International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Pp-110-115 ISSN: 2249-6645

A Framework to Ensure Monitoring for Data Sharing with Trust in the Cloud

M. Pavithraa[#1], B. Muruganantham [#2]
[#1]M.Tech Student, Department of Computer science and Engineering, SRM University, Kattankulathur, Chennai.
[#2]Assistant Professor, (Sr.G), Department of computer Science and Engineering, SRM University, Kattankulathur, Chennai

ABSTRACT: *Users on the cloud fears of losing control of their own data. This leads to the lack of trust in clouds by dormant customers. While preventive controls for security and privacy measures are earnestly being researched, there is still diminutive focus on detective controls related to cloud accountability and auditability. This paper discusses key challenges in achieving a trusted cloud through the use of access control policy rules, and presents a novel approach namely Cloud Information Accountability (CIA), which addresses consequential challenges, including exclusively identifying CSPs, protecting the reliability of the logs, conditioning to a highly decentralized infrastructure. We influence the JAR programmable capabilities to both create a dynamic and travelling object, and to ensure that any access to the users' data will trigger authentication and automated logging confined to the JARs.*

Index Terms : *users, cloud, lack of trust, accountability, auditability, JAR.*

I. INTRODUCTION

Cloud computing is the use of computing resources that are delivered as a service over a network (Internet). In other words Cloud computing is an umbrella term used to refer to Internet based development and services. In addition, the platform provides on demand services that are always on, anywhere, anytime and anyplace. A special way to think of cloud computing is to acknowledge users experience with email. Users email client, whether it is Gmail, Yahoo, Hotmail, and so on, takes care of housing all of the hardware and software needful to hold up users personal email account. When users want to access their email they open their web browser, go to the email client, and log in. The most substantial part of the arrangement is having internet access. Users email is not housed on their physical computer; they access it through an internet connection, and can be accessed anywhere. If a user is on a trip, at work, or down the street getting coffee, they can check their email as long as they have access to the internet.

1.1 Classification of clouds

There are varied forms of clouds that user can subscribe to depending on their needs. For a home user or small business owner, users will most probably use public cloud services.

i. Public Cloud - A public cloud can be accessed by several subscribers with an internet link and access to the cloud space. These services are free of cost or provided on a pay-per-use model.
ii. Private Cloud - A private cloud is achieved for a definite group or organization and restricts access entirely to that group.
iii. Community Cloud - A community cloud is shared between two or more organizations that have alike cloud requisites.
iv. Hybrid Cloud - A hybrid cloud is basically a mixture of at least two clouds, where the clouds contained are a mix of private, public or community.

1.2 Choosing a cloud provider

Each provider serves a distinct task, giving users more or less control by their cloud susceptible on the type. When we select a provider, we analyze our needs to the cloud services applicable.
Cloud needs will vary relying on how we attempt to avail the space and resources allied with the cloud. If it is for personal home use, then we need a different cloud type and provider than if we using the cloud for business.

There are three types of cloud providers that a user can subscribe to:

i. Software as a Service - A SaaS provider provides subscribers access to both resources and applications. With the use of SaaS users need not have a physical copy of software to be installed on their devices. SaaS also makes it easier to have the identical software on all user devices at once by accessing it on the cloud. In a SaaS agreement, users have the least control over the cloud.
ii. Platform as a Service - A PaaS system goes a position beyond the Software as a Service system. A PaaS provider provides subscriber's access to the units that they need to develop and operate functions over the internet.
iii. Infrastructure as a Service - An IaaS agreement, deals fundamentally with computational infrastructure. In an IaaS agreement, the subscriber entirely outsources the storage and resources (hardware and software) that they require.

II. EXISTING SYSTEM

The service in the cloud has become very flexible over internet to afford effortless access to the user. A user can gain a little or a big part of the services which is managed by the service provider. Details of the services provided are withdrawn from the users who need not to be proficient of technology infrastructure. Likewise, users may not know the machines which indeed process and host their data. While enjoying the advancement brought by this latest technology, users also start bothering about losing control of their own data. The data handled on clouds are often outsourced, foremost to a number of concerns related to accountability, in addition to the manipulation of personally identifiable information [12]. So it is essential to provide an effective mechanism for users to monitor the usage of their data in the cloud. Further, their solution requires third-party services to complete the monitoring.

