

# Fortify Apportion Liability for Data Sharing In Cloud Computing

**Syed Mubashirul Huda**

Nimra College of Engineering and Technology.

**G. preeti Jyotsna**

Nimra College of Engineering and Technology.

## ABSTRACT:

Cloud computing enables highly scalable services to be easily consumed over the Internet on an as-needed basis a major feature of the cloud services is that users' data are usually processed remotely in unknown machines that users do not own or operate.

While enjoying the convenience brought by this new emerging technology, users' fears of losing control of their own data (particularly, financial and health data) can become a significant barrier to the wide adoption of cloud services.

To address this problem, here, we propose a novel highly decentralized information accountability framework to keep track of the actual usage of the users' data in the cloud. In particular, we propose an object-centered approach that enables enclosing our logging mechanism together with users' data and policies.

We leverage the JAR programmable capabilities to both create a dynamic and traveling object, and to ensure that any access to users' data will trigger authentication and automated logging local to the JARs. To strengthen user's control, we also provide distributed auditing mechanisms. We provide extensive experimental studies that demonstrate the efficiency and effectiveness of the proposed approaches.

## 1. INTRODUCTION:

Cloud computing presents a new way to supplement the current consumption and delivery model for IT services based on the Internet, by providing for dynamically scalable and often virtualized resources as a service over the Internet. To date, there are a number of notable commercial and individual cloud computing services, including Amazon, Google, Microsoft, Yahoo, and Sales force.

Details of the services provided are abstracted from the users who no longer need to be experts of technology infrastructure. Moreover, users may not know the machines which actually process and host their data. While enjoying the convenience brought by this new technology, users also start worrying about losing control of their own data.

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.

Cloud computing enables highly scalable services to be easily consumed over the Internet on an as-needed basis a major feature of the cloud services is that users' data are usually processed remotely in unknown machines that users do not own or operate. While enjoying the convenience brought by this new emerging technology, users' fears of losing control of their own data (particularly, financial and health data) can become a significant barrier to the wide adoption of cloud services.

To address this problem, here, we propose a novel highly decentralized information accountability framework to keep track of the actual usage of the users' data in the cloud. In particular, we propose an object-centered approach that enables enclosing our logging mechanism together with users' data and policies.

We leverage the JAR programmable capabilities to both create a dynamic and traveling object, and to ensure that any access to users' data will trigger authentication and automated logging local to the JARs. To strengthen user's control, we also provide distributed auditing mechanisms. We provide extensive experimental studies that demonstrate the efficiency and effectiveness of the proposed approaches.

## 2) FEASIBILITY STUDY:

### A) Economic feasibility study:

This involves questions such as whether the firm can afford to build the system, whether its benefits should substantially exceed its costs, and whether the project has higher priority and profits than other projects that might use the same resources. This also includes that whether the project is in the condition to fulfill all the eligibility criteria and the responsibility of both sides in case there are two parties involved in performing any project.

### B) Technical feasibility study:

This involves questions such as whether the technology needed for the system exists, how difficult it will be to build, and whether the firm has enough experience using that technology. The assessment is based on an outline design of system requirements in terms of Input, Output, Fields, Programs, and Procedures. This can be qualified in terms of volumes of data, trends, frequency of updating, etc. in order to give an introduction of technical system.

### C) Schedule Feasibility study:

This involves questions such as how much time is available to build the new system, when it can be built (i.e. during holidays), interference with normal business operation, etc.

### D) Organizational Feasibility study:

This involves questions such as whether the system has enough support to be implemented successfully, whether it brings an excessive amount of change, and whether the organization is changing too rapidly to absorb it.

### E) Cultural Feasibility study:

In this stage, the project's alternatives are evaluated for their impact on the local and general culture. For example, environmental factors need to be considered.

### F) Legal Feasibility study:

Not necessarily last, but all projects must face legal scrutiny. When an organization either has legal council on staff or on retainer, such reviews are typically standard. However, any project may face legal issues after completion too.

### G) Marketing Feasibility study:

This will include analysis of single and multi-dimensional market forces that could affect the commercialization or success of a project's actual revenue (sales) potential

## 3) SOFTWARE TECHNIQUES:

### A) Object Oriented Analysis And Design:

Object-oriented analysis and design (OAD) is often part of the development of large scale systems and programs often using the Unified Modeling Language (UML). OAD applies object-modeling techniques to analyze the requirements for a context — for example, a system, a set of system modules, an organization, or a business unit — and to design a solution. Most modern object-oriented analysis and design methodologies are use case driven across requirements, design, implementation, testing, and deployment.

