Volume 4, Issue 7, July 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Accountability and Security of Data Stored in the Cloud

Sheetal A. Deshpande
Department of Computer Engineering,
Sinhgad College of Engineering,
Pune – 411041, Maharashtra, India

Prof. Deepali Gatade
Department of Computer Engineering,
Sinhgad College of Engineering,
Pune – 411041, Maharashtra, India

Abstract*– *Cloud Computing is Internet based computing on a pay-as-you-use basis where-in virtualized servers provide software, infrastructure, platform, devices and other resources in a shared environment to customers. The most important feature of the cloud services is that users' data are processed in remote, unknown machines that users do not possess or control. While taking benefit of this new emerging technology, users' fear of losing command of their own data, is becoming a noteworthy hurdle to the extensive implementation of cloud services.

To address this problem we propose a framework based on information accountability to keep trail of the authentic handling of the users' data in the cloud. The system secures the data stored on the cloud server while allowing the users to share the data with others. Based on an object-centric approach the system permits logging mechanism on the users' data. It allows the data owners to restrict sharing of their data by providing access policies on their data. To reinforce user's control on his/her data, we provide two different auditing mechanisms.

Keywords – Data accountability, Data sharing, Access control, Cloud Computing, Security.

I. INTRODUCTION

Cloud computing facilitates highly scalable services to be easily used over the Internet without having any previous knowledge on controlling the resources involved. Cloud computing customers do not own the physical infrastructure; rather they rent them from a third-party provider. They consume resources as a service and pay only for the resources that they have used. Most Cloud computing infrastructure consists of services distributed through common centres and built on servers. Sharing resources can improve, as servers are not unnecessarily left inoperative, which reduces cost extensively, and increases the speed of application development.

When it comes to cloud storage, providers of the cloud are giving people more than just data storage. They are giving them the experience of accessing, sharing, collaborating, streaming, and integrating data. Cloud storage refers to saving data to an remote storage system maintained by a third party. Instead of storing information on your computer's hard drive or other local storage device, you save it to a remote database. The Internet provides the connection between your computer and the database.

The concept of the cloud computing model is that customers' data, which may be of individuals, organizations or enterprises, is stored remotely on unknown servers that users cannot control. The convenience and competence of cloud computing services, however, comes with privacy and security risks. A significant barricade to the acceptance of cloud services is the users' panic of private data leak and loss of confidentiality in the cloud. The course of fortification of users' data starts from the stage the user commences his cloud experience. The user has to manually classify the cloud provider that meets his privacy requirements, which is difficult for end-users. Also many more problems arise after storing data on the clouds.

II. LITERATURE SURVEY

The cloud infrastructures are much more commanding and dependable than personal computing devices, hence wide range of both internal and external fear of data integrity still exist. Since users do not keep hold of a local copy of outsourced data, there exist various reasons for CSP to act treacherously towards the cloud users concerning the status of their outsourced data. In order to accomplish the guarantee of cloud data integrity and availability and enforce the quality of cloud storage service, proficient methods that permit on-demand data correctness verification on behalf of cloud users have been designed.

A. Policy-driven framework for protecting data privacy during service provisioning.

The framework consists of three key components: policy ranking, policy integration, and policy enforcement. To understand the framework let us see an example, where a user Bob joined the cloud and found six cloud service providers, each of them able to provide the service that Bob needs. In order to find the service provider whose privacy policies match approximately with Bob's privacy requirements, Bob's privacy requirements and policies from service

providers are fed into the policy ranking module together. The ranking module will help to select a service provider S2 for Bob. But still S2's privacy policies may not exactly match Bob's requirements. In the second step the policies of both Bob and S2 are sent to the policy integration module which will help generate an integrated policy as approved by both parties. The integrated policy is in two formats. One is in an actual policy format, i.e., a policy written in certain policy language. The other is in an executable format (like a Java JAR file) which will be used for enforcing the policy subsequently [4]. Bob's data privacy will be protected by the executable policy throughout the service. The executable policy also travels among contractors associated with service provider S2. It is worth noting that here they have focused on user-related privacy policies rather than security policies at server side. Policy Ranking will help the user to find the service provider with the most similar privacy policies compared to the users' privacy requirements (or policies) and it is carried out at initial stage when user is searching for service providers. Ranking of policies can be carried out by either users who want to avail cloud service or the service providers itself or the brokers. Policy Integration is the next step in which the user selects a service provider, is to integrate the privacy policies of the user as well as the service provider, resolve all possible conflicts and achieve in harmony, agreement of all the requirements from both the parties. Policy integration module takes all privacy requirements as input and helps generate policies to be adopted by participating parties[4]. After the policies have been created the last step is to correctly enforce them to guarantee the protection promised by the policies. Policy enforcement should occur while satisfying integrity, availability and confidentiality of data and policies. Policy enforcement can be achieved using two ways : Tight Coupling; and Loose Coupling.

