ISSN(Online):2320-9801
ISSN(Print): 2320- 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Special Issue 3, July 2014

Enhancing Authentication for Out Sourced Data in the Cloud

T.N.Sandya[1], J.Shanthini[2]

P.G.Student, Department of CSE, Info Institute of Engineering, Coimbatore, India

Assistant Professor, Department of CSE, Info Institute of Engineering, Coimbatore, India.

ABSTRACT: Cloud computing is sharing of resources and its main advantages are SaaS, IaaS, PaaSas done with the providence of internet. Cloud computing has the feature of processing the data which is remotely placed. The people concern about three security problems in cloud. One is loss of trust, then multi tendency and finally loss of control. To address the loss of control problem, in this paper, I propose a decentralized Cloud Information Accountability(CLA) framework to keep track of the actual usage of the user's data in the cloud associated with the introduction of the LOG file once the authentication is triggered. I provide a session key for the data download. I provide extensive experimental studies that demonstrate the efficiency and effectiveness of the proposed approaches.

KEYWORDS: Accountability, Cloud computing, Data sharing.

I.INTRODUCTION

Cloud computing is highly based on a traditional technology: grid computing, gave a way for cloud computing. Cloud computing mainly focus on sharing of information and computation of task over a large network, which are quite likely to be owned by different companies. It is proved to believe that cloud computing has been one of the sources for success in several major companies such as Google and Amazon. Cloud computing takes the user to the next generation in which clients to be as smart phones which stores their entire data in the cloud and request various tasks to be performed. A browser acts as an interface between clients and the cloud. The data processed on clouds are outsourced, that leads to many issue such as accountability and loss of control over data. Such problem becomes a barrier to use cloud. To avoid this problem owner should be able to track the user's usage in the cloud.

There are three security issues in the cloud: loss of control, multi tendency and loss of trust. My project deals with the loss of control. The administrator plays a major role who takes care of the registration of user and the data owner. The data owner uploads the file and the user downloads the file. Tracking of misbehaved user is provided the security enhancement.
In summary, my main contribution is as follows:
My main contributions are as follows:

- I propose a novel automatic and enforces logging mechanism in the cloud. To my knowledge, this is the first time a systematic approach to data accountability through the novel usage of LOG files is proposed.
- My proposed architecture is platform independent and highly decentralized, in that it does not require any dedicated authentication or storage system in place.
- I go beyond traditional access control in that I provide a certain degree of usage control for the protected data after these are delivered to the receiver.

II.RELATED WORK

In this section, I first review related works addressing the privacy and security issues in the cloud. Then, I briefly discuss works which adopt similar techniques as my approach but serve for different purposes.

2.1 Cloud Privacy and Security

Cloud computing has raised a range of important privacy and security issues.

ISSN(Online):2320-9801
ISSN(Print): 2320- 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Special Issue 3, July 2014

G. Ateniese et al [1]. describes retrieving their file from the server. The client is stored in the server. If the client wants to retrieve the file it is checked over the server. Pre-processing, is done and the client may be stored at the server side. The client expands the file or adds the data which is stored in the server . Before deleting a local copy of file it checks whether it is stored in the server. Client encrypts the outsourced the data.

P. Buneman et al [3]. Focus on recording the provenance of data that is copied in databases. They describes an approach to track the user's usage while browsing the database. This proves the flexibility of database for provenance management.

I do not aim at controlling the information workflow in the clouds. In a summary, all these works stay at a theoretical level and do not include any algorithm for tasks like mandatory logging.

2.2 Other Related Techniques

With respect to LOG-based techniques for security, my methods are related to decentralised approach . It is an extension of the object-oriented programming paradigm, where software objects that offer sensitive functions or hold sensitive data are responsible for protecting those functions/data. Similarly, I also extend the concepts of object-oriented programming. The key difference in my 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. In my work, a Log-based approach to prevent leakage from indexing , which could be integrated with the CIA framework. I aim at providing a session key while downloading the files from the cloud which is sent by the data owner at the time of uploading the file.

III. PROBLEM STATEMENT

The existing system to use passwords that can be easily guessed or hacked and the other technique is biometrics, but these two techniques has its own disadvantages. Biometrics, such as finger prints, iris scan or facial recognition have been introduced but not yet widely adopted. A user needs to recognize pass-objects and click inside the convex hull formed by all the pass-objects. In order to make the password hard to guess large number of objects can be used but it will make the display very crowded and the objects almost indistinguishable, but using fewer objects may lead to a smaller password space, since the resulting convex hull can be large. In the authentication stage, the user must type the unique codes of the pass objects variants in the scenes provided by the system.

