DATA ACCOUNTABILITY IN CLOUD USING RELIABLE LOG FILES FORWARDING

R.N.Heames II-M.E. (CSE),
MKCE, Karur - 639 113.
Email: rnheames@gmail.com

Dr.P.SUDHAKAR M.Tech., Ph.D.,
HOD / CSE.
MKCE, Karur - 639 113.

ABSTRACT: Cloud computing is receiving a great deal of attention, both in publications and among users, from individuals at home to the government. The cloud makes it possible for you to access your information from anywhere at any time. While a traditional computer setup requires you to be in the same location as your data storage device, the cloud takes away that step. The cloud removes the need for you to be in the same physical location as the hardware that stores your data. A major feature of the cloud services is that user's data are usually processed remotely in unknown machines that users do not own or operate. Data handling in the cloud goes through a complex and dynamic hierarchical service chain which does not exist in conventional environments. This can be addressed by a novel approach, namely Cloud Information Accountability (CIA) framework. In this every access to the data are correctly and automatically logged. 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 any time by their data owners when needed regardless the location where the files are stored. The end user is allowed to access the data as per their access privileges which they specifies while registering to access the data in the cloud and authentication is provided and data user will be verified.

*****

1. INTRODUCTION

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet). The name comes from the use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams. Cloud computing entrusts remote services with a user's data, software and computation. The data processed on clouds are often outsourced, leading to a number of issues related to accountability, including the handling of personally identifiable information. Such fears are becoming a significant barrier to the wide adoption of cloud services. It is essential to provide an effective mechanism for users to monitor the usage of their data in the cloud Conventional access control approaches developed for closed domains such as databases and operating systems, or approaches using a centralized server in distributed environments, are not suitable, due to the following features characterizing cloud environments. First, data handling can be outsourced by the direct cloud service provider (CSP) to other entities in the cloud and theses entities can also delegate the tasks to others, and so on. Second, entities are allowed to join and leave the cloud in a flexible manner. As a result, data handling in the cloud goes through a complex and dynamic hierarchical service chain which does not exist in conventional environments.

To overcome the above problems, a novel approach, namely Cloud Information Accountability (CIA) [01] framework was proposed, based on the notion of information accountability .Information account-ability focuses on keeping the data usage transparent and traceable. The proposed CIA framework provides end-to-end accountability in a highly distributed fashion. By means of the CIA [02], data owners can track not only whether or not the service-level agreements are being honoured, but also enforce access and usage control rules as needed. Associated with the accountability feature, two distinct modes for auditing: push mode and pull mode.

The push mode refers to logs being periodically sent to the data owner or stakeholder while the pull mode refers to an alternative approach whereby the user (or another authorized party) can retrieve the logs as needed. The design of the CIA framework presents substantial challenges, including uniquely identifying CSPs, ensuring the reliability of the log,

adapting to a highly decentralized infrastructure, etc.

The basic approach toward addressing these issues is to leverage and extend the programmable capability 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 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 will trigger an automated and authenticated logging mechanism local to the JARs. This type of enforcement as "strong binding" since the policies [07] and the logging mechanism travel with the data. This strong binding exists even when copies of the JARs are created; thus, the user will have control over his data at any location.

Access Control Mechanism by which the end use will be allowed to access the data based on their access control policy which they given in the initial registration with CSPs. Accessed data of the end user will be verified and authentication will be provided.

Accountability: Implementation of the Cloud Information Accountability (CIA) will give the way for data accountability .The components are logger and log harmonizer. Logger will log the events and send the log harmonizer. Log harmonizer will have two modes pull and push. In push mode the log files will send to the data owner and in pull mode the log details will be sending by owner request. Pull mode will be responsible for the history of the log files. Log files are get encrypted and send to the CIA where the data providers can see the log details and alternatively the log files can be send to the data providers and it can be decrypted. Along with the Accountability [08] it is possible to verify the data which is accessed and this verification of the accessed data is optional [03].

main contributions are ,

- A novel automatic and enforceable logging mechanism in the cloud. To the knowledge, this is the first time a systematic approach to data accountability through the novel usage of JAR files is proposed.
- The proposed architecture is platform independent and highly decentralized, in that it does not require any dedicated authentication or storage system in place.
- Beyond traditional access control in that a certain degree of usage control was provided for the protected data after these are delivered to the receiver [05], [06].
- Verification of the user is done who is accessing the data.

2. PROBLEM STATEMENT

A novel automatic and enforceable logging mechanism in the cloud was implemented. the existing system. The Cloud Information Accountability framework was given so that the work 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.