Use cases were invented with object oriented programming, but they're also very well suited for systems that will be implemented in the procedural paradigm. The Unified Modeling Language (UML) has become the standard modeling language used in object-oriented analysis and design to graphically illustrate system concepts. Part of the reason for OAD is its use in developing programs that will have an extended lifetime.

### B) Object Oriented Systems:

An object-oriented system is composed of objects. The behavior of the system is achieved through collaboration between these objects, and the state of the system is the combined state of all the objects in it. Collaboration between objects involves those sending messages to each other.

The exact semantics of message sending between objects varies depending on what kind of system is being modeled. In some systems, “sending a message” is the same as “invoking a method”. In other systems, “sending a message” might involve sending data via a socket.

## C) Object Oriented Analysis:

Object-Oriented Analysis (OOA) aims to model the problem domain, the problem we want to solve by developing an object-oriented (OO) system. The source of the analysis is a written requirement statements, and/or written use cases, UML diagrams can be used to illustrate the statements. An analysis model will not take into account implementation constraints, such as concurrency, distribution, persistence, or inheritance, nor how the system will be built.

The model of a system can be divided into multiple domains each of which are separately analyzed, and represent separate business, technological, or conceptual areas of interest. The result of object-oriented analysis is a description of what is to be built, using concepts and relationships between concepts, often expressed as a conceptual model. Any other documentation that is needed to describe what is to be built is also included in the result of the analysis. That can include a detailed user interface mock-up document. The implementation constraints are decided during the object-oriented design (OOD) process.

## D) Object Oriented Design:

Object-Oriented Design (OOD) is an activity where the designers are looking for logical solutions to solve a problem, using objects. Object-oriented design takes the conceptual model that is the result of object-oriented analysis, and adds implementation constraints imposed by the environment, the programming language and the chosen tools, as well as architectural assumptions chosen as basis of design. The concepts in the conceptual model are mapped to concrete classes, to abstract interfaces in APIs and to roles that the objects take in various situations. The interfaces and their implementations for stable concepts can be made available as reusable services. Concepts identified as unstable in object-oriented analysis will form basis for policy classes that make decisions, implement environment-specific or situation specific logic or algorithms.

The result of the object-oriented design is a detail description how the system can be built, using objects. Object-oriented software engineering (OOSE) is an object modeling language and methodology. OOSE was developed by Ivar Jacobson in 1992 while at Objectory AB. It is the first object-oriented design methodology to employ use cases to drive software design. It also uses other design products similar to those used by OMT. The tool Objectory was created by the team at Objectory AB to implement the OOSE methodology. After success in the marketplace, other tool vendors also supported OOSE. After Rational bought Objectory AB, the OOSE notation, methodology, and tools became superseded.

- As one of the primary sources of the Unified Modeling Language (UML), concepts and notation from OOSE have been incorporated into UML.
- The methodology part of OOSE has since evolved into the Rational Unified Process (RUP).
- The OOSE tools have been replaced by tools supporting UML and RUP.

OOSE has been largely replaced by the UML notation and by the RUP methodology.

## 4) INPUT AND OUTPUT DESIGN:

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- » What data should be given as input?
- » How the data should be arranged or coded?
- » The dialog to guide the operating personnel in providing input.
- » Methods for preparing input validations and steps to follow when error occur.



1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

## OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

- Confirm an action.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the

- Future.

- Signal important events, opportunities, problems, or warnings.

- Trigger an action.

## 5) MODULE IMPLEMENTATION

### A) Cloud Information Accountability (CIA) Framework:

CIA framework lies in its ability of maintaining light-weight and powerful accountability that combines aspects of access control, usage control and authentication. By means of the CIA, 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.

### B) Distinct mode for auditing: Push mode:

The push mode refers to logs being periodically sent to the data owner or stakeholder. Pull mode: Pull mode refers to an alternative approach whereby the user (Or another authorized party) can retrieve the logs as needed.

### C) Logging and auditing Techniques:

1. The logging should be decentralized in order to adapt to the dynamic nature of the cloud. More specifically, log files should be tightly bounded with the corresponding data being controlled, and require minimal infrastructural support from any server.

2. Every access to the user's data should be correctly and automatically logged. This requires integrated techniques to authenticate the entity that accesses the data, verify, and record the actual operations on the data as well as the time that the data have been accessed.

3. Log files should be reliable and tamper proof to avoid illegal insertion, deletion, and modification by malicious parties. Recovery mechanisms are also desirable to restore damaged log files caused by technical problems.