3.1 Major Components of CIA

There are two major components of the CIA, the first one is logger which is concerned with user's data, and the second being the administrator, which takes care of the data owner uploading and the user's downloading. Once the data owner uploads the file the user name and password is provided. IF the data owner wants to upload the file he/she uses the user name and password. The user enter the cloud for downloading it follows the same procedure as the data owner and it is provided by the session password for downloading purpose. The log file is created that stores the log record of both data owner and the user.

3.2 Data Flow

Using the generated session key, the user will download the file which is stored in the cloud. The LOG file includes a set of simple access control rules specifying whether and how the cloud servers, and possibly other data owners are authorized to access the content itself. Subscribes to logging functionality.

ISSN(Online):2320-9801
ISSN(Print): 2320- 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Special Issue 3, July 2014

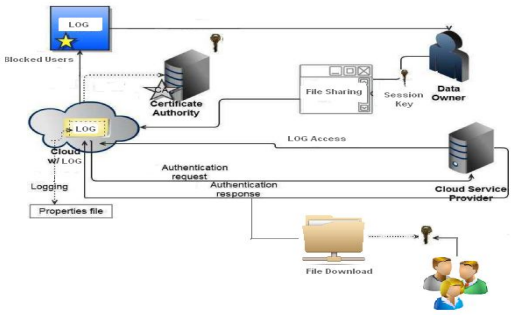

Fig. 1. Overview of Cloud Information Accountability

IV. AUTOMATED LOGGING MECHANISM

In this section, I first elaborate on the automated logging mechanism and then present techniques to guarantee dependability.

4.1 The Logger Structure

I leverage the programmable capability of LOGs to conduct automated logging. A file which stores the user's data items and corresponding log files. The misbehave of the user will be stored in the log file and it will be sent to the owners E-mail.

- AccessLog. It has two functions: logging actions and enforcing access control. In case an access request is denied, the LOG will record the time when the request is made. If the access request is granted, the LOG will additionally record the access information along with the duration for which the access is allowed. I propose a specific method to correctly record or enforce it depending on the type of the logging module, which are elaborated as follows:
- Download. The entity is allowed to save a raw copy of the data and the entity will have control over this copy by log records regarding access to the copy. When an entity clicks this download link, the LOG file associated with the data will record the entity. The user can just have a single click to download the file.
- Timed_access. There is no timed access for entering the session key. This provides the user friendly of access with security.

4.2 Dependability of Logs

In this section, I discuss how I ensure the dependability of logs. In particular, I aim to prevent the following two types of attacks. First, an attacker may try to hack the user name and password remotely. Second, the attacker may try to compromise the JRE used to run the LOG files.

4.2.1 LOGs Availability

The log can be created in any number to manage the logs of the user and the data owner. The availability of the log file is taken care by the admin.

V. ALGORITHMS

5.1 Push and Pull Mode

The push and pull mode acts together by the admin. The push mode describes the functionality of automatically sending the log records to the data owner's E-mail whereas, the pull mode automatically receives the log records of user. The mode deals with the receive of misbehaved user.

ISSN(Online):2320-9801
ISSN(Print): 2320- 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Special Issue 3, July 2014

5.2 Diffie Hellman Algorithms

The Diffie–Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher. The key can then be used by both parties to encrypt and decrypt data. This algorithm is limited to the exchange of the keys. No KDC or central authentication server is required. Lets assume two parties A and B want to establish a shared secret. Assume there are two publicly known numbers: a prime number q and an integer α, which is a primitive root of q. User A selects a random integer XA < q and computes $YA = \alpha^{XA} \bmod q$. Similarly, user B independently selects a random integer XB < q and computes $YB = \alpha^{XB} \bmod q$. Each side keeps the X value private and makes the Y value available publicly to the other side. User A computes the key as $K = (YB)^{XA} \bmod q$ and user B computer the key as $K=(YA)^{XB} \bmod q$. These two calculations produce identical results:

$$
\begin{aligned}
K &= (YB)^{XA} \bmod q \\
&= (\alpha^{XB} \bmod q)^{XA} \bmod q \\
&= (\alpha^{XB})^{XA} \bmod q \\
&= (\alpha^{XA})^{XB} \bmod q \\
&= (\alpha^{XA} \bmod q)^{XB} \bmod q \\
&= (YA)^{XB} \bmod q
\end{aligned}
$$

Thus, the two sides have exchanged a secret key. Note at the start of each session, both A and B can come up with a new value for XA and XB.

