotely in unknown machines that users do not own or operate. While enjoying the convenience brought by this new emerging technology, users' fears of losing control of their own data (particularly, financial and health data) can become a significant barrier to the wide adoption of cloud services. To solve the above problem in this paper we provide effective mechanism to using accountability frame work to keep track of the actual usage of the users' data in the cloud. In particular, we propose an object-centered approach that enables enclosing our logging mechanism together with users' data and policies. Accountability is checking of authorization policies and it is important for transparent data access. We provide automatic logging mechanisms using JAR programming which improves security and privacy of data in cloud. To strengthen user's control, we also provide distributed auditing mechanisms. We also provide secure JVM, this secure JVM is provide the high security to the user's or customers. We provide extensive experimental studies that demonstrate the efficiency and effectiveness of the proposed approaches.

Keywords: Cloud Computing, Logging, Audit Ability, Accountability, Data Sharing, Secure JVM.

I. INTRODUCTION

Cloud computing is the computing the resources (hardware and software) that are delivered as a service over a network .The name comes from the shape of the cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams. Cloud computing entrusts remote services with a user's data, software and computation. The below Fig.1.shows that overview of cloud computing. Cloud computing presents a new way to supplement the current consumption and delivery model for IT services based on the Internet, by providing for dynamically scalable and often virtualized resources as a service over the Internet. Now a day's most of the persons are accessing the large volumes of data from clouds. In this way they don't provide the security because of the wide adaption of cloud services. In this now I am provide the accountability and secure JVM. This two are providing the security.

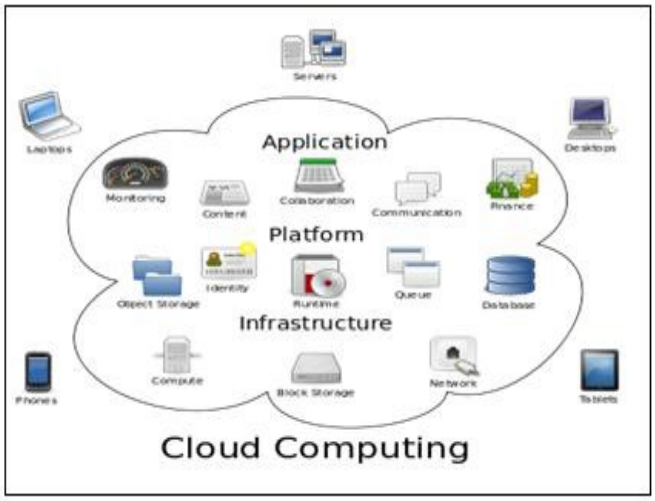

Figure 1. Overview of the cloud computing.

The architectural service layers of cloud computing are:

Software as a Service (SaaS): This service is a Software deployment model whereby a provider licenses an application to customers for use as a service on demand.

Examples:

Google app's, Salesforce.com, Social Networks.

Platform as a Service (PaaS): Optimized IT and developer tools offered through Platform as a Service (PaaS) for database and testing environments.

Examples:MS-Azure, Operating Systems, Infrastructure Scaling

Infrastructure as a Service (IaaS): On-demand highly scalable computing, storage and hosting services.

Examples: Mainframes, Storage

Cloud computing as a fast growing technology provides many scalable services. It moves user's data to the centralized large data centers, where the management of the data and services may not be fully trustworthy. Users may not know the machines which actually process and host their data in a cloud environment. Users also start worrying about losing control of their own data. The data processed on clouds are often outsourced, leading to a number of issues related to accountability, including the handling of personally identifiable information. Such fears are becoming a significant to the wide adoption of cloud services [1]. It is essential to provide an effective mechanism for users to monitor the usage of their data in the cloud. For example, users need to be able to ensure that their data are handled according to the service level agreements made at the time they sign on for services in the cloud.

