ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, Issue 4, April 2013

DATA SHARING IN THE CLOUD USING DISTRIBUTED ACCOUNTABILITY

Epuru Madhavarao[1], M Parimala[2],Chikkala JayaRaju[3]

***Abstract*— 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 provide extensive experimental studies that demonstrate the efficiency and effectiveness of the proposed approaches.**

***Keywords*- cloud computing, logging, auditability, accountability, data sharing**

1. INTRODUCTION

Cloud computing is a technology which uses internet and remote servers to store data and application. In cloud there is no need to install particular hardware, software on user machine, so user can get the required infrastructure on his machine in cheap charges/rates. Cloud computing is an infrastructure which provides useful, on demand network services to use various resources with less effort. Features of Cloud computing are, huge access of data, application, resources and hardware without installation of any software, user can access the data from any machine or anywhere in the world, business can get resource in one place, that's means cloud computing provides scalability in on demand services to the business users. Everyone kept their data in cloud, as everyone kept their data in cloud so it becomes public so security issue increases towards private data. Data usage in cloud is very large by users and businesses, so data security in cloud is very important issue to solve. Many users want to do business of his data through cloud, but users may not know the machines which actually process and host their data. While enjoying the convenience brought by this new technology, users also start worrying about losing control of their own data [1], [8].

Cloud provides three service models, which are; platform as a service, infrastructure as a service and software as a service. Under the Database as a service, this is having four parts which are as per mentioned below,

- Encryption and Decryption - For security purpose of data stored in cloud, encryption seems to be perfect security solution.
- Key Management - If encryption is necessary to store data in the cloud, encryption keys can't be store their, so user requires key management.
- Authentication - For accessing stored data in cloud by authorized users.
- Authorization – Rights given to user as well as cloud provider.

To solve the security issues in cloud; other user can't read the respective users data without having access. Data owner should not bother about his data, and should not get fear about damage of his data by hacker; there is need of security mechanism which will track usage of data in the cloud. Accountability is necessary for monitoring data usage, in this all actions of users like sending of file are cryptographically linked to the server, that performs them and server maintain secured record of all the actions of past and server can use the past records to know the correctness of action. It also provides reliable information about usage of data and it observes all the records, so it helps in make trust, relationship and reputation. So accountability is for verification of authentication and authorization. It is powerful tool to check the authorization policies [9].Accountability describes authorization requirement for data usage policies. Accountability mechanisms, which rely on after the fact verification, are an attractive means to enforce authorization policies [7].

There are 7 phases of accountability

1. Policy setting with data
2. Use of data by users
3. Logging
4. Merge logs
5. Error correctness in log
6. Auditing
7. Rectify and improvement.

These phases may change as per framework

First the data owner will set the policies with data and send it to cloud service provider (CSP), data will be use by users and

Manuscript received Feb, 2013.

Epuru Madhavarao, Department of CSE, M.Tech, Vignan's Lara Institute of Technology & Science, Vadlamudi, AP,India.

M Parimala, Department of CSE,Asst Professor, Vignan's Lara Institute of Technology & Science,Vadlamudi,AP,India.

Chikkala JayaRaju, Department of CSE, M.Tech,, Vignan's Lara Institute of Technology & Science, Vadlamudi, AP,India.

logs of each record will be created, then log will be merged and error correction in log has been done and in auditing logs are checked and in last phase improvement has been done [12].

Cloud provides three service models, which are; platform as a service, infrastructure as a service and software as a service. Under the Database as a service, this is having four parts which are as per mentioned below,

- Encryption and Decryption - For security purpose of data stored in cloud, encryption seems to be perfect security solution.
- Key Management - If encryption is necessary to store data in the cloud, encryption keys can't be store their, so user requires key management.
- Authentication - For accessing stored data in cloud by authorized users.
- Authorization – Rights given to user as well as cloud provider.

To solve the security issues in cloud; other user can't read the respective users data without having access. Data owner should not bother about his data, and should not get fear about damage of his data by hacker; there is need of security mechanism which will track usage of data in the cloud. Accountability is necessary for monitoring data usage, in this all actions of users like sending of file are cryptographically linked to the server, that performs them and server maintain secured record of all the actions of past and server can use the past records to know the correctness of action. It also provides reliable information about usage of data and it observes all the records, so it helps in make trust, relationship and reputation. So accountability is for verification of authentication and authorization. It is powerful tool to check the authorization policies [9].Accountability describes authorization requirement for data usage policies. Accountability mechanisms, which rely on after the fact verification, are an attractive means to enforce authorization policies [7].

There are 7 phases of accountability

1. Policy setting with data
2. Use of data by users
3. Logging
4. Merge logs
5. Error correctness in log
6. Auditing
7. Rectify and improvement.

