The International Journal Of Engineering And Science (IJES)
|| Volume || 3 || Issue || 3 || Pages || 01-06 || 2014 ||
ISSN (e): 2319 – 1813 ISSN (p): 2319 – 1805

Enhanced Distributed Accountability Framework with Indexing In the Cloud and Its Security Analysis

[1,] Shyamasree Ghosh, [2,] D.RamyaDorai. M.E (Ph.D)
[1,] *PG Scholar,* [2,] *Associate Professor*
Department of Computer Science Adhiyamaan College of Engineering

----------ABSTRACT----------
Cloud computing is the practice of using a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server or a personal computer. It provides the highly scalable services as on demand basis. As it facilitates the use of virtual servers and can therefore be moved around and scaled up (or down) on the fly without affecting the end user - arguably, rather like a cloud, the user does not know about the actual location of their data and thus user looses the control on their own data. To overcome this problem have proposed a novel highly decentralized information accountability framework with indexing mechanism. Here the accountability framework provides automatic logging and end to end auditing mechanism that enables the user to keep track of their own data along with their usage policies and the indexing policy will enable the user for the fast, easy retrieval of accurate and reduced search result set. Here we have used the JAR programmable capabilities which works on the object oriented approach. This JAR files are the key components which combines the user data and the logging information of that data. It can create the dynamic objects and ensure that any access to the user's data will trigger an automated logging and authentication mechanism associated with that data. A distributed auditing mechanism has also been used to make the user control tighter on the user's data. In this paper we have explained some security analysis and how this framework has overcome those problems.

Index Terms------*Cloud computing, accountability, data sharing.*

Date of Submission: 05 February 2014 Date of Acceptance: 15 March 2014

I. INTRODUCTION

In cloud environment a number of servers from the different locations are connected by a real time network. The servers can join or leave the cloud dynamically. Today there are a number of commercial cloud computing servers are there like Goole, Amazon, Microsoft, Yahoo, Salesforce[5] The users who are storing their data, applications on the cloud are actually storing their data on virtual servers so they do not know about the actual location of their data as all the storing, processing and handling of the data are done in the user transparent manner. This can be an advantage of cloud users that they need not to be experts on the technology infrastructure. But like a pros and cons of a coin this has the problem of loosing user's control over the user's own data. In addition to this, the data processed on the cloud can be outsourced most of the time. These lead to a number of accountability related issues in the cloud, including the handling of personally identifiable information. Not only this but also users are also worrying about a range of privacy and security issues [5][9][13]. So the loosing of user's control on their own data forces users not to adopt cloud environment [13] and thus become a significant barrier for greeting the cloud computing. Concerning the above fact, the user should be provided an effective mechanism by which they can keep track of the usage of their data. They should be ensured that their data are handled and processed according to the service level agreement. All the centralized approaches for access control are not applicable here as cloud is itself a distributed environment and high dynamicity is the main characteristic of the cloud environment (entities can dynamically leave or join in the cloud). Sometimes the data handling in cloud can be outsourced to the other entities by the cloud service providers (CSP) and these entities also can outsourced the data handling to the other entities. Thus in cloud environment data processing and handling always go through complex and dynamic service chain. For these reasons the centralized approach to provide usage and access control mechanism is not suitable here. To overcome this above problem we have proposed CIA (cloud Information Accountability) approach[1] which is based on information accountability [18]. This novel approach provides an end to end auditing mechanism in highly distributed fashion for ensuring accountability. This approach is quite similar with the approach in [7]. CIA combines the aspects of access control, usage control, authentication and auditing. For the access and usage control the data owner not only ensured about the honor of their service level agreements but also enforce their

