IJREAT International Journal of Research in Engineering & Advanced Technology, Volume 3, Issue 3, June-July, 2015
ISSN: 2320 – 8791 (Impact Factor: 2.317)
www.ijreat.org

A Key Approach To Maintain Security Through Log Management For Data Sharing In Cloud

Shriram Vaidya[1], Prof. Aruna Gupta[2]

[1,2]**Computer Department, Savitribai Phule Pune University, JSCPE Hadapsar, Pune, Maharashtra State 411028, India**

Abstract

Cloud computing ensures highly scalable services to be easily available on the Internet. Users' data are usually processed remotely in unknown machines that users do not own in cloud. While enjoying the convenience of this new emerging technology, users' are worried about losing control of their own data mainly in financial and health sectors. This is becoming a significant problem to the wide use of cloud services. To solve this problem, a novel highly decentralized information accountability framework that keeps track of the actual usage of the users' data is proposed in the cloud.

It contains an object-centered approach that enables enclosing our logging mechanism together with users' data and policies. For this we leverage the JAR programmable capabilities to both create a dynamic and traveling object, and to ensure that any access to users' data will trigger authentication and automated logging local to the JARs. In addition it also provides distributed auditing mechanisms and sends notification of data access via SMS (Short Message service) to data owners' mobile.

Keywords: *Cloud Computing, accountability, data sharing, auditing, log management.*

1. Introduction

Cloud computing provides cloud services and their delivery to users. Cloud computing is a technology which uses internet and remote servers to stored data and application and provides on demand services [2]. It also supplements the current consumption and delivery of services based on the Internet. Cloud users worried about losing control of their own data while enjoying the convenience brought by this new technology. The data processed on clouds are generally outsourced that leads to a number of issues related to accountability that includes the handling of personally identifiable information. These are becoming barriers to the wide adoption of cloud services [3] [4]. These problems can solve by the proposed topic. In Cloud, Cloud Service Provider transfers their work to other entities hence, data sharing goes through complex and dynamic service chain. To solve this problem the concept of **CIA** (Cloud Information Accountability) is proposed that is based on Information Accountability which focuses on keeping the data usage transparent and traceable [2] [3]. CIA is able to provide end to end accountability. CIA has ability of maintaining lightweight and powerful accountability that combines aspects of access control, usage control and authentication through JAR. JAR which has programming capability is used in proposed work. There are two distinct modes for auditing provided in CIA i.e. 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. JAR (Java Archives) files automatically log the usage of the user's data by any entity in the cloud to ensure the reliability of the log, adapting to a highly decentralized infrastructure etc. Data owner will get log files through which he/she can detect who is using their data. This ensures distributed accountability for shared data in cloud.

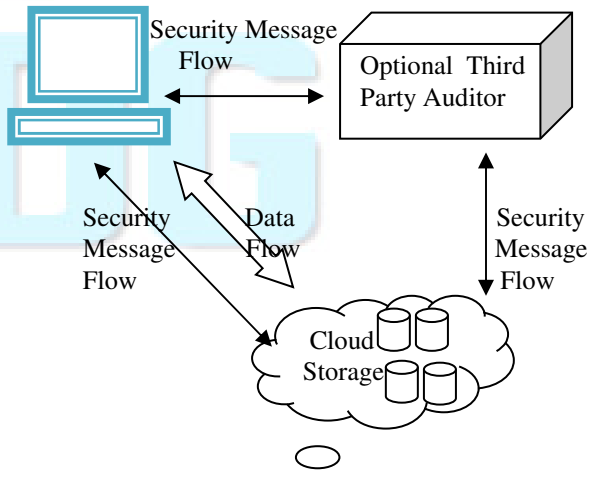


Figure-cloud data storage architecture that is used to share data on cloud.

