International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 05 Issue: 05 | May-2018 www.irjet.net p-ISSN: 2395-0072

Sharing of Data through Distributed Accountability in the Cloud

Archana Karnik K.M

Assistant Professor ,Bachelor of Computer Applications,AIMS&R,Karnataka,India

---

Abstract - *Accountability is an important aspect of any computer system. It assures that every action executed in the system can be traced back to some entity. Accountability is even more crucial for assuring the safety and security. 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. Cloud computing enables highly scalable services to be easily consumed over the Internet on an as-needed basis. This paper proposes a Third party auditor(TPA) between data owner and cloud service provider(CSP) which reduce the burden of data owner to audit the data in the cloud and it also make the data owner free from worrying about the data lose in cloud storage . To highlight the security purpose a novel highly decentralized information accountability framework is introduced. When any access is made to the user's data will be trigger the authentication and automated logging control to JARs. A distributed auditing mechanism is used to control the users. To strengthen user's control, the framework also provide distributed auditing mechanism.*

Key Words: Cloud computing,TPA,CSP,CIA, Framework,JAR files,open SSL

1.INTRODUCTION

Cloud computing is the access to computers and their functionality via the Internet or a local area network. Users of a cloud request this access from a set of web services that manage a pool of computing resources (i.e. machines, network, storage, operating systems, application development environments, application programs). When granted, a fraction of the resources in the pool is dedicated to the requesting user until he or she releases them. It is called "cloud computing" because the user cannot actually see or specify the physical location and organization of the equipment hosting the resources they are ultimately allowed to use 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 obstacle to the wide acceptance of cloud services. Accountability is the obligation to act as a responsible steward of the personal information of others, to take responsibility for the protection and appropriate use of that information beyond mere legal requirements, and to be accountable for any misuse of that personal information. Associated with the accountability feature, two distinct modes for auditing are developed: 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 can retrieve the logs as needed.

2 LITERATURE SURVEY

Cloud computing has raised a range of important privacy and security issues. Such issues are due to the fact that, in the cloud, users' data and applications reside at least for a certain amount of time on the cloud cluster which is owned and maintained by a third party. In [1], Authors proposes a fully functional identity-based encryption scheme (IBE).This system is based on bilinear maps between groups. The Weil pairing on elliptic curves is an example of such a map. For the security of this IBE system, cipher text based encryption is chosen. The scheme has chosen cipher text security in the random oracle model assuming a variant of the computational Difie-Hellman problem. In[2], Authors propose the Security Assertion Markup Language (SAML) standard defines a framework for exchanging security information between online business partners. This document provides a technical description of SAML V2.0. In[5], Authors present an open framework for foundational proof-carrying code (FPCC). It allows program modules to be specified and certified separately using different type systems or program logics. Certified modules (i.e., code and proof) can be linked together to build fully certified systems. The framework supports modular verification and proof reuse. In[8], Authors used a signed application descriptor file instead of X.509 to authenticate a portable application code, such as java archive (JAR) file.

3. POSSIBLE ATTACKS TO THE FRAME WORK

The attackers may have sufficient Java programming skills to disassemble a JAR file and Prior Knowledge of the CIA architecture.

3.1 Copying attack

The most intuitive attack is that the attacker copies entire JAR files. The attacker may assume that doing so allows accessing the data in the JAR file without being noticed by the data owner. However, such attack will be detected by auditing mechanism that is every JAR file is required to send log records to the harmonizer. Even if the data owner is not aware of the existence of the additional copies of its JAR files, he will still be able to receive log files from all existing copies. Thus, the logger component provides more transparency than conventional log files encryption; it allows the data owner to detect when an

attacker has created copies of a JAR, and it makes offline files inaccessible.

3.2 Disassembling attack

Another possible attack is 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 the architecture. Since an attacker cannot be prevented to gain possession of the Jars, strong cryptographic schemes are applied to preserve the integrity and confidentiality of the logs. Since the encryptions are used, 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. 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.

3.3 Man-in-the-middle attack

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.

4. CLOUD INFORMATION ACCOUNTABILITY

The proposed Cloud Information Accountability framework 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.

4.1 CIA Frame work

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. 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.