VI. SECURITY DISCUSSIONS

I now analyze possible attacks to my framework. My analysis is based on a semihonest adversary model by assuming that a data owner does not release his/her session key to unauthorized parties, while the attacker may try to learn extra information from the log files. I assume that attackers may have sufficient hacking skills to disassemble a LOG file and prior knowledge of my CIA architecture. I first assume that the JVM is not corrupted, followed by a discussion on how to ensure that this assumption holds true.

6.1 Copying Attack

The most intuitive attack is that the attacker copies entire LOG files. The attacker may assume that doing so allows accessing the data in the LOG file without being noticed by the data owner. However, such attack will be detected by my auditing mechanism. Recall that every LOG file is required to send log records to the administrator. In particular, with the push mode, the administrator will send the logs to data owners periodically. That is, even if the data owner is not aware of the existence of the additional copies of its LOG files, he will still be able to receive log files from all existing copies. If attackers move copies of LOGs to places where the administrator cannot connect, the copies of LOGs will soon become inaccessible.

6.2 Disassembling Attack

Another possible attack is to disassemble the LOG 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 LOG files, this attack poses one of the most serious threats to my architecture. Since I cannot prevent an attacker to gain possession of the LOGs, I rely on the strength of the cryptographic schemes applied to preserve the integrity and confidentiality of the logs. The class loader is found to be a custom classloader, the LOGs will throw an exception and halt.

VII. PERFORMANCE STUDY

In this section, I first introduce the settings of the test environment and then present the performance study of my system.

7.1 Experimental Settings

I tested my CIA framework by setting up a small cloud, using the Emulab testbed.

ISSN(Online):2320-9801
ISSN(Print): 2320- 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Special Issue 3, July 2014

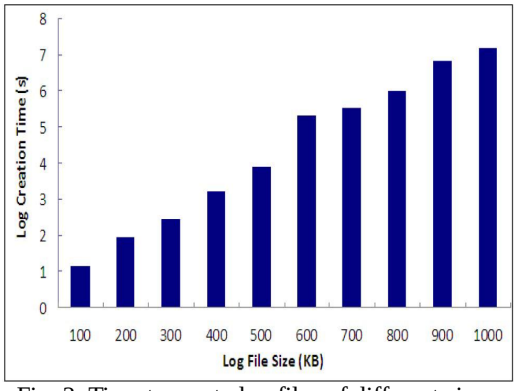

Fig. 2. Time to create log files of different sizes.

one head node which is the certificate authority, and several computing nodes. Each of the servers is installed with Eucalyptus. Eucalyptus is an open cloud implementation for both Linux-based and Windows based systems. It is loosely based on Amazon EC2, therefore bringing the powerful functionalities of Amazon EC2 into the open domain. I used Windows-based servers running Windows 7 OS. Each server has a 64-bit Intel Quad Core Xeon E5530 processor, 4 GB RAM, and a 500 GB Hard Drive.

7.2 Experimental Results

In the experiments, I first examine the time taken to create a log file and then measure the overhead in the system. With respect to time, the overhead can occur at only one points: during the authentication. Also, with respect to storage overhead, I notice that my architecture is very lightweight, in that the only data to be stored are given by the actual files and the logs.

7.2.1 Log Creation Time

In the first round of experiments, I are interested in finding out the time taken to create a log file when there are entities continuously accessing the data, causing continuous logging. Results are shown in Fig. 2. It is not surprising to see that the time to create a log file increases linearly with the size of the log file. Specifically, the time to create a 100 Kb file is about 114.5 ms while the time to create a 1 MB file averages at 731 ms. With this experiment as the baseline, one can decide the amount of time to be specified between dumps, keeping other variables like space constraints or network traffic in mind.

7.2.2 Authentication Time

The next point that the overhead can occur is during the authentication of a CSP. If the time taken for this authentication is too long, it may become a bottleneck for accessing the enclosed data. To evaluate this, the head node issued OpenSSL certificates for the computing nodes and I measured the total time for the OpenSSL authentication to be completed and the certificate revocation to be checked. Considering one access at the time, I find that the authentication time averages around 920 ms which proves that not too much overhead is added during this phase. As of present, the authentication takes place each time the CSP needs to access the data. The performance can be further improved by caching the certificates. The time for authenticating an end user is about the same when I consider only the actions required by the LOG, viz. obtaining a SAML certificate and then evaluating it. 78.2.3 Time Taken to Perform Logging

This set of experiments studies the effect of log file size on the logging performance. I measure the average time taken to grant an access plus the time to write the corresponding log record. The time for granting any access to the data items in a LOG file includes the time to evaluate and enforce the applicable policies and to locate the requested data items. In the experiment, I let multiple servers continuously access the same data LOG file for a minute and recorded the number of log records generated. Each access is just a view request and hence the time for executing the action is negligible. As a result, the average time to log an action is about 10 seconds, which includes the time taken by

ISSN(Online):2320-9801
ISSN(Print): 2320- 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Special Issue 3, July 2014

a user to double click the LOG or by a server to run the script to open the LOG. I also measured the log encryption time which is about 300 ms (per record) and is seemingly unrelated from the log size.