controls as needed. It also provides automated logging mechanism which trigger the automatic generation of log records at any access to the associated data item. This CIA approach is different from the traditional privacy protection as the letter based on hide-it-or-lose-it perspective. But here the data are not hidden and they can be seen by all but they can only be accessed according to the rules and policies made by the data owner and any Violation of this will not give access to the particular data. The CIA has used the JAR (Java Archives) programmable capabilities with the help of self-defending objects (SDO)[4] which are the java based techniques so at the same time it has become a lightweight accountability mechanism. This JAR file is responsible for generating the automatic logging information on any access to the particular data item. The JAR file contains the data item along with the user enforced access or usage policies and this JAR file is sent to the CSP. Now any access to this particular data item will trigger an automated and authenticated local mechanism and stored in that JAR file locally. So the polices, logging mechanism and the associated data are strongly bounded with each other and it travels altogether as they are enclosed within one JAR file. All these information in the JAR file are in the encrypted format. For encryption we have used Identity-Based Encryption(IBE)[2]. This strong binding among the elements of JAR file exists in case of multiple copies of JAR files. So the user can have control on their data at any location. The decentralized logging mechanism also suits with the distributed nature of the cloud. To get the information of these JAR files the CIA works with two approaches: Push mode and Pull mode. In push mode the log records are periodically sent to the data owner and in Pull mode the data owner (or any authorized party) can retrieve the logs whenever they need. In addition to this we have introduced the indexing technique for easy, fast and accurate retrieval of files. However, when the number of documents to search is potentially large, or the quantity of search queries to perform is substantially large the full-text search is not suitable here. In that case the searching is based on this indexing technique and it is done before searching. The indexing stage will scan the text of all the documents and build a list of search terms (often called an index, but more correctly named a concordance). In the search stage, when performing a specific query, only the index is referenced, rather than the text of the original documents.

The features of the framework are as follows:

 The framework enables novel usage of JAR files which provides a systematic approach to data accountability. So our framework provides an automatic and enforceable logging mechanism in the cloud.
 This architecture is platform independent and highly decentralized, in that it does not require any dedicated authentication and storage system in place.
  Not only traditional access control butalso we provide a certain degree of usage control for the protected data after these are delivered to the receiver.
 The indexing technique for accurate, fast and easy retrieval of reduced result set has been introduced.

Some security problem which has been overcome by this approach has been discussed. Four types of attacks has been addressed, they are copying attack, Disassembling attack, Man-in Middle attack and Compromised JVM attack.

II. MAIN MODULES

A. CIA Framework

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 basis. It provides automated logging and end to end auditing mechanism which are the backbone of CIA. This framework has two components for this, Logger and Harmonizer.

Logger Component: Previously we have told that along with the each user data the usage and access control policies and the logging information are included. This is accomplished by this logger component. The logger is the major component of CIA and it is coupled with the user data strongly. Whenever the data is copied the associated logger is also copied and it is downloaded whenever the data is downloaded. The logger is responsible for automatically generating the logging information. Other important tasks of the logger are encrypting the log record using public key of the content owner, and periodically sending them to the log harmonizer. The advantages and features of the logger are:

 Minimal support is required from the servers in order to deploy.
 The tight coupling between data and logger suitable for highly distributed logging environment
 Strong access and usage control policy can be enclosed with it.
 Logger can handle errorsof the log records by generating error correction information.

Harmonizer: It is also a major component of CIA we can say it is the central component of CIA. It is responsible for providing permissions to access the log records enclosed in logger component. The harmonizer provides the second feature of the CIA i.e. end to end auditing mechanism
The features of harmonizer are:
 It generates the master key which is used for the decryption of the data encrypted in logger. If the path between the harmonizer and client is not secure the decryption can be carried out at client part. The key is sent by the harmonizer in a secure key exchange manner.
 It uses two auditing strategies push and pull(those will be discussed later)
 If for a same data item multiple loggers exist the harmonizer combines them and sends this merged log record to the data owner.

B. Two Distinct Mode for Auditing

To enable the CIA frame work to combine automated logging and end to end auditing mechanism we have included two distinct mode for auditing. These two modes are used to provide the data usage and logging information to the data owner in timely and accurate manner and it complements the logging mechanism though it is mode for auditing. These to modes acts as the glue between these to mechanisms (logging and auditing) to work CIA efficiently.

***Push mode*:** In push mode the log records are periodically sent to the data owner by the harmonizer as we have told earlier. This mode is the basis mode for auditing and this mode can work with both pure log and access log (discussed above). It has two features 1) It ensures that the size of the log file should not exceed from a specified size and 2) It enables detection and correction of the any damaged or lost log files periodically.