4. 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 anytime by their data owners when needed regardless the location where the files are stored.

5. The proposed technique should not intrusively monitor data recipients' systems, nor it should introduce heavy communication and computation overhead, which otherwise will hinder its feasibility and adoption in practice.

### D) Major components of CIA:

There are two major components of the CIA, the first being the logger, and the second being the log harmonizer. The logger is strongly coupled with user's data (either single or multiple data items). 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. It may also be configured to ensure that access and usage control policies associated with the data are honored. For example, a data owner can specify that user X is only allowed to view but not to modify the data. The logger will control the data access even after it is downloaded by user X. The log harmonizer forms the central component which allows the user access to the log files. The log harmonizer is responsible for auditing.

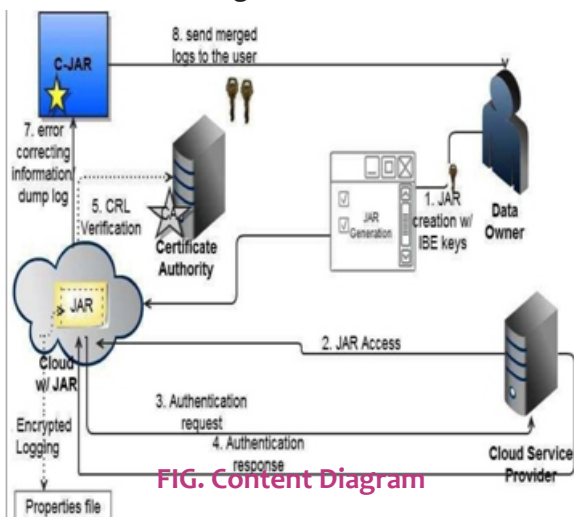
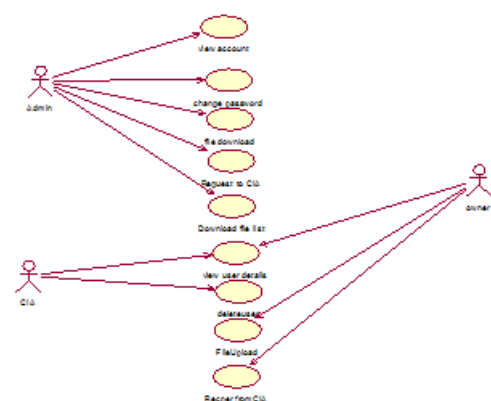


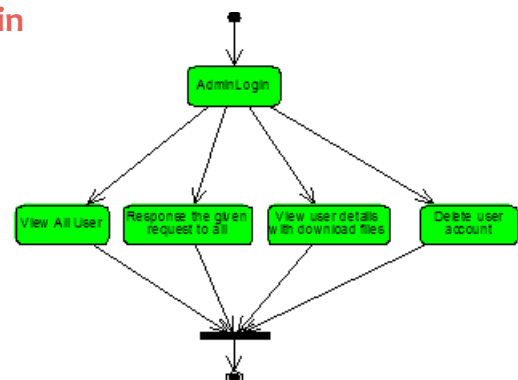
FIG. Content Diagram

Cloud computing enables highly scalable services to be easily consumed over the Internet on an as-needed basis. A major feature of the cloud services is that users' data are usually processed remotely in unknown machines that users do not own or operate. While enjoying the convenience brought by this new emerging technology, users' fears of losing control of their own data (particularly, financial and health data) can become a significant barrier to the wide adoption of cloud services. To address this problem, here, we propose a novel highly decentralized information accountability framework to keep track of the actual usage of the users' data in the cloud. In particular, we propose an object-centered approach that enables enclosing our logging mechanism together with users' data and policies. We leverage the JAR programmable capabilities to both create a dynamic and traveling object, and to ensure that any access to users' data will trigger authentication and automated logging local to the JARs. To strengthen user's control, we also provide distributed auditing mechanisms. We provide extensive experimental studies that demonstrate the efficiency and effectiveness of the proposed approaches.