B. Flexible distributed storage integrity auditing mechanism

Moving data into the cloud offers immense expediency to users since they don't have to care about the complexities of direct hardware management. But this is resulting on dependency on clouds. There are a lot of possibilities for the CSPs to behave deceitfully towards the cloud users as regards to the status of their own outsourced data. The CSP may cast off rarely accessed data without informing the user to increase the profit margin; they can also attempt to hide data loss incidents so as to maintain a reputation[6]. Thus, even if outsourcing of data into the clouds is beneficial for long-term large-scale data storage, it is deficient in assurance of data integrity and availability and hence its wide and tremendous use will affect both enterprise and individual cloud users. In order to achieve the assurances of cloud data integrity and availability and enforce the quality of cloud storage service, proficient methods that facilitate on-demand data correctness substantiation on behalf of cloud users have to be designed. The first step is to prepare the file distribution across cloud servers by making use of coding theory. The erasure-correcting code is used to abide to multiple failures in distributed storage systems. Before file distribution the user precomputes a definite number of short verification tokens on individual vector, where each token covers a random subset of data blocks[6]. Later, when the user wants to make sure the storage correctness for the data in the cloud, he challenges the cloud servers with a set of randomly generated block indices. Upon receiving the dare, each cloud server figures out a short "signature" over the specified blocks and returns them to the user. In the Correctness Verification and Error Localization step the response values from servers for each test determine the accuracy of the distributed storage, and also maintain data to locate budding data errors. This is accomplished using the challenge-response protocol. The last step is File Retrieval and Error Recovery in which the error recovery is done and the original file is recovered. The user can regenerate the original file by downloading the data vectors from the first m servers. After the data corruption is identified, the comparison of precomputed tokens and received response values can certifiy the identification of disobedient servers.

C. Data protection as a service architecture

DPaaS is a group of security primitives easily reachable on a cloud podium, which imposes data security and privacy and offers verification of privacy to data owners, even in the incidence of likely negotiated or malevolent applications[8]. DPaaS implements fine-grained admission control rules on data units through application captivity and information flow checking. It utilizes cryptographic security at rest and presents vigorous logging and auditing to provide accountability. Significantly, DPaaS moreover directly tackles the matters of brisk expansion and maintenance. DPaaS is a data-fortification resolution at the platform layer. The cloud platform providers need to offer DPaaS besides their current hosting environment, since it could be profitable for small companies or developers because they do not have enough in-house security knowledge, to help them for gaining user confidence more faster[8]. The DPaaS methodology offers logging and auditing at the platform level, contributing the profit to all the applications running over it. The inspector verifies offline whether the platform implements all the data protection attributes. The platform supplier, online, can utilize trusted computing (TC) technologies to confirm that the meticulous software is running. DPaas exercises grouping of encryption at rest, application detention, information flow checking, and auditing to certify the security and privacy of users' data.

III. PROBLEM DEFINITION

The concept of the cloud computing model is that customers' data, which can be of individuals, organizations or enterprises, is processed remotely in unknown machines that users do not own or operate. The convenience and efficiency of this approach, however, comes with privacy and security risks. A considerable obstacle to the adoption of cloud services is the users' fear of confidential data leakage and loss of privacy in the cloud. The process of protection of users' data begins from the stage the user starts his cloud experience. The user has to manually identify the cloud provider that meets his privacy requirements, and this is often significant burden for end-users. Also many more problems arise after storing data on the clouds.

With respect to the above mentioned scenario we can identify some common requirements needed to achieve data accountability in the cloud. With the above scenario in mind, we identify the common requirements and develop several guidelines to achieve data accountability in the cloud. A user who subscribed to a certain cloud service, usually needs to send his/her data as well as associated access control policies (if any) to the service provider. After the data are received by the cloud service provider, the service provider will have granted access rights, such as read, write, and copy, on the data[2].