These phases may change as per framework

First the data owner will set the policies with data and send it to cloud service provider (CSP), data will be use by users and logs of each record will be created, then log will be merged and error correction in log has been done and in auditing logs are checked and in last phase improvement has been done [12].

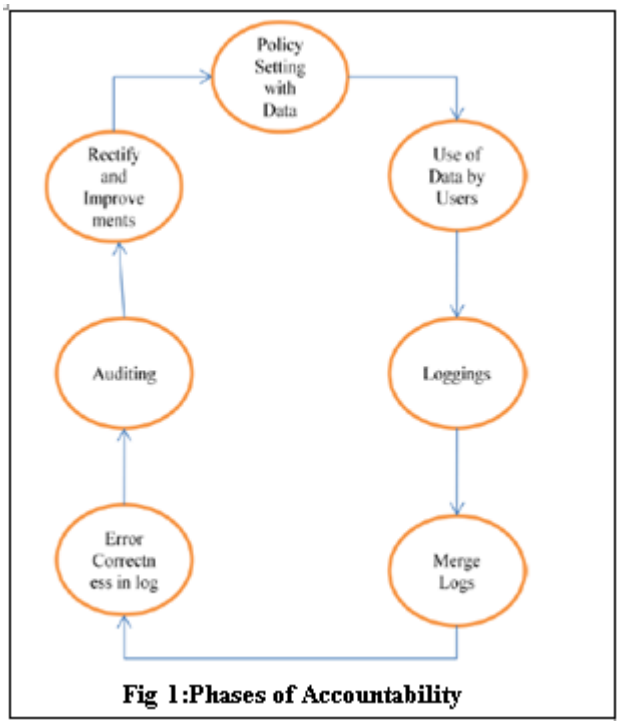

Fig 1:Phases of Accountability

In the Fig 1 Steps of accountability is given these are 7 steps each step is important to perform next step, accountability is nothing but validation of user actions means user having rights for accessing this data or not. Suppose user will do misuse of data or resources then network or data owner will take action on it so users, businesses and government should not bother about their data on cloud.

2. LITERATURE SURVEY

In this section review related works addressing security in cloud. Security issue is very important in cloud there are many techniques available so here is review of all these.

S. Pearson et al describes privacy manager mechanism in which user's data is safe on cloud , in this technique the user's data is in encrypted form in cloud and evaluating is done on encrypted data, the privacy manager make readable data from result of evaluation manager to get the correct result. In obfuscation data is not present on Service provider's machine so there is no risk with data, so data is safe on cloud, But this solution is not suitable for all cloud application, when input data is large this method can still require a large amount of memory[2]. In [3], the authors present procedural and technical solution both are producing solution to accountability to solving security risk in cloud in this mechanism these policies are decided by the parties that use, store or share that data irrespective of the jurisdiction in which information is processed. But it has limitation that data processed on SP is in unencrypted at the point of processing so there is a risk of data leakage. In [4], the author gives a language which permits to serve data with policies by agent; agent should prove their action and authorization to use particular data. In this logic data owner attach Policies with data, which contain a description of which actions are allowed with which data, but there is the problem of Continuous auditing of agent, but they provide solution that incorrect behavior. Should monitor and agent should give justification for their action, after that authority will check the

justification. In [5], authors gives a three layer architecture which protect information leakage from cloud, it provides three layer to protect data, in first layer the service provider should not view confidential data in second layer service provider should not do the indexing of data, in third layer user specify use of his data and indexing in policies, so policies always travel with data. In [6], authors present accountability in federated system to achieve trust management. The trust towards use of resources is accomplished through accountability so to resolve problem for trust management in federated system they have given three layers architecture, in first layer is authentication and authorization in this authentication does using public key cryptography. Second layer is accountability which perform monitoring and logging. The third layer is anomaly detection which detects misuse of resources. This mechanism requires third party services to observe network resources.

3. PROPOSED WORK

Cloud computing is a large infrastructure which provide many services to user without installation of resources on their own machine. This is the pay as you use model. Examples of the cloud services are Yahoo email, Google, Gmail and Hotmail. There are many users, businesses, government uses cloud, so data usage in cloud is large. So data maintenance in cloud is complex. Many Artists wants to do business of their art using cloud. For example one of the artist want to sell his painting using cloud then he want that his paintings must be safe on cloud no one can misuse his paintings.

There is need to provide technique which will audit data in 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 his data is handled according to service level agreement and his data is safe on cloud. Data owner will decide the access rules and policies and user will handle data using this rule and logs of each data access have been created. In this mechanism there are two main components i.e. logger and log harmonizer.

The logger is with the data owner's data, it provides logging access to data and encrypts log record by using public key which is given by data owner and send it to log harmonizer. The log harmonizer is performing the monitoring and rectifying, it generates the master key it holds decryption key decrypting the logs, and at the client side decryption it sends key to client. In this mechanism data owner will create private key and public key, using generated key owner will create logger which is a JAR file (JAVA Archives), it includes his policies like access policies and logging policies with data send to cloud service provider.