III. PROPOSED SYSTEM

To address the problems in the existing system, we propose a novel highly decentralized information accountability framework to keep track of the actual usage of the users' data in the cloud namely Cloud Information Accountability (CIA), based on the notion of information accountability[16]. Our proposed CIA framework provides end-to-end accountability in an exceedingly distributed pattern. One of the main innovatory features of the CIA framework lies in its strength of sustaining lightweight and powerful accountability that integrates aspects of access control, usage control and authentication. By ways of the CIA, data owners can track not only whether or not the access policies are being honored, but also reinforce access and usage control rules as required.

Associated with the accountability feature, two distinct modes for auditing have also been developed: push mode and pull mode. The push mode quotes to logs which are being periodically sent to the data owner or stakeholder while the pull mode quotes to a flipside approach whereby the user (or another authorized party) can get back the logs as required. For the proposed design on the CIA framework, we influence and enhance the programmable potency of JAR (Java ARchives) files to automatically log the usage of the users' data by any entity in the cloud. Users will send their data adjacent with any policies such as access control policy rules and logging policies that they want to reinforce, enfolded in JAR files, to the CSPs. Any access made to the data will create trigger an automated and authenticated logging mechanism confined to the JARs. We specify this type of reinforcement as "strong binding" since the policies and the logging mechanism move with the data. This strong binding prevails even when copies of the JARs are created; thus, the user will have control over the data at each and every location. Such decentralized logging technique meets the dynamic nature of the cloud but also charges challenges on protecting the integrity of the logging. To endure with this issue, we grant the JARs with a central point of connection which forms a bond between them and the user. It registers the error correction information issued by the JARs, which allows it to monitor the loss of any logs from any of the JARs. Furthermore, if a JAR is not able to contact its central point, any access to its enfolded data will be dropped.

In short, our main improvements are as follows:
- First time a systematized approach for data accountability through the novel usage of JAR files.
- It does not need any loyal authentication or storage system.
- Highly decentralized and Platform independent.
- The granularity, scalability and efficiency of our approach.

IV. PROBLEM STATEMENT

We begin this segment by considering an explanatory example which aids as the basis of our problem statement to validate the main features of our system.

Example 1. For an assumption, Mitrajith a professional photographer , plans to sell his photographs by using the Google Cloud Services. For his business in the cloud, he has the following specifications.
- His photographs are accessed only by the users who have paid for his services.
- Dormant buyers are allowed to only read his pictures who have been made the payment for obtaining 6 months membership.
- Due to the nature of some of his works only users who had made the payment for obtaining 37 months membership can read and comment.
- For some of his works users are allowed to read, write & execute who had made the payment for obtaining premium membership
- In case any dispute arises with a client he wants to have all the access information of that client.
- He wants to ensure that the CSP of Google do not share his data with other service providers, so that the accountability furnished for individual users can also be expected from the CSP

With the above outline in mind, we analyze the common requirements and develop several instructions to achieve data accountability in the cloud. A user, who enrolled to a certain cloud service, regularly needs to send his/her data as well as allied access control policies (if any) to the service provider. After the data are gotten by the cloud service provider, the service provider will be having the granted access rights, such as read, write, and execute, on the data. Using conventional access control systems, once the access rights are granted, the data will be fully accessable at the service provider. In order to trace the actual usage of the data, we aim to flourish novel logging and auditing techniques which satisfy the following specifications:

i. The logging should be decentralized in order to reconcile to the dynamic nature of the cloud. More specifically, log files should be firmly bounded with the equivalent data being controlled, and needs minimal infrastructural support from any server.

ii. Every access to the user's data should be correctly and automatically logged. This requires integrated mechanisms to authenticate the entity who accesses the data, verify, and record the absolute operations on the data as well as the time that the data have been accessed.
iii. Log files should be candid and tamper proof to avoid illegal insertion, deletion, and modification by deleterious parties. Rescue mechanisms are also desirable to restore damaged log files caused by technical problems.
iv. Log files should be issue back to their data owners periodically to inform them of the current usage of their data. More importantly, log files should be fetchable anytime by their data owners when required regardless the location where the files are stored.
v. The proposed mechanism should not intrusively monitor data recipients' systems, nor it should introduce excessive communication and counting aloft, which otherwise will hinder its feasibility and adoption in practice.

