IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 2, Ver. XII (Mar-Apr. 2014), PP 08-13
www.iosrjournals.org

Secured Cloud Information Accountability for Data Sharing Using Identity Based Encryption Algorithm

Rohan M. Bangar[1], Prof. B. B. Gite [2]
[1]*(Department of Computer Engineering, Pune University, India)*
[2]*(HOD, Department of Computer Engineering, Pune University, India)*

Abstract : *Cloud Computing provides scalable services which can be used by user on a need basis. These services can be accessed through the internet. User stores important data (mainly, financial and health data) on server. This data is processed by the remote server which will be unknown to the user. Providing security is the main challenge for cloud services. To address this problem, we propose a framework which will keep record of the usage of the data in the cloud. Mainly it is an object-centered approach which helps to enable logging mechanism for the data. We provide the JAR mechanism which helps to create dynamic and travelling objects. These travelling objects will be used for authentication for the access of the data. To provide security, we have introduced auditing mechanisms, data possession for integrity verification and identity based encryption algorithm.*
Keywords: *About Accountability, Cloud Computing, Data sharing.*

I. Introduction

In this paper we have proposed Cloud information accountability framework which does automated logging and distributed auditing mechanisms. It has two main components: logger and log harmonizer. Access control rules are defined in JAR file which controls the access over data for stakeholders and authorized users. Logger components are initiated on JRE. We are also checking the integrity of JRE on the system. Integrity check is handled by oblivious hashing. Additional layer of security is provided to the infrastructure of Cloud by converting the JAR into obfuscated code. Apart from this we are providing more security to the user data. JAR will provide the logging functionality and usage control on the data. This is handled by configuration settings at the time of JAR creation. JAR will keep the log record for all the users.

II. Literature Survey

[1] Title: **"**A Secure Erasure Code-Based Cloud Storage System with Secure Data Forwarding ".
Author: Hsio Ying Lin,Tzeng.W.G, ,IEEE transactions on parallel and distributed systems,2012.
Summary: A cloud storage system, consisting of a collection of storage servers, provides long-term storage services over the Internet. Storing data in a third party's cloud system causes serious concern over data confidentiality. General encryption schemes protect data confidentiality, but also limit the functionality of the storage system because a few operations are supported over encrypted data. Constructing a secure storage system that supports multiple functions is challenging when the storage system is distributed and has no central authority. We propose a threshold proxy re-encryption scheme and integrate it with a decentralized erasure code such that a secure distributed storage system is formulated. The distributed storage system not only supports secure and robust data storage and retrieval, but also lets a user forward his data in the storage servers to another user without retrieving the data back. The main technical contribution is that the proxy re-encryption scheme supports encoding operations over encrypted messages as well as forwarding operations over encoded and encrypted messages. Our method fully integrates encrypting, encoding, and forwarding. We analyze and suggest suitable parameters for the number of copies of a message dispatched to storage servers and the number of storage servers queried by a key server. These parameters allow more flexible adjustment between the number of storage servers and robustness.

[2]Title: "Identity-Based Encryption from the Weil Pairing,"
Author: D. Boneh and M.K. Franklin,Proc. Int'l Cryptology Conf. Advances in Cryptology,pp. 213-229, 2001.
Summary: We propose a fully functional identity-based encryption scheme (IBE). The scheme has chosen ciphertext security in the random oracle model assuming an elliptic curve variant of the computational Diffie-Hellman problem. Our system is based on bilinear maps between groups. The Weil pairing on elliptic curves is an example of such a map. We give precise definitions for secure identity based encryption [4] schemes and give several applications for such systems.

[3] Title: "A Logic for Auditing Accountability in Decentralized Systems,"
Author: R. Corin, S. Etalle, J.I. den Hartog, G. Lenzini, and I. StaicuProc. IFIP TC1 WG1.7 Workshop Formal Aspects in Security and Trust,pp. 187-201, 2005
Summary: We propose a language that allows agents to distribute data with usage policies in a decentralized architecture. In our framework, the compliance with usage policies is not enforced. However, agents may be audited by an authority at an arbitrary moment in time. We design a logic that allows audited agents to prove their actions, and to prove their authorization to process particular data. Accountability is defined in several flavors, including agent accountability and data accountability. Finally, we show the soundness of the logic.