***Pull mode*:** In this mode the data owner or auditor (or any authorized party) can retrieve the logs whenever they need. When any auditor wants to see the logging information of its data they can simply request FTP pull command to the harmonizer and harmonizer will provide the integrated copy of the authenticated and sealed log file.

C. Logging and Auditing Techniques

The CIA framework provides two mechanisms to provide accountability in cloud. The CIA demands the logging and auditing techniques to satisfy the following requirements.

1. To support the dynamic nature of the cloud the logging mechanism should be decentralized.
2. A strong binding should exist between the log files and the corresponding data being controlled, and require minimal infrastructural support from any server.
3. The logging and auditing mechanism should provide the automatic and accurate logging information with each and every access to the corresponding data. It not only requires the proper a mechanism to authenticate and verify an entity but also the mechanism to record any access to the data with its time of access, should be coupled together.
4. The error correction and detection mechanism for the log files should be there. The log file should be temper proof and reliable so that any unauthorized insertion, modification and deletion cannot be done. It should properly handle the problems of log files caused by some technical problems.
5. Log files should be sent back to their data owners periodically to inform them of the current usage of their data. More importantly, log files should be retrievable by their data owners irrespective of time and the location where the files are stored.
6. The technique should not intrusively monitor data recipient's systems and it should introduce light communication and computation overhead, which otherwise will hinder its feasibility and adoption in practice.

Now we will discuss the logging and auditing mechanism one by one.

Automated Logging Mechanism

As we have told earlier this logging mechanism is carried out by logger component of CIA. At first we will see the logger structure. The Structure of the logger as follows:

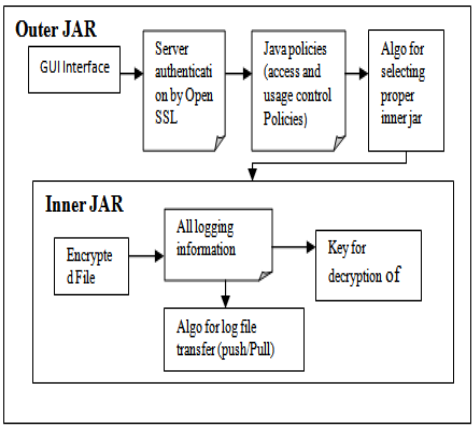

Fig 2.1: Logger structure

As shown in the above fig we can see the logger is implemented with the help of JAR's programmable capabilities. Logger component is nothing but the nested Java JAR file which encapsulates logging information (in the form of log file) along with the associated data item. The logger contains two JAR files: inner JAR and Outer JAR. One outer JAR can contain one or more inner JARs. The task of the outer JAR is to handle the authentication of the entities which want to access the particular data item stored in the JAR file. Outer JAR is also responsible for the access and usage control policies based on owner's requirement. It checks the policies while giving permissions for accessing the data item. All the access and usage control policies are encapsulate here. Now the inner JAR contains the actual data in encrypted format, the log records for each encrypted data item and the class file to retrieve the of log records. After getting authenticated by the outer JAR the entity can select the proper inner JAR which is meant for his intended data item by the help of the algorithm for selecting the proper JAR.

End To End Auditing Mechanism

To provide the auditing mechanism the logger and harmonizer work together along with the two modes (push and pull). With the help of these two methods the auditing is done. We can see in some situation the push mode is beneficial and at sometime pull mode. We need a proper combination of push and pull mode. Here we used the following algorithm for retrieving the log records.