1.1Accountability

Accountability is fundamental concept in cloud that helps for growth of trust in cloud computing. The term Accountability refers to a contracted and accurate requirement that met by reporting and reviewing mechanisms. Accountability is the agreement, to act as an authority to protect the personal information from others [2]. Accountability is used for security and protects against use of that information beyond legal boundaries and it will be held responsible for misuse of that information. Here Users data along with any policies such as access control policies and logging policies that they want to enforce are encapsulate in JAR and user will send that JAR to cloud service providers[1][3]. Any access to the data will trigger an automated and authenticated logging mechanism local to the JARs; we call it as Strong Binding. Moreover JARs are able to send error correction information which allows it to monitor the loss of any logs from any of the JARs. To test or experiment the performance proposed work focus on image, pdf, swf files which are common content type for end user. We send log files to users' mobile phone as notification about who is accessing his data in the proposed work, architecture used is platform independent and highly decentralized which does not required any dedicated authentication or storage system in place.

2. RELATED WORK

Cloud computing has raised an amount of privacy and security issues [4]. Such issues are due to outsourcing of data to third party by cloud service providers. As Cloud store the amount of users' data, for that well- known security is very important. The vendor of the data does not aware about where their data is stored and they do not have control of where data to be placed. Pearson et al. have proposed accountability to address privacy concerns of end users [4] and then develop a privacy manager [5] [9].

Attribute Based Encryption *Sahai* and *Waters* proposed notion of ABE and it was introduced as a new method for fuzzy identity based encryption [5]. The first drawback of the scheme is that its threshold semantics lacks expressibility. Several efforts followed in the literature that tries to solve the expressibility problem. In the ABE scheme, cipher texts are not encrypted to one particular user as in traditional public key cryptography. Rather, both cipher texts and users' decryption keys are associated with a set of attributes or a policy over attributes. A user is able to decrypt a cipher text only if there is a match between his decryption key and the cipher text. ABE schemes are classified into key policy attribute- based encryption (KP-ABE) [5] and cipher text-policy attribute- based encryption (CPABE), depending how attributes and policy are associated with cipher texts and users' decryption keys. *Jagadeesan et al.* proposed logic for designing accountability-based distributed systems in his "Towards a Theory of Accountability and Audit" journal. In that the usage of policies attached to the data and presents logic for accountability data in distributed settings. The few existing systems are partial and restricted to single domain [4][5].

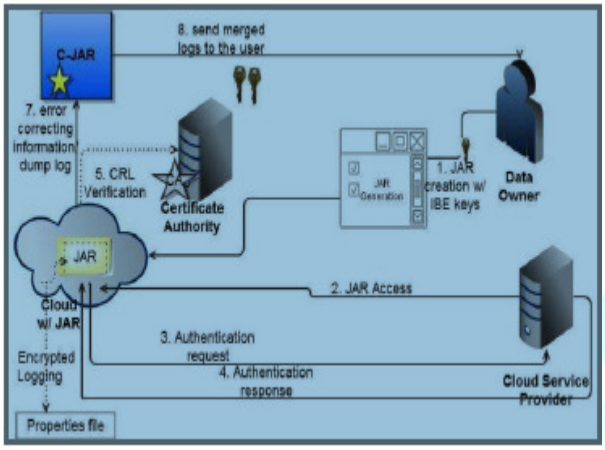
2.1 Other Techniques

J.W. Holford proposed that, With respect to Java-based techniques for security, here methods are related to self defending objects (SDO). SDOs are an extension of the object oriented programming paradigm, where software objects that offer sensitive functions or hold sensitive data and that are responsible for protecting those data. The key difference in implementations is that the authors still rely on a centralized database to maintain the access records, while the items being protected are held as separate files. Previous work provided a Java-based approach to prevent privacy leakage from indexing which could be integrated with the CIA framework proposed in this work so they build on related architectures. In this proposed paper we do not cover issues related to data storage security which are a complementary aspect of the privacy issues.