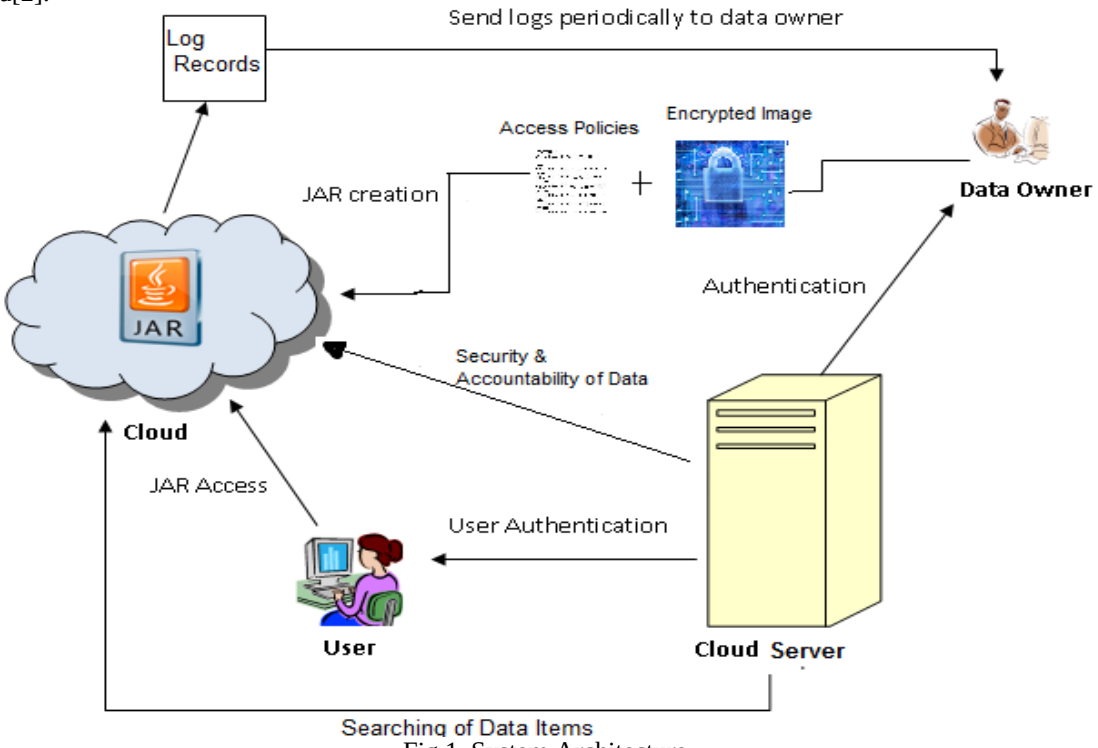

Fig 1. System Architecture

Using conventional access control mechanisms, once the access rights are granted, the data will be fully available at the service provider. In order to track the actual usage of the data, we need to develop logging and auditing techniques.

IV. FRAMEWORK FOR THE SHARED DATA ACCOUNTABILITY IN THE CLOUD

To address the above discussed problem we have developed a system with automated logging and auditing of relevant access performed by any entity, carried out at any point of time at any cloud service provider. At the beginning, each user(data owner) will upload his data along with some access policies for that data to be hosted on the cloud. The data is encrypted using powerful encryption techniques and stored in JAR along with the policies. The JAR file contains encrypted data items and set of simple access control rules to access the data items. The process of transferring uploading the data on the cloud is carried out in a secured environment using protocols which provide security and reliability at transport level. In the meanwhile if a user tries to access data available on cloud, the user is first authenticated and then access is provided to the user based on the access control policies provided in the JAR. Each time there is an access to the data, the access automatically triggers an event which generates a log record, encrypts it and store it in a specific format. The logs are further sent to the data owner periodically for auditing purposes. The data owner is also able to access the log at any point of time, whenever required.

If the access request is granted, the log record contains the access information along with the duration for which the access is allowed. The users are allowed four types of actions on the data shared by data owners. They are view, download, timed_access, and Location-based_access. The system identifies the location of the user before granting the user any access to the data[3]. The data owner can restrict usage of his/her data for specific regions and also for specific interval of time. For each action, we propose a specific method to correctly record or enforce it depending on the type of the logging module.