III. Existing System

Cloud computing provides services rather than a product. Cloud provides access to shared resources, software and information to various devices (such as computers, PDA). A Cloud server handles multiple requests at the same time. The server processing time is high as no of requests are huge. This may lead to corrupt data and may lead to delay in packages. So data management becomes essential. While enjoying the convenience brought by this new technology, users also start bothering about losing control of their own data. The data operated on clouds are often outsourced, which lead to a number of issues related to accountability, including the management of personally identifiable information. So it is necessary to provide an effective mechanism to monitor the usage of the data in cloud. Example, data should be handled according to the service level agreement. Conventional approaches used in database and operating systems or approaches in centralized server has following drawbacks.

Disadvantage: Database management system does not have trustworthiness.

IV. Proposed System

To overcome the above problems, we propose a new method, namely Cloud Information Accountability (CIA) framework[7]. CIA framework is based on information accountability concept. Cloud provides various functionalities such as read write and copy of the original data. Data owner can upload the data into cloud server. The log and log Harmonizer keeps track of the access logs. These reports are later sent to the data owner. Over here, identity based encryption algorithm is applied on data which is uploaded on the cloud.

4.1 Advantages: Data can be shared in a secured manner.

4.2 Modification:

If any unauthorized actions are performed by any user then immediately data owner will be informed by Automatic reporting mechanism. It would generate the random numbers set for every user along with the data owner.

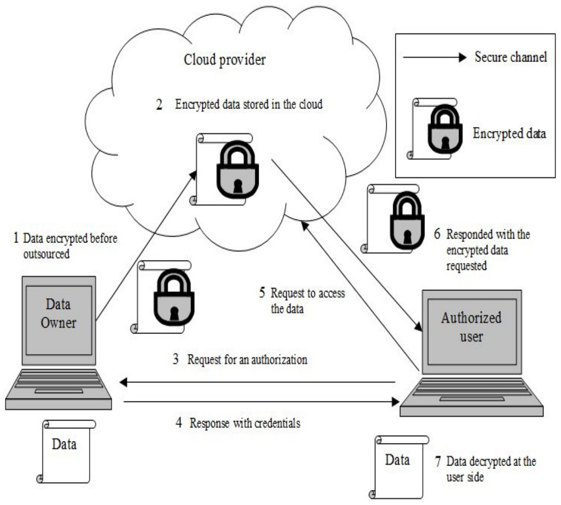
Cloud provider
2 Encrypted data stored in the cloud
Secure channel
Encrypted data
1 Data encrypted before outsourced
6 Responded with the encrypted data requested
5 Request to access the data
Data Owner
Authorized user
3 Request for an authorization
4 Response with credentials
Data
Data
7 Data decrypted at the user side

Fig.1: Overview of the cloud information accountability framework.

So if the user accessing the account, the user has to give the random number set and that will be verified by the server. If the resultant verification is positive then only the users will be allowed to access their account.

V. Implementation

5.1 Encryption Algorithm:

The reason to use encryption is to protect data so that only a specific person or a machine can access it. However, until now, encryption techniques have relied on long, randomly generated keys that must be mapped to identities using digitally-signed documents, called digital certificates – traditional PKI. The management of these certificates and the process by which they are managed, and the need to fetch a certificate before encrypting to a person or machine, has made encryption using traditional approaches very difficult for end users, costly to operate and complex for IT operations.

Identity-Based Encryption (IBE) takes a breakthrough approach to the problem of encryption key management. IBE can use any arbitrary string as a public key, enabling data to be protected without the need for certificates. Protection is provided by a key server that controls the dynamic generation of private decryption keys that correspond to public identities and the key servers base root key material. By separating authentication and authorization from private key generation through the key server, permissions to generate keys can be controlled dynamically on a granular policy driven basis, facilitating granular control over access to information in real time.