2.4 Indexing Module

Indexing module is responsible for easy and accurate retrieval of files. The indexer will filter the search queries and will provide more accurate set of results. Full-text search which is the traditional search policy refers to techniques for searching a single computer stored document or a collection in a full-text database. In a full-text search, the search engine examines all of the words in every stored document as it tries to match search criteria. When dealing with a small number of documents, it is possible for the full text search engine to directly scan the contents of the documents with each query, a strategy called "serial scanning .But when the number of documents to search is potentially large, or the quantity of search queries to perform is substantial high this traditional search technique will not work. This leads to more time consuming process and provides unrelated items in the result set those may not at all beneficial to the user. So we have proposed indexing technique here. Indexing is basically two step process namely indexing and searching, where indexing has to be done before actual searching for data item. The indexing stage will scan the text of all the documents and build a list of search terms (often called an index, but more correctly named a concordance). In the search stage, when performing a specific query, only the index is referenced, rather than the text of the original documents. The indexer will make an entry in the index for each term or word found in a document, and possibly note its relative position within the document. Usually the indexer will ignore stop words (such as "the" and "and") that are both common and insufficiently meaningful to be useful in searching. Some indexers also employ language-specific stemming on the words being indexed. We have implemented indexing at the server side and it is performed after the authentication of the user has made. As filtering has performed before searching so it will give more accurate search result set which is free of unnecessary stuffs.

III. DISCUSSIONS ON SECURITY

Here we are going to analyze the security issues and how the framework provides security to these attacks. But before that we need to take some assumptions.

1.The first and most important condition is JVM should not be corrupted
2.The master key should not be released by the user to any unauthorized party when attacker can try to get information from the log file.
3. It can possible that the attacker has enough java programming skills to disassemble the JAR file and he has the prior knowledge about the framework.

A.copying attack

The attacker can copy the entire JAR file by accessing the data item encapsulated within the JAR file without being noticed by the data owner. The auditing mechanism prevents this attack. As the harmonizer will send the periodic logging information to the data owner (in push method) after getting it from the JAR file periodically so this attack can be prevented. If any copy has made without the knowledge of data owner still he can get the information about that copy. If attacker make the copy in the place where the harmonizer will not connect those JAR copies will become inaccessible. The logger component enables user to detect the owner when any unauthorized copy has made and make inaccessible to all offline files by providing conventional log file encryption and transparency.

B.Disassembling of JAR files

We know that the logger component contains the JAR files. The attacker can dissemble the JAR files to get extract information (like the public IBE key used for encryption, the encrypted log file and the .class file) or destroy the log records in it. This is the most dangerous attack in this framework. We can handle this attack by providing the strong encryption mechanism. Here the Weil Pairing encryption algorithm has been used which ensured that this framework provides both plaintext security and cipher text security. Though the attacker gets the information but cannot decrypt any information. The log files cannot be modified by the attacker as the integrity check mechanism has been applied to each log record and at the time of verification it will be detected. Writing the fake records can also be prevented by the application of Reed Solomon encoding. Finally the attacker can try to modify the java class loader in the JARs to subvert the class file at the time of loading. This has been prevented by using the sealing techniques offered by java.

C. Man –in –the-middle attack

In this attack the attacker masquerade itself as the Service provider at time of authentication of the service provider with the certificate authority**.** In two situations this attack can be possible. First when the session between the service provider and the certificate authority has ended and they have disconnected their connection. But in this condition the attack will not be succeed as each certificate has timestamp associated with it. So if the attacker wants to access the certificate after the disconnection the certificate becomes useless at that point of time. Secondly when the Service provider has disconnected but the session has not got over. Here also the attack not possible as the attacker cannot renegotiate as the renegotiation is prohibited in the current version of open SSL and cryptographic checking is also included.

D. Attack by compromising the JVM

This attack has been prevented by using oblivious hashing technique [3]. It guarantees the correctness of JRE.

IV. CONCLUSION

We proposed innovative approaches for automatically logging any access to the data in the cloud together with an auditing mechanism. User can keep track of any access to their data and be ensured with the assurance of maintaining the proper service level agreements. Our approach allows the data owner to not only audit its content but also enforce strong back end protection if need. Moreover one main feature of our work is that it enables the data owner to audit even those copies of its data that are made without its knowledge. Here we are providing support of indexing policy for more improved search result set. Further we have discussed about the security issues which has been considered and how does framework prevent those attacks.

V. AKNOWLEDGEMENT

I express my humble thanks to the Almighty for having showered his blessings and divine light to raise me to the height of presenting this beneficial paper successfully and continuing my project work. I extend sincere gratitude to my guide **Prof. D.RAMYA DORAI M.E., (Ph.D.).,** Associate Professor, Department of Computer Science & Engineering, Adhiyamaan College of Engineering, Hosur, for her valuable directions, suggestions and exquisite guidance with ever enthusiastic encouragement ever since the commencement of the project and my college **ADHIYAMAAN COLLEGE OF ENGINEERING, HOSUR** to provide this opportunity to carry out my work.