Conventional access control approaches developed for closed domains such as databases and operating systems, or approaches using a centralized server in distributed environments, are not suitable, due to the following features characterizing cloud environments. First, data handling can be outsourced by the direct cloud service provider (CSP) to other entities in the cloud and theses entities can also delegate the tasks to others, and so on. Second, entities are allowed to join and leave the cloud in a flexible manner. As a result, data handling in the cloud goes through a complex and dynamic hierarchical service chain which does not exist in conventional environments.

To overcome the above problems, we propose a novel approach, namely Cloud Information Accountability (CIA) framework, based on the notion of information accountability [3]. Unlike privacy protection technologies which are built on the hide-it-or-lose-it perspective, information accountability focuses on keeping the data usage transparent and track able. Our proposed CIA framework provides end-to-end accountability in a highly distributed fashion. One of the main innovative features of the CIA framework lies in its ability of maintaining lightweight and powerful accountability that combines aspects of access control, usage control and authentication. By means of the CIA, data owners can track not only whether or not the service-level agreements are being honored, but also enforce access and usage control rules as needed. Associated with the accountability feature, we also develop two distinct modes for auditing: push mode and pull mode. The push mode refers to logs being periodically sent to the data owner or stakeholder while the pull mode refers to an alternative approach whereby the user (or another authorized party) can retrieve the logs as needed.

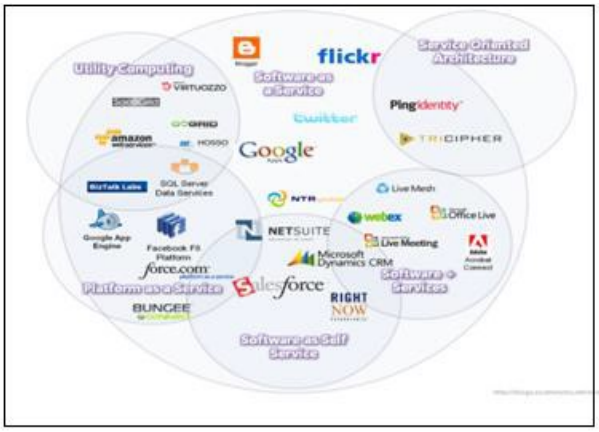

Figure 2. Cloud Computing Services

The above Fig.2.shows that cloud computing services. There is a number of notable commercial and individual cloud computing services, like Amazon, Google, Microsoft, Yahoo, and Sales force [2].

II. EXISTING PROBLEM STATEMENT

The use of Conventional access control approaches developed for closed domains such as databases and operating systems, or approaches using a centralized server in distributed environments, are not suitable, due to the two following reasons.

First, data handling can be outsourced by the direct cloud service provider (CSP) to other entities in the

cloud and theses entities can also delegate the tasks to others, and so on.

Second, entities are allowed to join and leave the cloud in a flexible manner.

As a result, data handling in the cloud goes through a complex and dynamic hierarchical service chain which does not exist in conventional environments [4].

In one existing system the user's private data are sent to the cloud in an encrypted form, and the processing is done on the encrypted data. The output of the processing is de-obfuscated by the privacy manager to reveal the correct result [5][6]. However, the privacy manager provides only limited features in that it does not guarantee protection once the data are being disclosed. In another existing system an agent-based system specific to grid computing. Distributed jobs, along with the resource consumption at local machines are tracked by static software agents. But it is mainly focused on resource consumption and on tracking of sub-jobs processed at multiple computing nodes, rather than access control.

Drawbacks from existing system given below:

1. The conventional access control approach must require any dedicated authentication and storage system.
2. The user cannot have the information regarding usage or access of data by other users.
3. Requires third-party services to complete the monitoring and focuses on lower level monitoring of system resources.