V. SYSTEM MODEL

In this segment, we present an outline of the Cloud Information Accountability framework and confer how the CIA framework meets the design specifications discussed in the previous section.

5.1 Data Flow

The overall CIA framework is sketched in Fig. 1. At the beginning, the data owner compresses their file to the JAR generation. The JAR file includes a set of a simple access control rules specifying whether and how the other stake holders are authorized to access the content itself. The JAR files also contains the user data item and the corresponding log files which is mainly responsible for authentication of entities which want to access the data stored in the file. Then the data owner sends the JAR file to the CSP that the data owner subscribes to. After the data's are received by CSP, the service provider will have granted access rights such as read, write and execute on the data.

On the need of data sharing for the user, the user registers based on the access control policy rules. Once the authentication succeeds, the user will be allowed to access the data enclosed in the JAR. In order to monitor the actual usage of the data in the cloud we use logging and auditing techniques. Any access to the data will trigger the logger component where access to the data can be found out and new log entries are appended sequentially. The generated log record is encrypted and sent to the data owner periodically. The encryption of the log file prevents unauthorized changes to the file by attackers.

In accession some error correction information will be shipped to the log harmonizer to handle credible log file corruption. Further individual records are hashed together to generate a chain structure to able to fastly detect possible errors or missing records. The encrypted log files can subsequently be decrypted and their integrity confirmed. They can be accessed by the data owner or other stakeholders at any moment for auditing scope with the help of the log harmonizer.

5.2 Proof of Concept

In order to demonstrate the novel approach CIA introduced in the proposed system. We implemented a proof of concept purely in software.

5.2.1 User Interface Design

In this phase we create a GUI page, which acts as a median to connect the user with the cloud and through which user can able to give request to the cloud, by mean time cloud server can send the response to the user. In other words, this phase establishes the communication between the user and the cloud. So with this GUI page user can able to know about the overview of the whole application.

5.2.2 Users and Cloud Storage Servers

The user credential is checked through login page, the service provider gets the username and password from the user and checks in the database is that the user have the credential or not to give request to the cloud. Here also we can add new user through user registration by taking all the important details like user's name, gender, username, password, address, email id, contact number from the user. Here, the cloud user also subscribes to the services offered by the cloud service provider. In this phase we will also create remote storage server in remote location where we can able to store the user data which is allocated to user from cloud service provider on the basis of storage cost.

5.2.3 JAR File Creation

Before the data owner stores the data in the cloud, the data is compressed to a JAR generation. The JAR file contains the original data to be stored as well as the access control policies associated with the data (if any) to the cloud service provider specifying how the end user are authorized to access the content itself. The JAR file which contains the user data item and the corresponding log files are mainly responsible for authentication of entities which want to access the data stored in the file.

5.2.4 Cloud Monitoring of data usage

In this phase we will process the data owner's access control policy rules to access the data. After the data's are received by the cloud service provider, the service provider will have granted access rights, such as time based read, complete read, write, and execute on the data. In order to monitor the actual usage of the data in the cloud, we use some conventional novel logging and auditing techniques. So that for every access to the data the data owner can able to know correctly and automatically logs and periodically inform to them the current usage of the data.

5.2.5 Log Record Generation

In this phase we will create the log records generated by the logger component. Any access to the data in the JAR will trigger the logger component where access to the data can be found out and new log entries are appended in sequentially order. The log record is generated with the content such as the username, the access time and date, the access locations, access type. In particular, a log record takes the following form:

$r_i = \{ID, act, t, Loc, h((ID,Act,T,Loc)|r_i\text{-}1|..|r_1), sig\}$
Here r_i indicates that an entity identified by the ID has