In this method we propose auditing mechanism which supports two auditing strategies: push and pull. In the push strategy, the log file is pushed back to the data owner periodically in an automated fashion. The pull mode is an on-demand approach, whereby the log file is obtained by the data owner as often as requested. If there are multiple users for the same set of data items, the log records will be merged before sending back to the data owner.

Also keyword based searching is provided to the users, for the ease of hunting for the data he requires. The meta data contains the keywords based on which the searching is carried out. This approach allows the data owner to not only audit his content but also enforce strong security to his data.

A. Generation of JARs

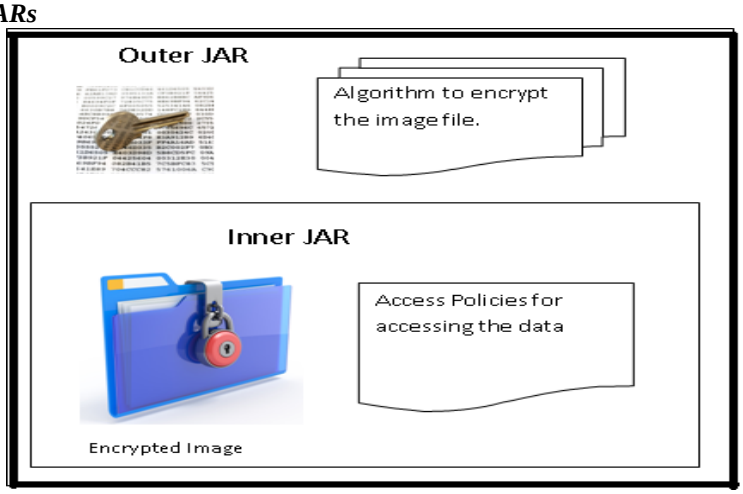

Fig. 2. Structure of JAR file

The data owner provides the system with data items and access policies to be shared on cloud. These data items are first encrypted using RSA encryption algorithm. These encrypted data items along with the keys are bundled together in an inner JAR. The inner JAR file is again bundled along with the access policies for the data items in an outer JAR. The outer JAR is locked using the locking mechanism available in java for JAR files and then they are stored on the cloud server. The JAR file includes a set of simple access control rules specifying whether and how the cloud servers, and possibly other data stakeholders (users, companies) are authorized to access the content itself.

B. Creating a Secure Transaction Environment

HTTPS provides authentication of the web site and associated web server that one is communicating with, which protects against man-in-the-middle attacks. Additionally, it provides bidirectional encryption of communications between a client and server, which protects against eavesdropping and tampering with and/or forging the contents of the communication. In practice, this provides a reasonable guarantee that one is communicating with precisely the web site that one intended to communicate with (as opposed to an imposter), as well as ensuring that the contents of communications between the user and site cannot be read or forged by any third party. We have implemented this secure environment using OpenSSL. The entire transaction of uploading data on cloud and downloading data from the cloud server takes place in this secure environment.

C. Authorization of Users

The users of the application are classified into two categories: free and paid. Both the type of users has to be register before they can access the data. While registering the users have to specify whether they are free users or paid users. Paid users have to mention their credit card details at the time of registration. They details are required for further use.

The paid users have all the access privileges to the data stored on the cloud. The paid users can view the data as well as download it. If the paid user downloads any data item, then he will be charged on his credit card accordingly. His credit card details will have been taken at the time of registration. The free users have the facility to only view the data items. They cannot download, nor can they copy it using keyboard controls. They keyboard controls and mouse keys are disabled for the free user.

D. Data Access

The data owner has the facility to restrict the visibility of his data for some particular locations and/or limited period of time. At the time of uploading the data the data owner can specify the name of countries for which the data is restricted. When the user (paid or free) logs on to the system to view the data, the IP address of the machine, through which the user is accessing the system, is tracked and based on the IP address the location of the user is identified. If the user is currently viewing the data from a particular country to which the data is restricted then the user is not able to view the data and is displayed a message accordingly.

Similarly the data owner can also specify the duration for which the data items are visible to the users. Once the user starts viewing the data , a timer is started in the reverse order according to the time mentioned by the data owner, which will automatically logout once the time is finished. Thus the data can be made available to the users for a specific period of time only.