Here, consider two users Alice and Bob. When the encryption algorithm needs to be applied on data; Alice will encrypt data with public key. The public key is generated from Bob's identity. Encrypted data is then sent to Bob. At the time of decryption, Bob needs to be authenticated. The authentication is taken at key generation server. Private key for decryption is then sent to Bob by key generator server. By using Bob's private key the data will get decrypted to it's original form. This complete process is illustrated in figure 2.

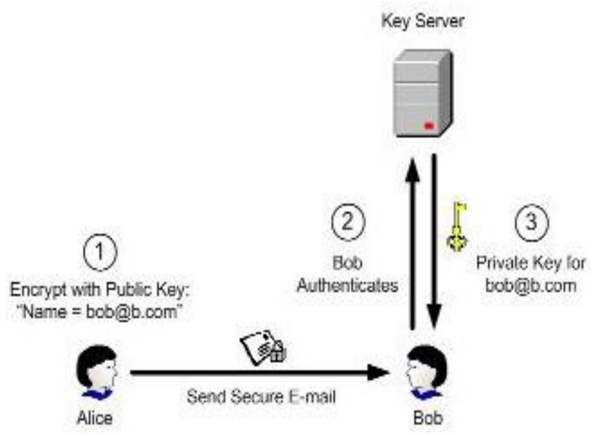
Key Server
1
Encrypt with Public Key: "Name = bob@b.com"
2
Bob Authenticates
3
Private Key for bob@b.com
Send Secure E-mail
Alice
Bob

Fig. 2: Identity Based Algorithm

5.2 Modules:

- User/Data Owner
- Cloud Sever
- Certificate Authority
- Logger
- Access Privileges
- Push And Pull
- Random Set Generation And Verification

5.2.1 User/Data Owner:

User is the person who is going to see or download the data from the Cloud server. Registration is mandatory for accessing the data from cloud server. So first user has to register his/her details like user name, password, email ID. This information will be used for authentication of user and for sending alert to user. Data Owner is the person who is going to upload the data in the Cloud Server. Registration is mandatory for accessing the data from cloud server. After the registration, the space will be assigned to the Data Owner for keeping his data in server.

5.2.2 Cloud Server:

Cloud Server is the area where the data owner will upload their data and the user going to request the data. Once the user send the request regarding the data he/she want, the request will be first send to the Cloud Server and the Cloud Server checks what access user has for data. And according to that action will be performed. (read/write/download) The Cloud Server manage the Data owner and Users information in their Database.

5.2.3 Logger:

The Logger is maintained by the Cloud Server. Loggers have the details of the data owner and users who are accessing the Cloud Server. So the Logger will be more useful for many purposes. Like which user / data owner accessing the Cloud Server, accessed at the particular time and the IP address from which the data is requested by user etc.

5.2.4 Certificate Authority

The Certificate Authority is used to verify the Cloud Server is recognized or not. The Cloud Server has to be recognized by the certificate authority. If not recognized, the Cloud Server is a Fraudulent Server. The data owner can check the whether the recognized or not. Because the data owner is going to upload their data in the Cloud Server.

5.2.5 Access Privileges

The access privileges are set by the data owner for accessing their data. Some Owners will provide read only, some of them will allow read and download. The Cloud Server will send the dynamic intimation when the user is accessing the data beyond their limits. This increases more security while sharing the data in the Cloud.

5.2.6 Push and Pull Mode

To allow users to be timely and accurately informed about their data usage, our distributed logging mechanism is complemented by an innovative auditing mechanism.

5.2.6.1 Push

For the every periodical time the Cloud Server will send the access details of the user to the data owner. So that the Data Owner may able to know who're all the accessing their data at the particular time period. During the registration phase, the Data owner will ask by the Cloud Server whether they're choosing the push or pull method.

5.2.6.2 Pull

In the Pull method, the data owner has to send the request to the Cloud Server regarding the access details of their data up to the particular time. Then the Cloud Server will send the response to the Data Owner regarding the user's access details.