III. PROPOSED SYSTEM

The Cloud Information Accountability frame work proposed in this work conducts automated logging Distributed auditing of relevant access performed by any entity, carried out at any point of time at any cloud service provider [7]. It has two major components: logger and log harmonizer. The JAR file includes a set of simple access control rules specifying whether and how the cloud servers and possibly other data stakeholders are authorized to access the content itself. Apart from that we are going to check the integrity of the JRE on the systems on which the logger components is initiated. This integrity checks are carried out by using oblivious hashing .The proposed methodology will also take concern of the JAR file by converting the JAR into obfuscated code which will adds an additional layer of security to the infrastructure. Apart from that we are going to extend the security of user's data by provable data possessions for integrity verification. Depending on the configuration settings defined at the time of creation, the JAR will provide usage control associated with logging, or will provide only logging functionality. As for the logging, each time there is an access to the data, the JAR will automatically generate a log record. In the above proposed system, we provide secure JVM; this secure JVM is providing the high security to the data owners. A **Java virtual machine** (**JVM**) is a virtual machine that can execute Java byte code. It is the code execution component of the Java platform. Sun Microsystems has stated that there are over 5.5 billion JVM-enabled devices.

A Java virtual machine is a program which executes certain other programs, namely those containing Java byte code instructions. JVM's are most often implemented to run on an existing operating system, but can also be implemented to run directly on hardware. A JVM provides a run-time environment in which Java byte code can be executed, enabling features such as automated exception handling, which provides *root-cause* debugging information for every software error (exception). A JVM is distributed along with Java Class Library, a set of standard class libraries (in Java byte code) that implement the Java application programming interface (API). These libraries, bundled together with the JVM, form the Java Runtime Environment (JRE).

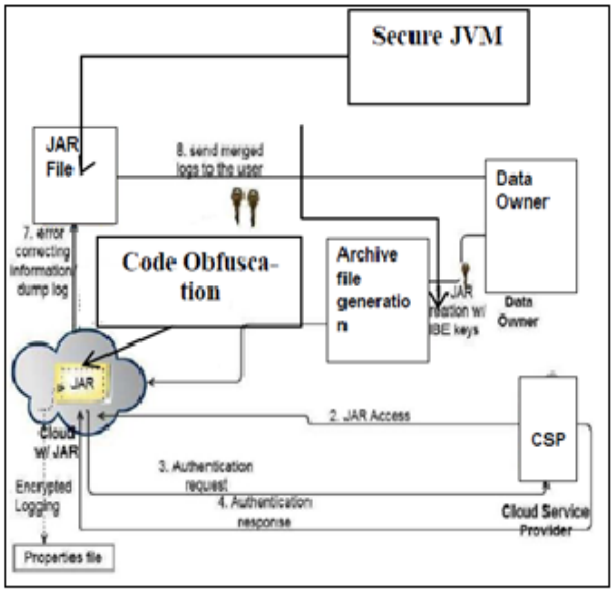

Figure 3. overview of the proposed system.

JVMs are available for many hardware and software platforms. The use of the same byte code for all JVMs on all platforms allows Java to be described as a *write once, run anywhere* programming language, versus *write once, compile anywhere*, which describes cross-platform compiled languages. Thus, the JVM is a crucial component of the Java platform. Java byte code is an intermediate language which is typically compiled from Java, but it can also be compiled from other programming languages. For example, Ada source code can be compiled to Java byte code and executed on a JVM.

Oracle Corporation, the owner of the *Java* trademark, produces the most widely used JVM, named Hotspot, that is written in the C++ programming language.JVMs using the *Java* trademark may also be developed by other companies as long as they adhere to the specification published by Oracle Corporation and to related contractual obligations.

IV. MODULES

The major buildings modules of proposed systems are Five. They are.

4.1. DATA OWNER MODULE
4.2. JAR CREATION MODULE
4.3. CLOUD SERVICE PROVIDER MODULE
4.4. Disassembling Attack
4.5. Man-in-the-Middle Attack

4.1. DATA OWNER MODULE:

In this module, the data owner uploads their data in the cloud server. The new users can register with the service provider and create a new account and so they can securely upload the files and store it. For the security purpose the data owner encrypts the data file and then store in the cloud. The Data owner can have capable of manipulating the encrypted data file. And the data owner can set the access privilege to the encrypted data file. To allay users' concerns, it is essential to provide an effective mechanism for users to monitor the usage of their data in the cloud. For example, users need to be able to ensure that their data are handled according to the service level agreements made at the time they sign on for services in the cloud.