3. IMPLEMENATATION

There is a need to provide technique which will audit data in widespread use of cloud. On the basis of accountability, we proposed one mechanism which keeps use of data transparent means data owner should get information about usage of his data. This mechanism support accountability in distributed environment. Data owner should not bother about his data. He may know that this data is handled according to service level agreement and his data is safe on cloud. In proposed work, we consider three kinds of files that are image file, swf files for video and pdf file documents. A logger component is consists of Java JAR file which stores a user's data items and corresponding log files

Figure: Architecture



The following diagram shows model architecture that specifies how data flows from one entity to another entity.

As shown in figure first user creates public and private key based on IBE (Identity Based Encryption) [3]. Logger component which is a JAR File will be created by user with the generated keys to store its data items. The JAR includes some set of simple access controls that specifies how those entities are authorized to access the contents. Then user sends JAR file to Cloud Service provider (CSP) that he/she subscribes to. (steps 3-5) Open SSL based certificates are used to authenticate CSP to the JAR wherein a trusted certificate authority certifies the CSP.

In event where access is requested by user occurred then we employ SAML-based authentication where a trusted identity provider issues certificates verifying the user's identity based on his username [7]. The CSP service provider or user will be allowed to access the data enclosed in the JAR when authentication completed. The JAR will provide usage control associated with logging depending on configuration setting defined at time of JAR creation. Whenever there is access to data, log is automatically generated by JAR [13][3]. That log file is encrypted by public key distributed by user and is stored along with data in step 6 in figure. Encryption ensures unauthorized changes to log files. Some error correction information will be sent to the log harmonizer to handle possible log file corruption. Individual records are hashed together to create a chain structure and that are able to quickly detect possible errors or missing records. Data owner and other stake holder can access data at time of auditing with aid to log harmonizer.

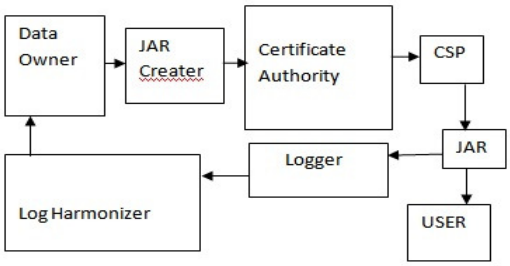


Figure-Accountability mechanism

Since the logger does not need to be installed on any system or don't require any special support from the server. It is not very intrusive in its actions. Since, the logger is 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 requirement.

A logger component is consists of Java JAR file which stores a user's data items and corresponding log files Log records are automatically generated by the logger component. Logging occurs at any access to the data in the JAR, and new log entries are appended sequentially, Each record ri is encrypted individually and appended to the log file. Here, ri indicates that an entity identified by ID has performed an action Act on the user's data at particular time. The generated log file's notification say short message service (SMS) will be sent to data owners' mobile. The SMS will contain IP address of user who accessed data, access type, date and time.

Algorithm for Push and Pull mode

Size: maximum size of 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 whether the command from owner is received.

1. Let TS(NTP) be the network time protocol timestamp
2. Pull=0
3. Rcc={UID,OID,ACCESSTYPE,Result,Time,Date}
4. Curtime=TS(NTP)
5. Lsize=size,of(log) // Current size of log

6. If ((curtime –tbeg) <time) && (lsize <size) && (pull= =0) then

7. Log=log+ENCRYPT(rcc) // Encryption function used to encrypt the record

8. PING to CJAR // send ping to harmonizer to check if it is alive
9. If PING-CJAR then
10. PUSH RS(rss) // write error correcting bits.
11. Else
12. Exit(1) // error if no PING is received
13. End if
14. End if
15. If ((curtime –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 harmonizer
19. RS(log):=NULL //reset the error correcting records
20. Tbcg=TS(NTP)// reset the tbcg variable
21. Pull:=0
22. Else
23. EXIT(1) // error if no PING is received
24. End if
25. End if