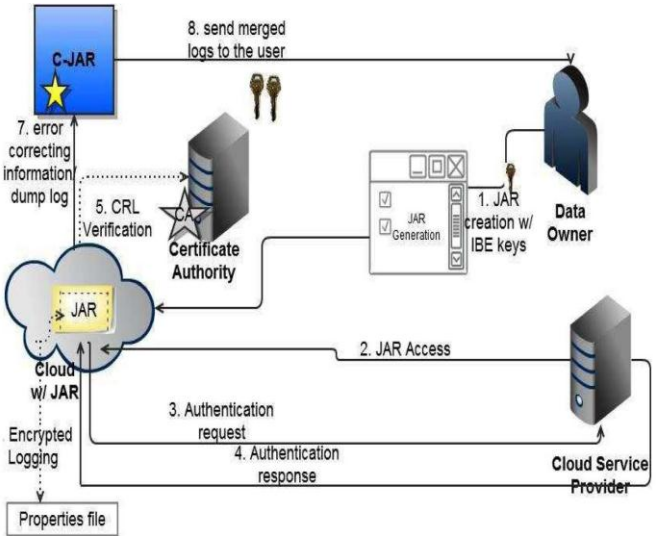

Fig 4.1 Architecture of the cloud information acccountability framework

The overall CIA framework, combining data, users, logger and harmonizer is sketched in Fig. 4.1 At the beginning, each user creates a pair of public and private keys based on Identity-Based Encryption [1] (step 1 in Fig.4.1). This IBE scheme is a Weil-pairing-based IBE scheme, which protects us against one of the most prevalent attacks to our architecture. The JAR file includes a set of simple access control rules specifying whether and how the cloud servers, and possibly other data stakeholders (users, companies) are authorized to access the content itself. Then, he sends the JAR file to the cloud service provider that he subscribes to. To authenticate the CSP to the JAR (steps 3-5 in Fig.4.1), OpenSSL based certificates are used, wherein a trusted certificate authority certifies the CSP. In the event that the access is requested by a user, SAML-based authentication [2] are used, wherein a trusted identity provider issues certificates verifying the user's identity based on his username.

Once the authentication succeeds, the service provider (or the user) will be allowed to access the data enclosed in the JAR. 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, encrypt it using the public key distributed by the data owner, and store it along with the data (step 6 in Fig.4.1). The encryption of the log file prevents unauthorized changes to the file by attackers. The data owner could opt to reuse the same key pair for all JARs or create different key pairs for separate JARs.

In addition, some error correction information will be sent to the log harmonizer to handle possible log file corruption (step 7 in Fig.4.1). To ensure trustworthiness of the logs, each record is signed by the entity accessing the content. The encrypted log files can later be decrypted and their integrity verified. They can be accessed by the data owner or other authorized stakeholders at any time for auditing purposes with the aid of the log harmonizer (step 8 in Fig.4.1).

Proposed framework prevents various attacks such as detecting illegal copies of user's data. This work is different from traditional logging methods which use encryption to protect log files. With only encryption, the logging mechanisms are neither automatic nor distributed. They require the data to stay within the boundaries of the centralized system for the logging to be possible, which is however not suitable in the cloud.

5 LOGGING MECHANISMS

A logger component is a nested Java JAR file which stores a user's data items and corresponding log files. The proposed JAR file consists of one outer JAR enclosing one or more inner JARs. The main responsibility of the outer JAR is to handle authentication of entities which want to access the data stored in the JAR file as shown in the fig 5.2. Each inner JAR contains the encrypted data, class files to facilitate retrieval of log files and display enclosed data in a suitable format, and a log file for each encrypted item as shown in Fig 5.1.

5.1 Inner jar

Figure 5.1 shows inner JAR data process, it has encrypted data and log record . The log record can be generated as pure log and access log where pure log have only the general information about every access made by users in the cloud and access log have general information and also the time duration about the access.

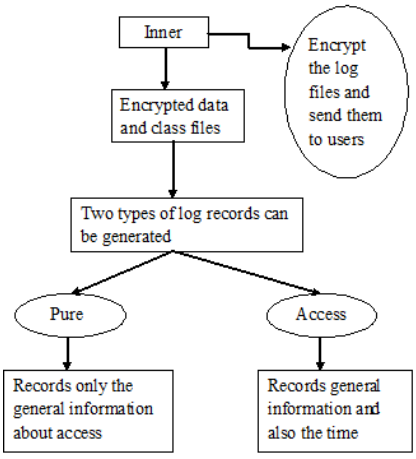

Fig 5.1 Inner JAR

5.2 Outer jar

Figure 5.2 shows the authentication process made in the outer JAR. It will authenticate the cloud service provider and users by means of the policies whenever the user do any malicious action it will be automatically intimated to the data owner.