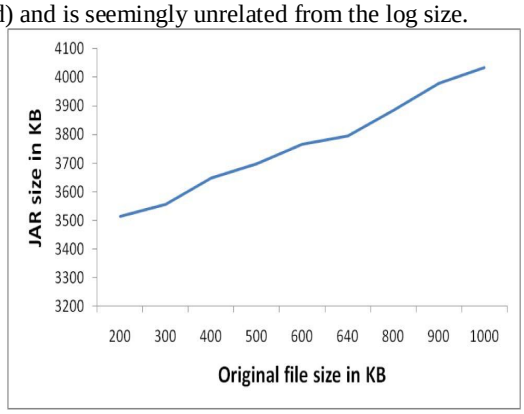

Fig. 3. Size of the logger component.

8.2.5 Size of the Data LOG Files

Finally, I investigate whether a single logger, used to handle more than one file, results in storage overhead. I measure the size of the loggers (LOGs) by varying the number and size of data items held by them. I tested the increase in size of the logger containing 10 content files (i.e.,images) of the same size as the file size increases. Intuitively, in case of larger size of data items held by a logger, the overall logger also increases in size. The size of logger grows from 4,500 to 5,035 KB when the size of content items changes from 200 KB to 1 MB. Overall, due to the compression provided by LOG files, the size of the logger is dictated by the size of the largest files it contains. Notice that I purposely did not include large log files (less than 5 KB), so as to focus on the overhead added by having multiple content files in a single LOG.

VIII.CONCLUSION

I propose an innovative approach for automatic log-in for any access to the data in the cloud. This approach allows the data owner enforcing a strong back-end protection if needed. Moreover, one of the main features of my work is that it enables the data owner to have their files safe in the cloud. Also, with respect to storage overhead, I notice that my architecture is very lightweight, in that the only data to be stored are given by the actual files and the logs. Further, LOG acts as an important feature in my project due to its secure log record providance. In particular, multiple files of the same data owner can also be handled by the same logger component. To this extent, I investigate whether a single logger component, used to handle more than one file, results in storage overhead.

REFERENCES

[1] 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.
[2] R. Bose and J. Frew, "Lineage Retrieval for Scientific Data Processing: A Survey," ACM Computing Surveys, vol. 37, pp. 1-28, Mar. 2005.
[3] P. Buneman, A. Chapman, and J. Cheney, "Provenance Management in Curated Databases," Proc. ACM SIGMOD Int'l Conf.Management of Data (SIGMOD '06), pp. 539-550, 2006.
[4] 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] 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] HUH, J.H., AND MARTIN, "A.Trusted Logging for Grid Computing." In Trusted Infrastructure Technologies Conference, 2008-APTC'08. Third Asia-Pacific(Wuhan, China, oct-2008), IEEE. PP. 30-42.
[9] 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.
[10] 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.
[11] LIPNER," S. The Trustworthy Computing Security Development Lifecycle." In ACSAC '04: Proceedings of the 20th Annual Computer Security Applications Conference (Washington, DC, USA, 2004), IEEE Computer Society, pp. 2–1.

ISSN(Online):2320-9801
ISSN(Print): 2320- 9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Special Issue 3, July 2014

[12] 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.
[13] M.C. Mont, S. Pearson, and P. Bramhall, "Towards Accountable Management of Identity and Privacy: Sticky Policies and Enforceable Tracing Services," Proc. Int'l Workshop Database and Expert Systems Applications (DEXA), pp. 377-382, 2003.
[14] 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] A. Pretschner, M. Hilty, and D. Basin, "Distributed Usage Control," Comm. ACM, vol. 49, no. 9, pp. 39-44, Sept. 2006.
[6] S. Sundareswaran, A. Squicciarini, D. Lin, and S. Huang, "Promoting Distributed Accountability in the Cloud," Proc. IEEE Int'l Conf. Cloud Computing, 2011.
[15] 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.
[16] 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.
[17] M. Xu, X. Jiang, R. Sandhu, and X. Zhang, "Towards a VMMBased Usage Control Framework for OS Kernel Integrity Protection," SACMAT '07: Proc. 12th ACM Symp. Access Control Models and Technologies, pp. 71-80, 2007.