5.2.6.3 Algorithms for Push and Pull Mechanism

The push strategy is used when there are large no access request are coming within a short span of time. This time log file will become very large and cost of copying file will increase so data is not pushed out frequently. The push mode will be used by Data Owner for keeping the track of data usage consistently over time. For such data owner receiving the logs will automatically lighten the size. The pull strategy is used when the data owner suspects some misuse of his content immediately. A hybrid strategy can actually be implemented to benefit the consistent information offered by pushing and pulling mode.

5.2.6.4 Push and Pull Log Mode

Size: the maximum size specified by the data owner
Time: the maximum time allowed to elapse before the log file is dumped.
Tbeg: timestamp at which the last dump occurred
Log: current log file
Pull: indicated whether the command from data owner has is received.

5.2.6.5 The log retrieval algorithm for push and pull mode

First the algorithm checks whether size of JAR has exceeded a stipulated size or the normal time between the two consecutive dumps has elapsed. The size and time threshold for a dump are specified by the data owner at the time of creating JAR. If none of the events has occurred, it proceeds to encrypt the record and write the error correction information to the harmonizer. The communication with harmonizer begins with a simple handshake. If no response is received them log file records an error. The data owner will be alerted through email. The JAR is configured to send error notifications. Our auditing mechanism has two main advantages. First, it gives a high level of availability of the logs. Second, the use of the harmonizer minimizes the amount of workload for human users in going through long log files sent by different copies of JAR files.

5.2.7 Random Set Generation and Verification

When the user request the data to be downloaded from the Cloud Server, the user have to enter the Random number set. If it is matched, the user is allowed to download the data. The Random number sets will be provide to the user during the registration Phase itself. Each and Every time the Random number set will vary. This ensures security while downloading the data.

VI. Experiment and Result

6.1 Experimental Settings

We tested our CIA framework by setting up a small cloud. We observed the below mentioned things: The attacker typically cannot access it as Log Harmonizer component is saved separately in either a secure proxy or at the user end. As a result, the attacker cannot extract the decryption keys from the log harmonizer.

6.2 Experimental Results

In the experiments, we first check log file creation time and then measure the overhead to the system. Overhead mainly occurs at three points:
With respect to time:

1. At the time of Authentication.
2. During encryption of a log record.
3. At the time of the merging of the logs.

With respect to storage overhead:

1. We observed that this architecture is very lightweight
2. Actual file and associated logs can only provide data which needs to be stored.
3. JAR works as a compressor for files.
4. Same logger component can manage multiple files.
5. We have checked whether single logger component is used to manage multiple files and result is stored in database.

6.2.1 Log Creation Time

We have concerned in finding out the time taken to create a log file when there are entities continuously accessing the data. Result is shown in below figure (Figure 3.) We identified that time to create a log file linearly increases with respect to size of log file. With this experiment, one can figure out the amount of time to be specified between dumps, keeping other variables like space constraints or network traffic in mind.

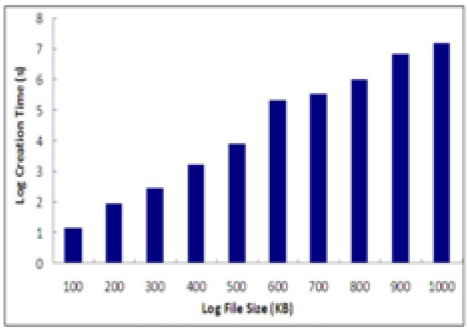

Fig.3: Time to merge log files

6.2.2 Authentication Time

Authentication of a CSP is a main part. If the time taken for authentication is too much then it will effect on the time for data accessing. We focused to reduce the time for authentication.

6.2.3 Time taken to Perform Logging

We have measured the average time taken to allow an access plus the time to write the corresponding log record. The time for allowing any access to the data items in a JAR file includes the time to evaluate and enforce the applicable policies and to locate the requested data items. In the experiment, we let multiple servers continuously access the same data JAR file for a minute and recorded the number of log records generated. Every access is just a view request and hence the time for executing the action is negligible. As a resultant, the average time to log an action is less, which involves the time taken by a user to double click the JAR or by a server to run the script to open the JAR. We also took the log encryption time and seemingly unrelated from the log size.