4.2. JAR CREATION MODULE

In this module we create the jar file for every file upload. The user should have the same jar file to download the file. This way the data is going to be secured. The logging should be decentralized in order to adapt to the dynamic nature of the cloud. More specifically, log files should be tightly bounded with the corresponding data being controlled, and require minimal infrastructural support from any server. Every access to the user's data should be correctly and automatically logged. This requires integrated techniques to authenticate the entity who accesses the data, verify, and record the actual operations on the data as well as the time that the data have been accessed. Log files should be reliable and tamper proof to avoid illegal insertion, deletion, and modification by malicious parties. Recovery mechanisms are also desirable to restore damaged log files caused by technical problems. The proposed technique should not intrusively monitor data recipients' systems, nor it should introduce heavy communication and computation overhead, which otherwise will hinder its feasibility and adoption in practice.

4.3. CLOUD SERVICE PROVIDER MODULE

The cloud service provider manages a cloud to provide data storage service. Data owners encrypt their data files and store them in the cloud with the jar file created for each file for sharing with data consumers. To access the shared data files, data consumers download encrypted data files of their interest from the cloud and then decrypt them.

4.4. DISASSEMBLING ATTACK

In this module we show how our system is secured by evaluating to possible attacks to disassemble the JAR file of the logger and then attempt to extract useful information out of it or spoil the log records in it. Given the ease of disassembling JAR files, this attack poses one of the most serious threats to our architecture. Since we cannot prevent an attacker to gain possession of the JARs, we rely on the strength of the cryptographic schemes applied[9]to preserve the integrity and confidentiality of the logs. Once the JAR files are disassembled, the attacker is in possession of the public IBE key used for encrypting the log files, the encrypted log file itself, and the *.class files. Therefore, the attacker has to rely on learning the private key or subverting the encryption to read the log records. To compromise the confidentiality of the log files, the attacker may try to identify which encrypted log records correspond to his actions by mounting a chosen plaintext attack to obtain some pairs of encrypted log records and plain texts. However, the adoption of the Weil Pairing algorithm ensures that the CIA framework has both chosen cipher text security and chosen plaintext security in the random oracle model. Therefore, the attacker will not be able to decrypt any data or log files in the disassembled JAR file. Even if the attacker is an authorized user, he can only access the actual content file but he is not able to decrypt any other data including the log files which are viewable only to the data owner.1 From the disassembled JAR files, the attackers are not able to directly view the access control policies either, since the original source code is not included in the JAR files. If the attacker wants to infer access control policies, the only possible way is through analyzing the log file. This is, however, very hard to accomplish since, as mentioned earlier, log records are encrypted and breaking the encryption is computationally hard. Also, the attacker cannot modify the log files extracted from a disassembled JAR. Would the attacker erase or tamper a record, the integrity checks added to each record of the log will not match at the time of verification, revealing the error. Similarly, attackers will not be able to write fake records to log files without going undetected, since they will need to sign with a valid key and the chain of hashes will not match.

4.5. Man-in-the-Middle Attack.

In this module, an attacker may intercept messages during the authentication of a service provider with the certificate authority, and reply the messages in order to masquerade as a legitimate service provider. There are two points in time that the attacker can replay the messages. One is after the actual service provider has completely disconnected and ended a session with the certificate authority. The other is when the actual service provider is disconnected but the session is not over, so the attacker may try to renegotiate the connection. The first type of attack will not succeed since the certificate typically has a time stamp which will become obsolete at the time point of reuse. The second type of attack will also fail since renegotiation is banned in the latest version of OpenSSL and cryptographic checks have been added.

V. CLOUD INFORMATION ACCOUNTABILITY

In this section, we present an overview of the Cloud Information Accountability framework and discuss how the CIA framework [8] meets the design requirements discussed in the previous section. The Cloud Information Accountability framework proposed in this work conducts automated logging and distributed auditing of relevant access performed by any entity, carried out at any point of time at any cloud service provider.

It has two major components: logger and log harmonizer.