In proposed work, the algorithm uses for push and pull model presents logging and synchronization steps with the harmonizer in Push mode. First, the algorithm checks whether the size of the JAR has exceeded a fixed size or the normal time between two consecutive dumps has elapsed. The size and threshold time for a dump are specified by the data owner at the time of creation of the JAR. It also checks whether the data owner has requested a dump of the log files. If events has not occurred then it proceeds to encrypt the record and write the error correction information to the harmonizer. The communication with the harmonizer begins with a simple handshake. If no response is received, the log file records an error. If the JAR is configured to send error notifications then the data owner is then alerted through notifications. Once the handshake is completed, the communication with the harmonizer proceeds using a TCP/IP protocol [3][13]. If any request of the log file, or the size or time exceeds the threshold has occurred, the JAR simply dumps the log files and resets all the variables to make space for new records. In case of Access Log (Pull mode), the algorithm is modified by adding an additional check after some step 6. Precisely, the Access Log (Pull mode) checks whether the CSP accessing the log satisfies all the conditions specified in the policies pertaining to it. If the conditions are satisfied, access is granted; otherwise, access is denied. Irrespective of the access control outcome, the attempted access to the data in the JAR file will be logged.

4. CONCLUSION and FUTURE SCOPE

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 modification 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. The kind of security is regarding with only image file which are often content in cloud, In future we can be able to add security to all types of files shared in cloud. This will give better approach which provides authentication, accountability and auditing features. We can also use encrypted algorithm to ensure security but it leads to increase in cost and time. The notifications can also send to genuinely used devices of data owner like mobile phones or iphones. In future we would like to develop a cloud, on which we will install JRE and JVM, to do the authentication of JAR. Proposed work tried to improve security of store data and to reduce log record generation time.

5. ACKNOWLEDGMENT

I wish to express my sincere thanks and deep gratitude towards Dr.M.G Jadhav [Principal JSCOE] and my guide Prof. Aruna Gupta for her guidance, valuable suggestions and constant encouragement in all phases. I am highly indebted to her help in solving my difficulties which came across whole Paper work. Finally I extend my sincere thanks to respected Head of the department Prof. S. M. Shinde and Prof. M .D. Ingle [P.G Co- Ordinator] and all the staff members for their kind support and encouragement for this paper. I extend my thanks to coordinators of organization for providing us platform.
Last but not the least, I wish to thank my friends.

6. References

[1]. Yathi Raja Kumar Gottapu and A. Raja Gopal Paper: "OPTIMAL STORAGE SERVICES IN CLOUD COMPUTING ENSURING SECURITY" *International*

Journal of Advanced Research in IT and Engineering ISSN: 2278-6244

[2]. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," Proc. ACM Conf. Computer and Comm. Security, pp. 598- 609, 2007.

[3]. Smitha Sundareswaran, Anna C. Squicciarini and DanLin,"Ensuring Distributed Accountability for Data Sharing in the Cloud," IEEE Transaction on dependable a secure computing, VOL. 9, NO 4, pg 556-568, 2012

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

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

[6]. B. Chun and A.C. Bavier, "Decentralized Trust Management and Accountability in Federated Systems," Proc. Ann. Hawaii Int'l Conf.System Sciences (HICSS), 2004.

[7]. OASIS Security Services Technical Committee, "Security Assertion Markup Language (saml) 2.0," http://www.oasis-open.org/committees/tc home.php?wg abbrev=security, 2012.

[8]. S. Pearson , Y. Shen, and M. Mowbray," A privacy Manager for Cloud Computing," Proc. Int'l Conf Cloud Computing (cloudcom),pp.90- 106,2009.

[10]. S. Pearson and A. Charlesworth, "Accountability as a Way Forward for Privacy Protection in the Cloud," Proc First Int'l conf.Cloud Computing, 2009.

[11]. Flickr, http://www.flickr.com/, 2012.

[12]. S. Sundareswaran, A. Squicciarini, D. Lin, and S. Huang, ―Promoting Distributed Accountability in the Cloud,‖ *Proc. IEEE* Int'l Conf. Cloud Computing, 2011.

[13]. ASHWINI N S AND MRS. TAMILARASI T "DISTRIBUTED ACCOUNTABILITY AND AUDITING IN CLOUD" International Journal of Internet Computing (IJIC) ISSN No: 2231 –6965, Vol-2, Iss-1, 2013.