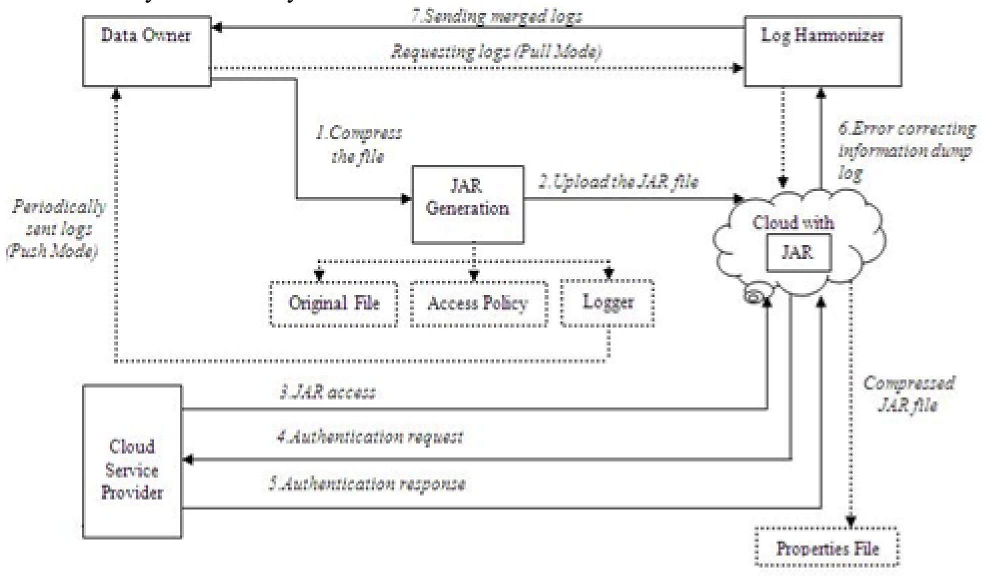

Fig 1: A Framework for Cloud Information Accountability

Performed an action Act on the user's data at time T at location Loc. The component $h((ID,Act,T,Loc)|r_i\text{-}1|..|r_1)$ corresponds to the checksum of the records preceding the newly inserted one, concatenated with the main content of the record itself.

The Logger supports two options:

- Pure Log: Its main task is to record every access to the data. The log files are used for pure auditing purpose.
- Access Log: It has two functions: logging actions and enforcing access control. In case an access request is denied, the JAR will record the time when the request is made. If the access request is granted, the JAR will additionally record the access information along with the duration for which the access is allowed.

5.2.6 Log Alerts

Finally, the data owner will be alert with the automatically generated log files about their data usage using two auditing modes such as push and pull mode. In the push mode the owner will be triggered automatically with the log files. Using this mode we can ensure the size of the log record and timely detection of the user access. In the Pull mode the data owner are allowed to request to retrieve the logs at anytime when they want to check the recent access of their data items.

VI. REFERRED AND USED ALGORITHM

The algorithm used for supporting the auditing strategies such as push and pull mode is as follows [1]

Require: size: maximum size of the log file specified by the data owner, time: maximum time allowed to elapse beforethe log file is dumped, tbeg: timestamp at which the last dump occurred, log: Current log file, pull: indicates whether a command from the data owner is received.

```
1.  Let TS (NTP) be the network time protocol timestamp
2.  Pull := 0
3.  rec := {UID, OID Access Type, Result, Time, Loc}
4.  curtime := TS (NTP)
5.  lsize := size of (log) //current size of the log
6.  if ((cutime -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
```

VII. SECURITY ANALYSIS

We now analyze possible attack to our framework.

7.1 Downloading Attack

The most perceptual attack is that the attacker downloads complete JAR files. The attacker may believe that doing so allows accessing the data in the JAR file without being known by the data owner. Despite, such attack will be encountered by our auditing technique. Recall that through the push mode every JAR file is required to send log records to data owners periodically. That is, even if the data owner is not aware of the existence of the additional copies of its JAR files, he will still be able to receive log files from all existing copies.

which could be integrated with the CIA framework proposed in this work since they build on related architectures.

VIII. CONCLUSION

We proposed innovatory way for automatically logging any access to the data in the cloud along with auditing techniques. Our approach allows the data owner to not only audit his content but also reinforce strong back-end protection if required. Furthermore, one of the main advantages of our work is that it enables the data owner to audit even those copies of its data that were made without his knowledge.