Authentication of cloud service provider has been done using open SSL based certificates after authentication of cloud service provider user can be able to access data in JAR, log of each data usage has been created and encrypted using public key and it automatically send to log harmonizer for integrity log records are signed by entity which is using the data and log records are decrypted and access by owner. In push mode logs are automatically send to data owner and in pull mode owner can demand logs, so he can see access of his data at anytime, anywhere and he can do monitoring of his data [1].

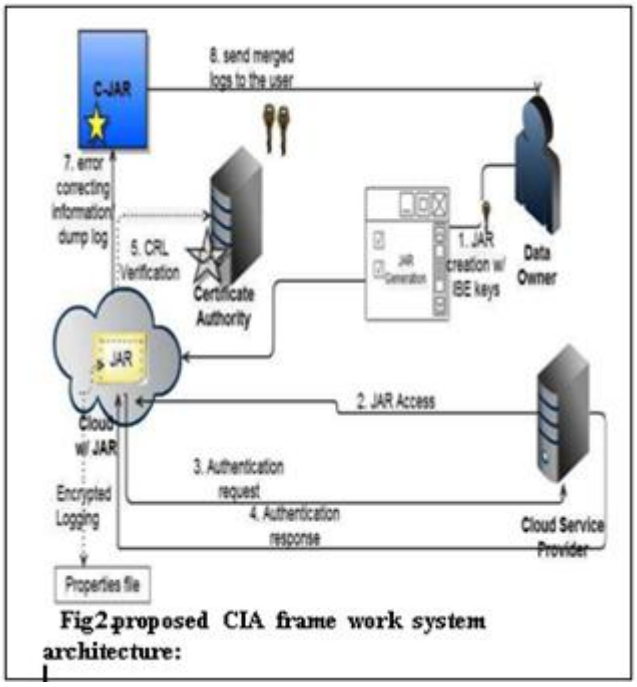

Fig2 proposed CIA frame work system architecture:

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 (in Fig. 2). 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. Using separate keys can enhance the security without introducing any overhead except in the initialization phase. In addition, some error correction information will be sent to the log harmonizer to handle possible log file corruption (in Fig. 2). To ensure trustworthiness of the logs, each record is signed by the entity accessing the content. Further, individual records are hashed together to create a chain structure, able to quickly detect possible errors or missing records. 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 (in Fig. 2).

As discussed , our proposed framework prevents various attacks such as detecting illegal copies of users' data. Note that our work is different from traditional logging methods which use encryption to protect log files. With only encryption, their 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.

3.1. Push and Pull Mode

To allow users to be timely and accurately informed about their data usage, our distributed logging mechanism is complemented by an innovative auditing mechanism. We support two complementary auditing modes: 1) push mode; 2) pull mode.

Push mode. In this mode, the logs are periodically pushed to the data owner (or auditor) by the harmonizer. The push action will be triggered by either type of the following two events: one is that the time elapses for a certain period according to the temporal timer inserted as part of the JAR file; the other is that the JAR file exceeds the size stipulated by the content owner at the time of creation. After the logs are sent to the data owner, the log files will be dumped, so as to free the space for future access logs. Along with the log files, the error correcting information for those logs is also dumped. This push mode is the basic mode which can be adopted by both the PureLog and the AccessLog, regardless of whether there is a request from the data owner for the log files. 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. Concerning the latter function, we notice that the auditor, upon receiving the log file, will verify its cryptographic guarantees, by checking the records' integrity and authenticity. By construction of the records, the auditor, will be able to quickly detect forgery of entries, using the checksum added to each and every record.

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 naive 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.

3.2. Algorithms

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. In this case, if the data are not pushed out frequently enough, the log file may become very large, which may increase cost of operations like copying data . The pushing mode may be preferred by data owners who are organizations and need to keep track of the data usage consistently over time. For such data owners, receiving the logs automatically can lighten the load of the data analyzers. The maximum size at which logs are pushed out is a parameter which can be easily configured while creating the logger component. 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. A hybrid strategy can actually be implemented to benefit of the consistent information offered by pushing mode and the convenience of the pull mode. Further, as supporting both pushing and pulling modes helps protecting from some nontrivial attacks.

The log retrieval algorithm for the Push and Pull modes is outlined in Fig. 3. The algorithm presents logging and synchronization steps with the harmonizer in case of PureLog. First, the algorithm checks whether the size of the JAR has exceeded a stipulated size or the normal time between two consecutive dumps has elapsed. The size and time threshold for a dump are specified by the data owner at the time of creation of the JAR. The algorithm also checks whether the data owner has requested a dump of the log files. If none of these events has occurred, it proceeds to encrypt the record and write the error correction information to the harmonizer.