E. Generation of Logs

According to the type of user and the access policies mentioned for the particular data the user starts accessing the data items. Every time the user access the data, an event is triggered which generates a log record, mentioning the action performed by the user on the data item[1]. The log record is of the form (ID, Act, Time, Loc) indicating that an entity identified by ID has performed an action Act on the user's data at time Time at location Loc.

The log records are immediately added to the log file after the record is generated. The log files are encrypted and stored on cloud server. MD5 hashing is performed on the log records. The log records are generated for every collection separately. Before sending log records to data owner the log records are merged and sent in a single file.

F. Auditing Mechanism

The logging mechanism is responsible for keeping a track of actual usage of data stored on the cloud and reporting it to the data owners respectively. The log files are encrypted and stored on the cloud server. To allow the data owners to be timely and accurately informed about their data usage, the logging mechanism is complemented by an auditing mechanism. It supports two auditing modes: 1) push mode; 2) pull mode.

1) In push mode, the logs are periodically pushed to the data owner (or auditor) by the harmonizer. The push action will be triggered whenever the time elapses for a certain period according to the temporal timer inserted. The log records are decrypted and sent to the email-id specified by the data owner at the time of registration. The log records are transported on a secure environment by making use of SMTPS protocol. By construction of the records, the auditor will be able to quickly detect forgery of entries.

2) The pull 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 accepting an email-id from the data owner and sending the log files using SMTPS protocol. Before sending the log file, the log records of all the collections of that particular data owner are merged and sent.

G. Keyword based Searching

The data items are stored inside the JAR files. The user is not aware of what exactly is the data about. Thus, when the data owner is uploading the data items, some information about the data is gathered and stored. This information is used for searching of a particular data. The data will be searched on the information provided by the data owner. A database is created in MYSQL for storing the metadata. Whenever the user is searching for a particular data, the data base is queried and the dataset is returned [2].

V. SECURITY DISCUSSION

The Accountability of Shared Data on the Cloud system conducts automated logging and distributed auditing of relevant access performed by any entity, carried out at any point of time at any cloud service provider. The basic approach is to leverage and extend the programmable capability of JAR (Java ARchives) files and automatically log the usage of the users' data by any entity in the cloud. By means of this system, 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.

We now analyze possible attacks to our system and their effects on our system. Our analysis is based on a semi honest adversary model by assuming that a user does not release his master keys to unauthorized parties, while the attacker may try to learn extra information from the log files. We assume that attackers may have sufficient Java programming skills to disassemble a JAR file and prior knowledge of our system. We also assume that the JVM is not corrupted.

A. Copying Attack

If the attacker copies the entire JAR file, he feels that it may allow him to allow access to data in JAR files without the notice of the data owner. But this attack is handled by the system. Every JAR file is required to be unlocked by the locking-unlocking mechanism of JAR files present in the system. If attackers move copies of JARs to places where the system cannot connect, the copies of JARs will soon become inaccessible. If the JAR cannot contact the system, the access to the content in the JAR will be disabled.

B. Disassembling Attack

One more type attack is to disassemble the JAR file of the logger and then attempt to extract useful information out of it or spoil the log records in it. The use of RSA algorithm does not allow the attacker to decrypt any data or log files in the disassembled JAR file. The attacker cannot modify the log files extracted from a disassembled JAR to write fake records because the log files are also encrypted by the system.

C. Man-in-the-Middle Attack

An attacker may intercept messages during the transaction of data transfer in the system. But it cannot succeed because the system run in a secure environment developed using OpenSSL[2]. The system uses HTTPS protocol instead of HTTP protocol to transfer data from the client to the server and from the server to the client. The HTTPS protocol guarantees data transfer at transport layer as also prevention of man-in-middle attack because of certificate generation.

VI. RESULTS

The Accountability of Shared Data on the Cloud system runs on Amazon Elastic Compute Cloud. The cloud server has Windows Server 2008 operating system.

A. Encryption & JAR creation Time

The time taken to upload data of the data owner is the combination of encrypting data items and bundling the data items along with access policies into a JAR file. The time taken for encrypting the data items increases as the size of the data items increases. The JAR file may consists of 1 data item, 2 data items or n number of data items. The time increases on the total size of all data items and not on the numbers of data items in a JAR.