- When the data providers publish the data in the cloud the access policy along with the data is provided.
- Authentication is not between the end user and the cloud storage. End user can unable to verify the data which he/she accessed from the CSPs. It can be addressed by use of logon passwords. In private and public computer networks (including the Internet), authentication is commonly done through the use of logon passwords. Knowledge of the password is assumed to guarantee that the user is authentic. Each user registers initially (or is registered by someone else), using an assigned or self-declared password. On each subsequent use, the user must know and use the previously declared password.

3. CLOUD INFORMATION ACCOUNTABILITY

The Cloud Information Accountability framework will perform automated logging and distributed auditing of relevant access performed by any entity, carried out at any point of time at any cloud service provider. It has two major components: logger and log harmonizer.

3.1 Major Components

There are two major components of the CIA, the first being the logger, and the second being the log harmonizer. The logger is the component which is strongly coupled with the user's data, so that it is downloaded when the data are accessed, and is copied whenever the data are copied. It handles a particular instance or copy of the user's data and is responsible for logging access to that instance or

copy. The log harmonizer forms the central component which allows the user access to the log files.

The logger is strongly coupled with users. Its main tasks include automatically logging access to data items that it contains, encrypting the log record using the public key of the content owner, and periodically sending them to the log harmonizer. The logger requires only minimal support from the server in order to be deployed. The tight coupling between data and logger, results in a highly distributed logging system, therefore meeting the first design requirement. Furthermore, since the logger does not need to be installed on any system or require any special support from the server, it is not very intrusive in its actions, thus satisfying the fifth requirement. Finally, the logger is also responsible for generating the error correction information for each log record and sends the same to the log harmonizer. The error correction information combined with the encryption and authentication mechanism provides a robust and reliable recovery mechanism, therefore meeting the third requirement. The log harmonizer is responsible for auditing.

Being the trusted component, the log harmonizer generates the master key. It holds on to the decryption key for the IBE key pair, as it is responsible for decrypting the logs. Alternatively, the decryption can be carried out on the client end if the path between the log harmonizer and the client is not trusted.

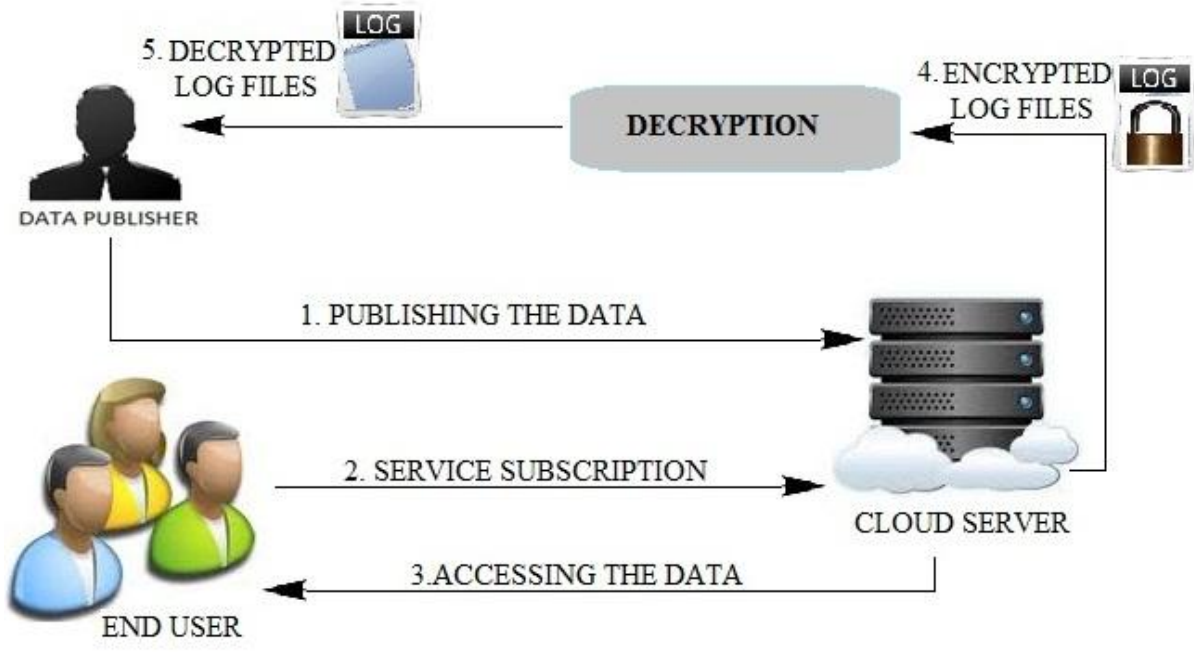
In this case, the harmonizer sends

The key to the client in a secure key exchange.