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,Acce**ss**Type,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

Fig. 3. Push and pull PureLog mode.

The communication with the harmonizer begins with a simple handshake. If no response is received, the log file records an error. The data owner is then alerted through e-mails, if the JAR is configured to send error notifications. Once the handshake is completed, the communication with the harmonizer proceeds, using a TCP/IP protocol. If any of the aforementioned events (i.e., there is 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 AcessLog is modified by adding an additional check . Precisely, the AccessLog 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.

Our auditing mechanism has two main advantages. First, it guarantees a high level of availability of the logs. Second, the use of the harmonizer minimizes the amount of

workload for human users in going through long log files sent by different copies of JAR files.

4.CONCLUSION AND FUTURE RESEARCH

We proposed innovative approaches for automatically logging any access to the data in the cloud together with an auditing mechanism. Our approach allows the data owner to not only audit his content but also enforce strong back-end protection if needed. Moreover, one of the main features of our work is that it enables the data owner to audit even those copies of its data that were made without his knowledge.

In the future, we plan to refine our approach to verify the integrity of the JRE and the authentication of JARs [13]. For example, wewill investigate whether it is possible to leverage the notion of a secure JVM [14] being developed by IBM. This research is aimed at providing software tamper resistance to Java applications. In the long term, we plan to design a comprehensive and more generic object-oriented approach to facilitate autonomous protection of traveling content. We would like to support a variety of security policies, like indexing policies for text files, usage control for executables, and generic accountability and provenance controls.

5.REFERENCESS

[1] Smitha Sundareswaran, Anna C. Squicciarini and Dan Lin, "Ensuring Distributed Accountability for Data Sharing in the Cloud,", IEEE Transaction on dependable a secure computing, VOL. 9, NO. 4, pg 556-568, 2012.
[2] S. Pearson , Y. Shen, and M. Mowbray," A privacy Manager for Cloud Computing," Proc. Int'l Conf. Cloud Computing (cloudcom), pp.90-106,2009.
[3] S. Pearson and A. Charlesworth, "Accountability as a Way Forward for Privacy Protection in the Cloud, " *Proc First Int'l conf. Cloud* Computing, 2009.
[4] 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 Workshop Formal Aspects in Security and Trust, pp. 187-201, 2005.
[5] A. Squicciarini , S. Sundareswaran and D. Lin, " Preventing Information Leakage from Indexing in the Cloud," *Proc. IEEE Int'l* Conf. Cloud Computing, 2010.
[6] B. Chun and A. C. Bavier ,"Decentralized Trust Management and Accountability in Federated System," *Proc. Ann. Hawaii Int'l Conf. System Science (HICSS), 2004.*
[7] B. Crispo and G. Ruffo, "Reasoning about Accountability within Delegation," Proc. Third Int'l Conf. Information and Comm. Security (ICICS), pp. 251-260, 2001.
[8] S. Sundareswaran, A. Squicciarini, D. Lin, and S. Huang, "Promoting Distributed Accountability in the Cloud," *Proc. IEEE* Int'l Conf. Cloud Computing, 2011.
[9] D.J. Weitzner, H. Abelson, T. Berners-Lee, J. Feigen-baum, J. Hendler, and G.J. Sussman, "Information Accountability," Comm. ACM, vol. 51, no. 6, pp. 82-87, 2008.
[10] B. Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C. John Wiley & Sons, 1993.
[11] Praveen Gauravaram, John Kelesy, Lars Knudsen, and Soren Thomsen, "On Hash function using Checksums".
[12] Ryan K L Ko, Peter Jagadpramana, Miranda Mowbray, Siani Pearson, Markus Kirchberg, Qianhui, "TrustCloud: A Framework for Accountability and Trust in Cloud Computing" HP Laboratories, pp 1 – 7, HPL-2011-38
[13] 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.
[14] Trusted Java Virtual Machine IBM, http://www.almaden.ibm. com/cs/projects/jvm/, 2012.

AUTHOR'S PROFILE:

EPURU MADHAVARAO, received the bachelor of engineering degree in Information Technology from NIET AndhraPradesh , India. Now Pursuing M.Tech in department of computer science and engineering in Vignan's Lara Institute of Technology & Science,Vadlamudi, affliated to JNTUK,Guntur,A.P,India

M PARIMALA, is currently serving as a Assistant professor (CSE) in Vignan's Lara Institute of Technology & Science, Vadlamudi, Guntur, A.P., India.

Chikkala JayaRaju, received the bachelor of engineering degree in Information Technology from QIS Engineering College AndhraPradesh , India. Now Pursuing M.Tech in department of computer science and engineering in Vignan's Lara Institute of Technology & Science,Vadlamudi,affliated to JNTUK,Guntur,A.P,India .