5.1. Major Components:

There are two major components of the CIA, the first being the logger, and the second being the log harmonizer. The logger is the component which is strongly coupled with the user's data, so that it is downloaded when the data are accessed, and is copied whenever the data are copied. It handles a particular instance or copy of the user's data and is responsible for logging access to that instance or copy. The log harmonizer forms the central component which allows the user access to the log files.

The logger is strongly coupled with user's data (either single or multiple data items). Its main tasks include automatically logging access to data items that it contains, encrypting the log record using the public key

of the content owner, and periodically sending them to the log harmonizer. It may also be configured to ensure that access and usage control policies associated with the data are honored. For example, a data owner can specify that user X is only allowed to view but not to modify the data. The logger will control the data access even after it is downloaded by user X.

The logger requires only minimal support from the server (e.g., a valid Java virtual machine installed) in order to be deployed. The tight coupling between data and logger, results in a highly distributed logging system, therefore meeting our first design requirement. Furthermore, since the logger does not need to be installed on any system or require any special support from the server, it is not very intrusive in its actions, thus satisfying our fifth requirement. Finally, the logger is also responsible for generating the error correction information for each log record and sends the same to the log harmonizer. The error correction information combined with the encryption and authentication mechanism provides a robust and reliable recovery mechanism, therefore meeting the third requirement. The log harmonizer is responsible for auditing. Being the trusted component, the log harmonizer generates the master key. It holds on to the decryption key for the IBE key pair, as it is responsible for decrypting the logs. Alternatively, the decryption can be carried out on the client end if the path between the log harmonizer and the client is not trusted. In this case, the harmonizer sends the key to the client in a secure key exchange.
It supports two auditing strategies: push and pull. Under the push strategy, the log file is pushed back to the data owner periodically in an automated fashion. The pull mode is an on-demand approach, whereby the log file is obtained by the data owner as often as requested. These two modes allow us to satisfy the aforementioned fourth design requirement. In case there exist multiple loggers for the same set of data items, the log harmonizer will merge log records from them before sending back to the data owner. The log harmonizer is also responsible for handling log file corruption. In addition, the log harmonizer can itself carry out logging in addition to auditing. Separating the logging and auditing functions improves the performance. The logger and the log harmonizer are both implemented as lightweight and portable JAR files. The JAR file implementation provides automatic logging functions, which meets the second design requirement.

5.1.1. Advantages:

One of the main innovative features of the CIA framework lies in its ability of maintaining lightweight and powerful accountability that combines aspects of access control, usage control and authentication. Providing defenses against man in middle attack, dictionary attack, Disassembling Attack, Compromised JVM At-tack, Data leakage attack.PDP allows the users to remotely verify the integrity of there data It's Suitable for limited and large number of storages.

5.2. Algorithm of Log Retrieval for Push and Pull mode:

Pushing or Pulling strategies have interesting tradeoffs. The pushing strategy is beneficial when there are a large number of accesses to the data within a short period of time. The pull strategy is most needed when the data owner suspects some misuse of his data; The pull mode allows him to monitor the usage of his content immediately. Supporting both pushing and pulling modes helps protecting from some nontrivial attacks. The algorithm presents logging and synchronization steps with the harmonizer.

The log retrieval algorithm for the push and pull modes:

```
Require: size:maximum size of the log file specified by
the data owner ,time:maximum time allowed to elapse
before the log file is dumped,tbeg:time stamp at which the
last dump occurred,log:current log file,pull:indicates
wheather a command from the data owner is received.

1:Let TS(NTP) be the network time protocal timestamp
2:pull=0
3:rec:=(UID,OID,AccessType,Result,Time,Loc)
4:CurrentTime:=TS(NTP)
5:lsize:=sizeof(log)//current size of the log
6:if((currenttime-tbeg)<time)&&(lsize<size)&&(pull==0)
then
7: log:=log+ENCRYPT(rec)//ENCRYPT is the encryption
function used to encrypt the record
8:  PING to CJAR//send a PING to the harmonizer to
check if it is alive
9: if PING-CJAR then
10: PUSH RS(rec)// write the error correcting bits
11: else
12: EXIT(1) // error if no PING is received
13:  end if
14: end if
15:  if((cutime-tbeg)> time)||(lsize >= size)||(pull !=0) then
16: //Check if PING is received
17: if PING-CJAR then
18: PUSH log // write the log file to the harmonizer
19: RS(log) := NULL // reset the error correction records
20: tbeg := TS(NTP) // reset the tbeg variable
21: pull := 0
22: else
23: EXIT(1) //error if no PING is received
24: end if
25: end if
```