It supports two auditing strategies: push and pull. Under 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. In case there exist multiple loggers for the same set of data items, the log harmonizer will merge log records from them before sending back to the data owner. The log harmonizer is also responsible for handling log file corruption. In addition, the log harmonizer can itself carry out logging in addition to auditing. Separating the logging and auditing functions improves the performance. The logger and the log harmonizer are both implemented as lightweight and portable JAR files. The JAR file implementation provides automatic logging functions.

3.2 Data Flow

The overall CIA framework, combining data, users, logger and harmonizer is sketched in Fig. 1. At the beginning, each user creates a pair of public and private keys based on Identity-Based Encryption [04] (step 1 in Fig. 1). This IBE scheme is a Weil-



OVERVIEW FOR LOG FILES GENERATION

pairing-based IBE scheme, using the generated key, the user will create a logger component which is a JAR file, to store its data items.

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. Then, he sends the JAR file to the cloud service provider that he subscribes to. To authenticate the CSP to the JAR (steps 3-5), use OpenSSL-based certificates, wherein a trusted certificate authority certifies the CSP. In the event that the access is requested by a user, employ SAML-based authentication, where in a trusted identity provider issues certificates verifying the user's identity based on his username.

Once the authentication succeeds, the service provider (or the user) will be allowed to access the data enclosed in the JAR. Depending on the configuration settings defined at the time of creation, the JAR will provide usage control associated with logging, or will provide only logging functionality. As for the logging, each time there is an access to the data, the JAR will automatically generate a log record, encrypt it using the public key distributed by the data owner, and store it along with the data (step 6). The encryption of the log file prevents unauthorized changes to the file by attackers. The data owner could opt to reuse the same key pair for all JARs or create different key pairs for separate JARs. Using separate keys can enhance the security without introducing any overhead except in the initialization phase. In addition, some error correction information will be sent to the log harmonizer to handle possible log file corruption. To ensure trustworthiness of the logs, each record is signed by the entity accessing the content. Further, individual records are hashed together to create a chain structure, able to quickly detect possible errors or missing records. The encrypted log files can later be decrypted and their integrity verified. They can be accessed by the data owner or other authorized stakeholders at any time for auditing purposes with the aid of the log harmonizer (step 8).

4. AUTOMATED LOGGING MECHANISM

4.1 The Logger Structure

The main responsibility of the outer JAR is to handle authentication of entities which want to access the data stored in the JAR file. In the context, the data owners may not know the exact CSPs that are going to handle the data. Hence, authentication is specified according to the servers' functionality, rather than the server's URL or identity. As discussed below, the outer JAR may also have the access control functionality to enforce the data owner's requirements, specified as Java policies, on the usage of the data. A Java policy specifies which permissions are available for a particular piece of code in a Java application environment. The permissions expressed in the Java policy are in terms of File System Permissions. However, the data owner can specify the permissions in user-centric terms as opposed to the usual code-centric security offered by Java, using Java Authentication and Authorization Services. Moreover, the outer JAR is also in charge of selecting the correct inner JAR according to the identity of the entity who requests the data.

Each inner JAR contains the encrypted data, class files to facilitate retrieval of log files and display enclosed data in a suitable format, and a log file for each encrypted item.

It support 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. CONCLUSION

Innovative approaches for automatically logging any access to the data in the cloud together with an auditing mechanism and user access privilege. Data access can be controlled by implementing the user privilege. Moreover, one of the main features of the work is that it enables the data owner to audit even those copies of its data that were made without his knowledge.

REFERENCES

[01] Anna.C.Squicciarini, Smitha Sundareswaran, "Ensuring Distributed Accountability for Data Sharing in the Cloud" IEEE Transactions On

Dependable And Secure Computing, Vol. 9, No. 4, July/August 2012.

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

[03] C.Wang, Qian Wang and Kui Ren "Privacy-Preserving Public Auditing for Secure Cloud Storage" Computers, IEEE Transactions on Publication Year: 2011

[04] D.Boneh and M.K.Franklin, "Identity-Based Encryption from the Weil Pairing" Proc. Int'l Cryptology Conf. Advances in Cryptology, pp. 213-229, 2001.

[05] E.Barka and A.Lakas, "Integrating Usage Control with SIP-Based Communications" J. Computer Systems, Networks, and Comm., vol. 2008, pp. 1-8, 2008.

[06] F.Martinelli and P.Mori, "On Usage Control for Grid Systems" Future Generation Computer Systems, vol. 26, no. 7, pp. 1032-1042, 2010.

[07] 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.

[08] 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