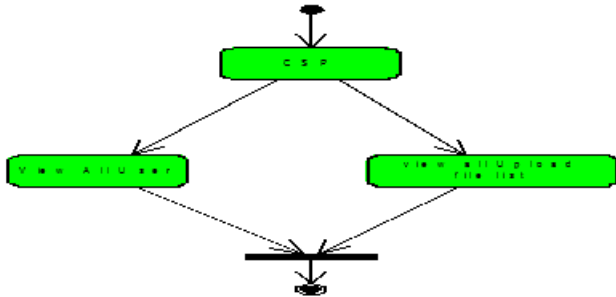
### Use Case Diagram:



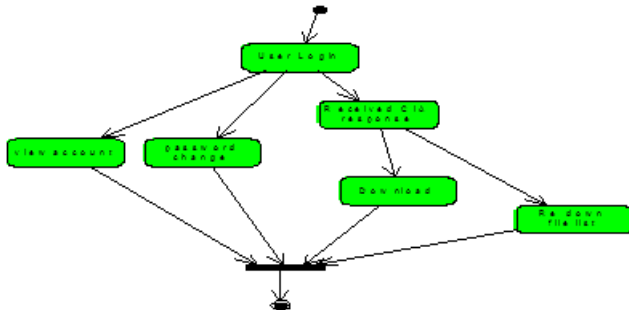
### Admin



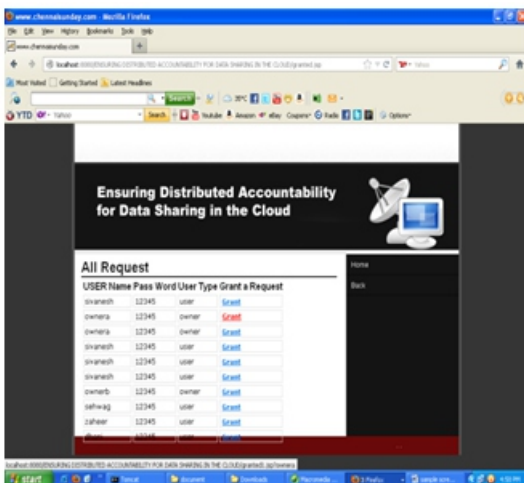
## CSP



## User



TO Grant Permission to upload file for owner and download and view the file for user ... Sending CIA Request Password



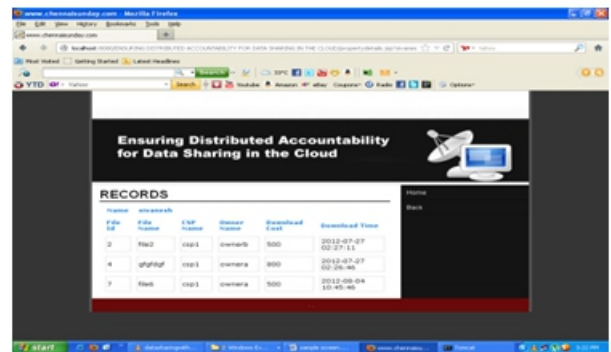
USER Name	Pass Word	User Type	Grant a Request
skanesh	12345	user	Grant
ownera	12345	owner	Grant
ownera	12345	owner	Grant
skanesh	12345	user	Grant
skanesh	12345	user	Grant
skanesh	12345	user	Grant
ownarb	12345	owner	Grant
ghafag	12345	user	Grant
zafhar	12345	user	Grant

## User list Page:



USER ID	USER NAME	EMAIL ID	DOWNLOAD PROPERTY
1	skanesh	skanesh@gmail.com	PROPERX22
13	hara	hara@gmail.com	PROPERX22

## User Property page:



ID	NAME	CSP	Owner	Download Cost	Download Time
2	hac2	cap1	ownarb	5000	2012-07-07 00:27:11
4	ghafag	cap1	ownarb	8000	2012-07-07 00:28:40
7	hac2	cap1	ownarb	5000	2012-08-04 00:40:00

## View ALL User & Owner & CSP Details:



Name	skaneshkan
User Name	skanesh
Email ID	skanesh001@gmail.com
Date of Birth	07/05/2012
Address	Sanjivani, Chennai
Mobile Number	9122475005
Country	India
Account Number	XXXXXXXXXXXX
Name	Muthuraman
User Name	Muthu

## Delete User Page:

If you Click delete all then All the user and owner and csp all will be removed from database.



DELETE THE RECORDS
DELETE ALL

## 6. CONCLUSION:

We proposed innovative approaches for automatically logging any access to the data in the cloud together with an auditing mechanism. Our approach allows the data owner to not only audit his content but also enforce strong back-end protection if needed. Moreover, one of the main features of our work is that it enables the data owner to audit even those copies of its data that were made without his knowledge.

## 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. OASIS Security Services Technical Committee, "Security Assertion Markup Language (saml) 2.0," .
9. 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.
10. B. Crispo and G. Ruffo, "Reasoning about Accountability within Delegation," Proc. Third Int'l Conf. Information and Comm. Security (ICICS), pp. 251-260, 2001.