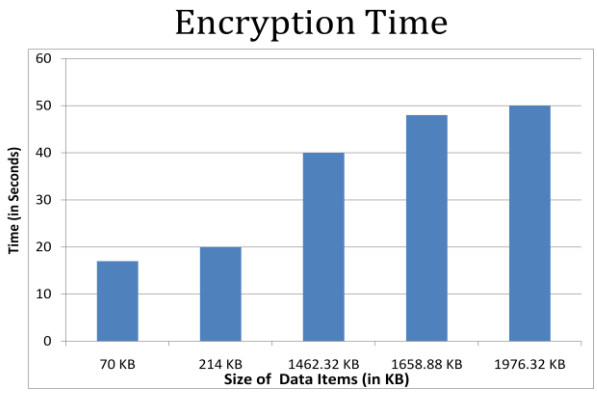

Fig 3. Encryption Graph

B. Decryption Time

The time taken for decryption is the sum of time taken for unlocking JAR file, extracting data, decrypting the data and displaying them to the user. The time taken for decrypting the data items increases as the size of the data items increases. The JAR file may consists of 1 data item, 2 data items or n number of data items. The time increases on the total size of all data items and not on the numbers of data items in a JAR. Comparatively the time taken for decryption is less than the time taken for encrypting the same size of JAR file.

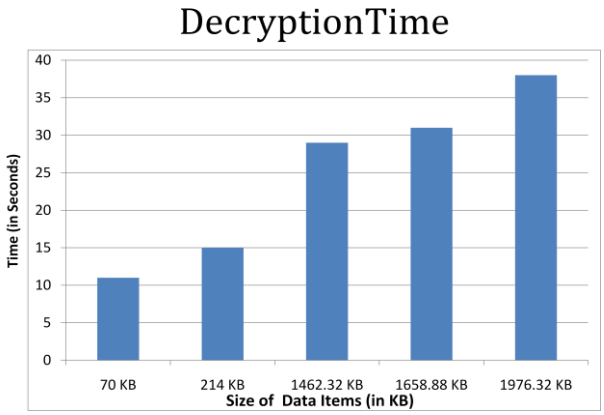

Fig 4. Decrption Graph

C. Log Merging Time

The system creates separate log files for all the collections. Whenever the log records are sent to the data owner, all the log files of the collections belonging to the data owner have to be sent. Hence the log records are first merged and then sent to the data owner. We can observe that the time increases almost linearly to the number of files and size of files. The least time being taken for merging two 100 KB log files at 63 ms, while the time to merge 300KB file was 50 seconds.

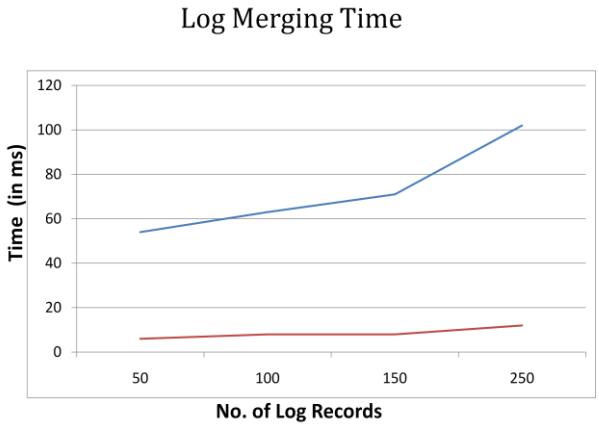

Fig 5 Log Merging Graph

D. Size of the JAR file

The original size of the data files is the total size of the all the data items in the collection. The size of the outer JAR after encryption of data and bundling it with access policies increases negligibly by 2 to 4 kB. This increase is because of encryption of data items and addition of access policies.

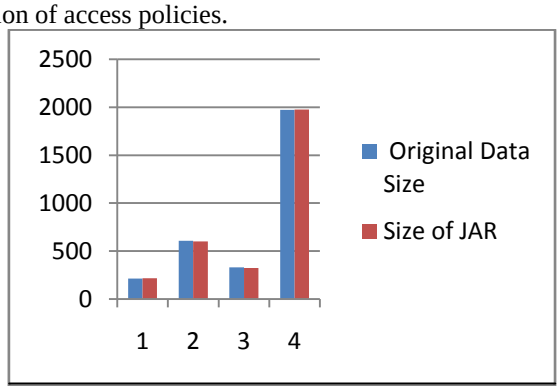

Fig 6. Size of JAR files Graph

VII. CONCLUSION AND FUTURE SCOPE

Cloud computing has raised a range of important *privacy* and *security* issues. Such issues are due to the fact that, in the cloud, users' data and applications reside at least for a certain amount of time on the cloud cluster which is owned and maintained by a third party. Concerns arise since in the cloud it is not always clear to individuals why their personal information is requested or how it will be used or passed on to other parties. The privacy problem in the cloud is also compounded by the fact that some of the issues are not technical in nature, and rather deal with law and regulations.