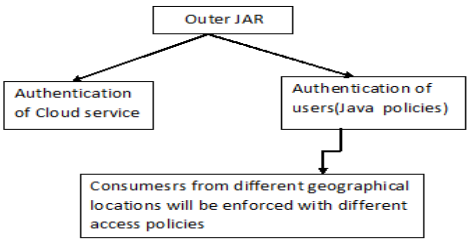

Fig:- 5.2 Outer JAR

6 PUSH & PULL MODE FOR AUDITING MECHANISM

Distributed auditing mechanism including the algorithms for data owners to query the logs regarding their data. Push and Pull Mode. To allow users to be timely and accurately informed about their data usage. Distributed logging mechanism is complemented by an innovative auditing mechanism. It support two complementary auditing modes:

1) push mode.
2) pull mode.

6.1 Push mode

In this mode, the logs are periodically pushed to the data owner (or auditor) by the harmonizer. This mode serves two essential functions in the logging architecture: 1) it ensures that the size of the log files does not explode and 2) it enables timely detection and correction of any loss or damage to the log files. By construction of the records, the auditor, will be able to quickly detect forgery of entries, sing the checksum added to each and every record.

6.2 Pull mode

This mode allows auditors to retrieve the logs anytime when they want to check the recent access to their own data. The pull message consists simply of an FTP pull command, which can be issues from the command line. For native users, a wizard comprising a batch file can be easily built. The request will be sent to the harmonizer, and the user will be informed of the data's locations and obtain an integrated copy of the authentic and sealed log file.

Algorithm

Let TS(NTP) be the network time protocol timestamp
Pull = 0
Rec := < UID,OID,AccessType,Result,Time,Loc>
Curtime := TS(NTP)
Lsize := sizeof(lig) //current size of the log

```
If ((curtime - tbeg) < time) && ( lsize < size) && (pull == 0)
then
log := log + ENCRYPT(rec)  //ENCRYPT is the encryption
function used to encrypt the record
PING to CJAR //send a PING to the harmonizer to check if
it is alive
If PING - CJAR then
PUSH RS(rec)  //write the error correcting bits
Else
EXIT(1)  //error if no PING is received
End if
End if
If ((curtime - tbeg) >  time) ||  (lsize >= size) || (pull ≠  0)
then
If PING - CJAR then  //check if PING is received
PUSH log  // write the log file to the harmonizer
RS(log) := NULL  //reset the error correction records
Tbeg := TS(NTP)  //reset the tbeg variable
Pull := 0
Else
EXIT(1)  //error if no PING is received
End if
End if
```

Fig. 6.1. Push and pull mode.

Here 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 : timestamp at which the last dump occurred, log : current log file, pull : indicates whether a command from the data owner is received.

7. CONCLUSIONS

The proposed approach allows the third party auditor to audit, not only audit the data but also enforce strong backend protection if needed. One of the main features of the CIA frame work is that it enables the third party auditor to audit even those copies of its data that were made without his knowledge.

REFERENCES

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

[2] OASIS Security Services Technical Committee, "Scurity Assertion Markup Language2.0,"http://www.Oasis open.org/committees/tchomephp? Wgabb rey=security, 2012

[3] R. Corin, S. Etalle, J.I. den Hartog, G. Lenzini, and I. Staicu, "A Logic for Auditing accountability in Decentralized Systems,"Proc. IFIP TC1 WG1.7 Work shop Formal aspects in Security and Trust,pp. 187-201, 2005.

[4] Y. Chen et al., "Oblivious Hashing: A Stealthy Soft Ware Integrity Verification Primitive," Proc. Int'l Workshop Information Hiding, F. Petitcolas, ed., pp. 400-414, 2003.

[5] X. Feng, Z. Ni, Z. Shao, and Y. Guo, "An Open Frame work for Foundational Proof Carrying Code," Proc. ACM SIGPLAN Int'l Workshop Types in Lan Gauges Design and Implementation, pp. 67-78,2007.

[6] P.T. Jaeger, J. Lin, and J.M. Grimes, "Cloud Compu Ting and Information Policy: Computing in a Policy Cloud?," J. Information Technology and Politics, vol. 5, no. 3, pp. 269-283, 2009.

[7] R. Jagadeesan, A. Jeffrey, C. Pitcher, and J. Riely, "Towards a Theory of Accountability and Audit," Proc. 14th European Conf.Research in Computer security (ESORICS), pp. 152-167, 2009.

[8] J.H. Lin, R.L. Geiger, R.R. Smith, A.W. Chan, and S. Wanchoo, Method for Authenticating a Java Archive(jar) for Portable Devices, US Patent 6,766,353, July 2004.