Figure 3. Push and Pull Algorithm

5.2. Tools used for implementing cloud

In the proposed model we are using the following tools:

5.2.1. Eucalyptus Cloud

The Eucalyptus Cloud platform is open source software for building AWS-compatible private and hybrid clouds. Eucalyptus supports Amazon web services EC2 and S3 interfaces. It pools together existing virtualized infrastructure to create cloud resources for compute, network and storage.

5.2.2. Amazon EC2

Amazon Elastic compute cloud (EC2) is a central part of Amazon.com's cloud computing platform, Amazon web Services (AWS). EC2 allows users to rent virtual computers on which to run their own Computer Applications. They are designed for control and management of VM instances, EBS volumes, elastic IPs, and security groups and should work well with EC2 and Eucalyptus [10].

VI. CONCLUSION

This paper presents effective mechanism, which performs automatic authentication of users and create log records of each data access by the user. Data owner can audit his content on cloud, and he can get the confirmation that his data is safe on the cloud. Data owner also able to know the duplication of data made without his knowledge. Data owner should not worry about his data on cloud using this mechanism and data usage is transparent, using this mechanism. In future we would like to develop a cloud, on which we will install JRE and JVM, to do the authentication of JAR. Try to improve security of store data and to reduce log record generation time.

VII. REFERENCES

[1]. S.Pearson and A.Charlesworth, "Accountability as a Way Forward for Privacy Protection in the cloud", Proc.Frist Int'1 Conf. Cloud Computing, 2009.

[2]. P.T Jaeger, J .Lin and J.M. Grimes, "Cloud Computing and Information Policy: Computing in a Policy Cloud?", J. Information Technology and Policies, vol. 5, no. 3, pp. 269-289, 2009

[3]. D.J. Weitzner, H. Abelson, T. Berners-Lee, J. Feigen-baum, J. Handler, and G.J. Sussman, "Information Accountability", Comm.ACM,vol. 51,no. 6,pp. 82-87,2008

[4]. Ensuring Distributed Accountability for Data Sharing in the Cloud Author, Smitha Sundareswaran, Anna C.Squicciarini, Member, IEEE, and Dan Lin, IEEE Transactions on Dependable and Secure Computing ,VOL 9,NO,4 July/August 2012

[5]. S.Pearson, Y.Shen, and M. Mowbray, "A Privacy Manager for Cloud Computing", proc.Int'1 Conf.Cloud Computing (CloudCom),pp. 90-106,2009.

[6]. T.Mather, S.Kumaraswamy, and S.Latif, Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance (Theory in Practice), first ed. O'Reilly, 2009.

[7]. Nilutpal Bose, Mrs. G. Manimala, "SECURE FRAMEWORK FOR DATA SHARING IN CLOUD COMPUTING ENVIRONMENT, Website: www.ijetae.com, Volume 3, Special Issue 1, January 2013)

[8]. Ensuring Distributed Accountability for Data Sharing in the Cloud Author, Smitha Sundareswaran, Anna C.Squicciarini, Member, IEEE, and Dan Lin, IEEE Transactions on Dependable and Secure Computing ,VOL 9,NO,4 July/August 2012

[9]. D.Boneh and M.K. Franklin, "Identity-Based Encryption from the Weil Pairing", Proc.Int'1 Cryptography Conf.Advances in Cryptology,pp. 213-229,2001.

[10]. EucalyptusSystems, http://www.eucalyptus.com/2012.