REFERENCES

[1] Smitha Sundareswaran, Anna C. Squicciarini, and Dan Lin "Ensuring Distributed Accountability for data Sharing in the Cloud", IEEE Trans. On Dependable and Secure Somputing. Vol 9 pp 555-567, 2012.
[2] D. Boneh and M.K. Franklin, "Identity-Based Encryption from the Weil Pairing", Proc Int‟l Cryptography Conf. Advances in Cryptography, pp. 213-229, 2001.
[3] Y.Chen et al., "Oblivious Hashing: A Stealthly Software Integrity Verification Primitive,"Proc Int‟l Workshop Information Hiding,F.Petitcolas, ed.,pp.400-414,2003.
[4] J.W. Holford, W.J. Caelli, and A.W. Rhodes, "Using Self- Defending Objects to Develop Security Aware Applications in Java," Proc. 27th Australasian Conf. Computer Science, vol. 26, pp.341-349, 2004.
[5] P.T. Jaeger, J. Lin, and J.M. Grimes, "Cloud Computing and Information Policy: Computing in a Policy Cloud?," J. Information Technology and Politics, vol. 5, no. 3, pp. 269-283, 2009.
[6] 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.
[7] W. Lee, A. Cinzia Squicciarini, and E. Bertino, "The Design and Evaluation of Accountable Grid Computing System," Proc. 29th IEEE Int‟l Conf. Distributed Computing Systems (ICDCS ‟09), pp.145-154, 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.
[9] 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.
[10] M.C. Mont, S. Pearson, and P. Bramhall, "Towards Accountable Management of Identity and Privacy: Sticky Policies and Enforce- able Tracing Services," Proc. Int‟l Workshop Database and Expert Systems Applications(DEXA), pp. 377-382, 2003.
[11] S. Oaks, Java Security. O‟Really, 2001.
[12] J. Park and R. Sandhu, "Towards Usage Control Models: Beyond Traditional Access Control," SACMAT „02: Proc. Seventh ACM Symp. Access Control Models and Technologies, pp. 57-64, 2002.
[13] S. Pearson and A. Charlesworth, "Accountability as a Way Forward for Privacy Protection in the Cloud," Proc. First Int‟l Conf. Cloud Computing, 2009.
[14] S. Pearson, Y. Shen, and M. Mowbray, "A Privacy Manager for Cloud Computing," Proc. Int‟l Conf. Cloud Computing (CloudCom),pp. 90- 106, 2009.
[15] A. Pretschner, M. Hilty, and D. Basin, "Distributed Usage Control," Comm. ACM, vol. 49, no. 9, pp. 39-44, Sept. 2006.
[16] NTP: The Network Time Protocol, http://www.ntp.org/, 2012.
[16] S. Sundareswaran, A. Squicciarini, D. Lin, and S. Huang, "Promoting Distributed Accountability in the Cloud," Proc. IEEE Int‟l Conf. Cloud Computing, 2011.
[17] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing," Proc. European Conf. Research in Computer Security (ESORICS), pp. 355-370, 2009.
[18] 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.
[19] Reed-Solomon Codes and Their Applications, S.B. Wicker and V.K. Bhargava, ed. John Wiley & Sons, 1999.
[20] Reed-Solomon Codes and Their Applications, S.B. Wicker and V.K. Bhargava, ed. John Wiley & Sons, 1999.

Shyamasree ghosh received the bachelor's degree in computer science and engineering in 2009 from West Bengal University of technology, West Bengal, India. She is currently persuing M.E degree from Adhiyamaan College of Engineering (Autonomous), Hosur.

Prof. D. RamyaDorai received her M.E degree in computer science and engineering in 2006 from Sona College of Tech, Salem, Anna University. She is an assistant professor at Adhiyamaan College of Engineering (Autonomous), Hosur and pursuing her Ph.D degree from Anna University, Chennai.