8 RELATED WORKS

In this section, we review the related works addressing the information leakage, lack of trust in cloud, accountability mechanism and SDO purpose

8.1 Information Leakage and Trust

Cloud computing enables highly scalable services to be easily consumed over the Internet on an as-needed basis. As cloud computing is expanding very quickly and used by many individuals and organizations internationally, data protection issues, Users' fear of confidential data leakage [13] and loss of privacy in the cloud as also increased. So they explore an idea of three-tier data protection architecture to accommodate various levels of privacy firms by users. With respect to the architecture, we introduce a novel portable data binding technique to ensure strong enforcement of users' privacy requirements at server. Also The Cloud Security Alliance (CSA) providing security assurance within Cloud Computing. The use of virtualization by CSPs allows to use the server resources to be used more efficiently.

8.2 Accountability and Audit

Researchers have investigated accountability mostly as a provable property through cryptographic techniques, specifically in the context of electronic commerce [4], [18]. The authors propose the usage of policies attached to the data and present logic for accountability data in distributed settings. Likewise, Jagadeesan et al. newly proposed logic for designing accountability-based distributed systems [9].

8.3 Self Defending Objects (SDO)

With respect to Java-based techniques for security, our procedures are affiliated to self-defending objects (SDO) [8].Self-defending objects are an continuation of the object-oriented programming paradigm, where software objects that proposes sensitive functions or hold sensitive data are responsible for conserving those functions/data. Likewise, we also extend the concepts of object-oriented programming. The key variance in our performance is that the authors still depend on a centralized database to keep up the access records, although the items being protected are taken as separate files. In former work, we provided a Java-based approach to prevent privacy leakage from indexing [13],

REFERENCES

[1] S. Sundareswaran, A. Squicciarini, and D. Lin, “Ensuring Distributed Accountability in the Cloud”, IEEE Transactions on Dependable and Secure Computing, vol.9, no.4, July/august 2012

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

[3] R. Corin, S. Etalle, J.I. den Hartog, G. Lenzini, and I. Staicu, “A Logic for Auditing Accountability in Decentralized Systems, ”Proc. IFIPTC1WG1.7 Workshop Formal Aspects in Security and Trust, pp. 187-201, 2005.

[4] B. Crispo and G. Ruffo, “Reasoning about Accountability within Delegation,” Proc. Third Int’l Conf. Information and Comm. Security (ICICS), pp. 251-260, 2001.

[5] Y. Chen et al., “Oblivious Hashing: A Stealthy Software Integrity Verification Primitive,” Proc. Int’l Workshop Information Hiding, F. Petitcolas, ed., pp. 400-414, 2003.

[6] X. Feng, Z. Ni, Z. Shao, and Y. Guo, “An Open Framework for Foundational Proof-Carrying Code,” Proc. ACMSIGPLAN Int’l Workshop Types in Languages Design and Implementation, pp. 67-78,2007.

[7] R.Hasan, R.Sion, and M.Winslett, “The Case of the Fake Picasso: Preventing History Forgery with Secure Provenance,” Proc. Seventh Conf. File and Storage Technologies,pp. 1-14, 2009.

[8] J.W. Holford, W.J. Caelli, and A.W. Rhodes, “Using Self- Defending Objects to Develop Security Aware Applications in Java,” Proc. 27th Australian Conf. Computer Science, vol. 26, pp. 341-349, 2004.

[9] 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

[10] 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.

[11] 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.

[12] S. Pearson and A. Charlesworth, “Accountability as a Way Forward for Privacy Protection in the Cloud,” Proc. First Int’l Conf.CloudComputing, 2009.

[13] Squicciarini, S. Sundareswaran, and D. Lin, “Preventing Information Leakage from Indexing in the Cloud,”Proc.IEEE Int’l Conf. Cloud Computing, 2010

[14] S.sundareswaran, A. Squnicciarini, D. lin and S.Huang, “promoting distribution accountability in the clound”, pro. 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] Reed-Solomon Codes and Teir Applications, S.B. Wicker and V.K. Bhargava, ed. John Wiley & Sons, 1999.

[18] R. Kailar, “Accountability in Electronic Commerce Protocols,” IEEE Trans. Software Eng., vol. 22, no. 5, pp. 313-328, May 1996

[19] S. Roman, Coding and Information Theory. Springer-Verleg, 1992