To achieve accountability along with security and privacy of data on the cloud we have developed a system based on Cloud Information Accountability Framework. This approach uses the programmable capability of JAR (Java Archives) files to automatically log the usage of the users' data by any entity in the cloud. Users send their data along with any policies such as access control policies and logging policies that they want to enforce, enclosed in JAR files, to cloud service providers. Any access to the data triggers an automated and authenticated logging mechanism local to the JARs. The data is encrypted and stored in the inner JAR. The inner JAR along with the access policies to access the data are stored in an outer JAR. The outer JAR is locked and the entire process of uploading and downloading data is carried out on a secure environment using OpenSSL. And the most important feature is that it enables the data owner to audit even those copies of its data that were made without his knowledge. Meta-data of the files is stored in a database, which helps the users to search for the data they need to view.

The project has been developed to work on image data files. It can be further modified to work on any type of file, making it useful for any educational institutes, universities, etc to share timetables, notices, subject notes, etc with the students and staff. The project can be enhanced further to verify the integrity of the JRE and the authentication of JARs. Also the project can be expanded to more than on CSPs, where the data owners choose to their store data on servers of more than one CSP.

REFERENCES

[1] Deepali Gatade , Sheetal Deshpande , "A Survey and Analysis for Accountability and Privacy of Shared Data in the Cloud", National Conference on Innovative Paradigms in Engineering & Technology (NCIPET-2013) , Proceedings published by International Journal of Computer Applications (IJCA), Number: 1, Pages(s):22-27, Publication Year: December 2013.

[2] Deepali Gatade , Sheetal Deshpande, "Accountability of Shared Data in the Cloud", Computer – Post Graduate Conference in Association with ACM Professional Chapter Pune, Year : March 2013.

[3] Anna C. Squicciarini , Dan Lin, Smitha Sundareswaran, "Ensuring Distributed Accountability for Data Sharing in the Cloud", IEEE Transactions on Dependable and Secure Computing, Volume:9 , Issue: 4, Page(s): 556-568, Publication Year: 2012

[4] Anna C. Squicciarini , Dan Lin, Shuo Huang, Smitha Sundareswaran, "Promoting Distributed Accountability in the Cloud", IEEE Transactions on Dependable and Secure Computing, Page(s): 113- 120, Publication Year: 2011

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

[6] Cong Wang, Qian Wang, Kui Ren, Wenjing Lou, "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing" Publication Year: 2011, Page(s): 1.

[7] B. Carstoiu, D. Carstoiu, "High Performance Eventually Consistent Distributed Database Zatara", *Networked Computing (INC)*, 978-1-4244-6986-4, Pages 1-6, May 2010.

[8] Cong Wang, Qian Wang, Kui Ren, Wenjing Lou, "Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing", INFOCOM, Publication Year: 2010, INFCOM.2010.5462173, Page(s): 1 - 9.

[9] David G. Campbell, Gopal Kakivaya, Nigel Ellis, "Extreme Scale with Full SQL Language Support in Microsoft SQL Azure", *SIGMOD'10* , ACM 978-1-4503-0032- 2/10/06, Pages 1021-1024, June 2010.

[10] Mateljan, V.Cisic, D.Ogrizovic,D., "Cloud Database-as- a-Service (DaaS) – ROI", MIPRO 2010 , 978-1-4244-7763-0, Pages 1185- 1188, May 2010.

[11] Kaufman L.M., "Data Security in the World of Cloud Computing" , Security & Privacy, IEEE MSP.2009.87, Publication Year: 2009, Page(s) : 61 – 64.

[12] 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.

[13] Crispo and G. Ruffo, "Reasoning about Accountability within Delegation," Proc. Third Int'l Conf. Information and Comm. Security (ICICS),pp. 251-260, 2001.