6.2.4 Size of the Data JAR Files

Data storage increases if a single logger is trying to handle more than one file. We have also checked the size of the loggers (JARs) by varying the number and size of data items held by them.

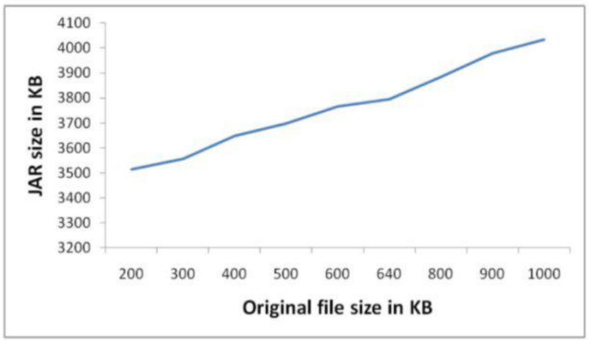

Fig.4: Size of the logger component

6.2.5 Overhead Added by JVM Integrity Checking

We have also investigated the overhead added by both the reinstallation/repair process, and by the time taken for computation of hash codes. This time was figured by taking the system time stamp at the beginning and end of the installation/repair. To calculate the time overhead added by the hash codes, we simply measure the time acquired for each hash calculation. The number of hash commands varies on the basis of size of the code in the code does not change with the content, the number of hash commands remain constant.

VII. Conclusion and Future Research

We introduced new approaches for automatically logging with an auditing mechanism. Using our approach Data Owner will be able to audit his contents. Strong Protection is provided to data in cloud. We have introduced a new methodology to enhance integrity for data in cloud.

In future, we plan to refine our approach to verify the integrity of JRE and authentication mechanism by JAR. We are planning to design a comprehensive and more simplified object oriented approach which will facilitate autonomous protection for contents. We will be able to support a variety of security policies, like indexing policies for text files, usage control for executables, and generic accountability and provenance controls

Acknowledgement

I take this opportunity to express our hearty thanks to all those who have helped me in the completion of this Paper. I would like to express our sincere gratis especially to guide **Prof. Mr.B.B.Gite** sir who have been source of motivation for completion of this Paper. I would also like to thank principal of our college **Dr.Avinash Kharat** for their kind support. I am very grateful to our staff for this co-operation and the authors of various internets, helping me become aware of net search currently going in the field. Last, I would like to thank my friends for their suggestions.

References

[1]. P. Ammann and S. Jajodia, “Distributed Timestamp Generation in Planar Lattice Networks,” ACM Trans. Computer Systems, vol. 11,pp. 205-225, Aug. 1993.
[2]. 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.
[3]. E. Barka and A. Lakas, “Integrating Usage Control with SIP-Based Communications,” J. Computer Systems, Networks, and Comm.,vol. 2008,pp. 1-8, 2008.
[4]. 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.
[5]. R. Bose and J. Frew, “Lineage Retrieval for Scientific Data Processing: A Survey,” ACM Computing Surveys, vol. 37, pp. 1-28, Mar. 2005.
[6]. 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
[7]. B. Chun and A.C. Bavier, “Decentralized Trust Management and Accountability in Federated Systems,” Proc. Ann. Hawaii Int’l Conf.System Sciences (HICSS), 2004.
[8]. 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.
[9]. B. Crispo and G. Ruffo, “Reasoning about Accountability within Delegation,” Proc. Third Int’l Conf. Information and Comm. Security (ICICS), pp. 251-260, 2001.
[10]. Y. Chen et al., “Oblivious Hashing: A Stealthy Software Integrity Verification Primitive,” Proc. Int’l Workshop InformationHiding, F. Petitcolas, ed., pp. 400-414, 2003.
[11]. P.T. Jaeger, J. Lin, and J.M. Grimes, “Cloud Computing and Information Policy: Computing in a Policy Cloud?,” Information Technology and Politics, vol. 5, no. 3, pp. 269-283,2009.
[12]. 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.