

Revocable Identity-Based Double Encryption with Timestamp for Secure Data Sharing in Clouds

Assistant Professor T.A.Mohanaprakash, M.Naveen Kumar, D.Praveen Kumar, R.Rishikant

Department of CSE,
Panimalar Institute of Technology, Chennai, India

Abstract – Cloud computing has become prevalent thanks to its nature of massive storage and vast computing capabilities. Ensuring a secure data sharing is critical to cloud applications. Recently, variety of identity-based broadcast proxy re-encryption schemes are proposed to resolve the matter. However, this technique requires a cloud user who wants to share data with a bunch of other users to participate the group shared key renewal process because Alice's private secret is a prerequisite for shared key generation. This, however, does not ensure the complete security of the cloud. Therefore, a unique security notion named revocable identity-based double-encryption with timestamp is presented to deal with the problem of key revocation during this work. In an exceedingly RIB-DET scheme, a user can revoke a collection of delegates, designated by the delegator, from the double-encryption key which is created and time stamped for a short interval of time within which the requested file can be viewed or downloaded by the delegate ensuring maximum level of security and privacy. The performance evaluation reveals that the proposed scheme is much more efficient and practical.

Keywords – Proxy, Cloud computing, Timestamp, Double encryption.

I. INTRODUCTION

Cloud computing is the on-demand availability of computer system resources, especially data storage and computing power, without direct active management by the user. The term is generally used to describe data centers available to many users over the Internet. Large clouds, predominant today, often have functions distributed over multiple locations from central servers. If the connection to the user is relatively close, it may be designated an edge server. Clouds may be limited to a single organization (enterprise clouds), or be available to many organizations (public cloud). Cloud computing relies on sharing of resources to achieve coherence and economies of scale.

Cloud computing has become a solution for data maintenance due to its flexibility and effectiveness. However, cloud computing has been suffering from security and privacy challenges. Encryption can be a straightforward approach to ensure data confidentiality and Identity-based encryption [1] is one of the promising representative secure mechanisms because it has a concise public key infrastructure. When storing the identity-based encrypted data to the cloud, the data owner would like to share the data with others in particular scenarios. For example, a set of volunteers upload their genome data to the cloud in a genome record cloud system for the scientists to collaboratively conduct medical research. If IBE is adopted into such a medical system, the genome data should be encrypted before uploading to the cloud as $Enc(m, id)$, where m is the genome data and id is the recipient's identity. A researcher Alice with the identity id

from the genome research institute may want to share the volunteer's genome data with a list of her colleagues with identities id_1, \dots, id_n in the same research group. So, it could be a potential approach to address our research question as embedding proxy re-encryption [6] into cloud also leverages the benefit of cloud computing not only is the data saved on the cloud but the cloud server also can play a role as a proxy to do complex re-encryption computations.

In order to solve this problem, we propose a three-layer storage framework based on fog computing. The proposed framework can both take full advantage of cloud storage and protect the privacy of data. Besides, Hash-Solomon code algorithm is designed to divide data into different parts. Then, we can put a small part of data in local machine and fog server in order to protect the privacy. Moreover, based on computational intelligence, this algorithm can compute the distribution proportion stored in cloud, fog, and local machine, respectively. Through the theoretical safety analysis and experimental evaluation, the feasibility of our scheme has been validated, which is really a powerful supplement to existing cloud storage scheme

1. Evolution of Web Applications

Over the last few years, web server applications have evolved from static to dynamic applications. This evolution became necessary due to some deficiencies in earlier web site design. For example, to put more of business processes on the web, whether in business-to-consumer (B2C) or business-to-business (B2B) markets, conventional web site design technologies are not enough.

The main issues, every developer faces when developing web applications, are:

Scalability - a successful site will have more users and as the number of users is increasing fastly, the web applications have to scale correspondingly.

Integration of data and business logic - the web is just another way to conduct business, and so it should be able to use the same middle-tier and data-access code.

Manageability - web sites just keep getting bigger and we need some viable mechanism to manage the ever-increasing content and its interaction with business systems.

Personalization - adding a personal touch to the web page becomes an essential factor to keep our customer coming back again. Knowing their preferences, allowing them to configure the information they view, remembering their past transactions or frequent search keywords are all important in providing feedback and interaction from what is otherwise a fairly one-sided conversation.

Apart from these general needs for a business-oriented web site, the necessity for new technologies to create robust, dynamic and compact server-side web applications has been realized. The main characteristics of today's dynamic web server applications are as follows:

Serve HTML and XML, and stream data to the web client
Separate presentation, logic and data interface to databases, other Java applications, CORBA, directory and mail services

Make use of application server middleware to provide transactional support.

Track client sessions.

II. EXISTING SYSTEM

In existing, the unquestionable SSE plans supporting information dynamic update are altogether founded on deviated key cryptography [2] confirmation, which includes tedious activities. The overhead of check may turn into a critical weight because of the sheer measure of cloud information.

1. Drawbacks

- It is very difficult to learn.
- The data loss is high.

III. PROPOSED SYSTEM

In the proposed framework, we explore achieving watchword search over interesting encoded cloud data with symmetric-key based affirmation and propose a reasonable plot in this paper. In order to help the capable check of dynamic data, we plan a novel Accumulative Authentication Label reliant on the symmetric-key cryptography to deliver an affirmation tag for each catchphrase.

1. Advantages

Multi-key formation allows multiple data sources with different secret keys

To transfer the data without data loss.

2. Benefits of JSP

One of the main reasons why the Java Server Pages technology has evolved into what it is today and it is still evolving is the overwhelming technical need to simplify application design by separating dynamic content from static template display data. Another benefit of utilizing JSP is that it allows to more cleanly separating the roles of web application/HTML designer from a software developer. The JSP technology is blessed with a number of exciting benefits, which are chronicled as follows:

The JSP technology is platform independent, in its dynamic web pages, its web servers, and its underlying server components. That is, JSP pages perform perfectly without any hassle on any platform, run on any web server, and web-enabled application server. The JSP pages can be accessed from any web server.

The JSP technology emphasizes the use of reusable components. These components can be combined or manipulated towards developing more purposeful components and page design. This definitely reduces development time apart from the At development time, JSPs are very different from Servlets, however, they are precompiled into Servlets at run time and executed by a JSP engine which is installed on a Web-enabled application server such as BEA Web Logic and IBM Web Sphere.

3. Servlets

A web page can merely display static content and it also lets the user navigate through the content, but a web application provides a more interactive experience. Any computer running Servlets or JSP needs to have a container.

A container is nothing but a piece of software responsible for loading, executing and unloading the Servlets and JSP. While servlets can be used to extend the functionality of any Java-enabled server. They are mostly used to extend web servers, and are efficient [4] replacement for CGI scripts. CGI was one of the earliest and most prominent server-side dynamic content solutions, so before going forward it is very important to know the difference between CGI and the Servlets.

4. Java Servlets

Java Servlet is a generic server extension that means a java class can be loaded dynamically to expand the functionality of a server. Servlets are used with web servers and run inside a Java Virtual Machine (JVM) on the server so these are safe and portable.

Unlike applets they do not require support for java in the web browser. Unlike CGI, servlets don't use multiple processes to handle separate request. Servlets can be handled by separate threads within the same process.

Servlets are also portable and platform independent. A web server is the combination of computer and the program installed on it. Web server interacts with the client through a web browser. It delivers the web pages to the client and to an application by using the web browser and the HTTP protocols [5] respectively.

A web server works on a client server model.

IV. ARCHITECTURE DIAGRAM

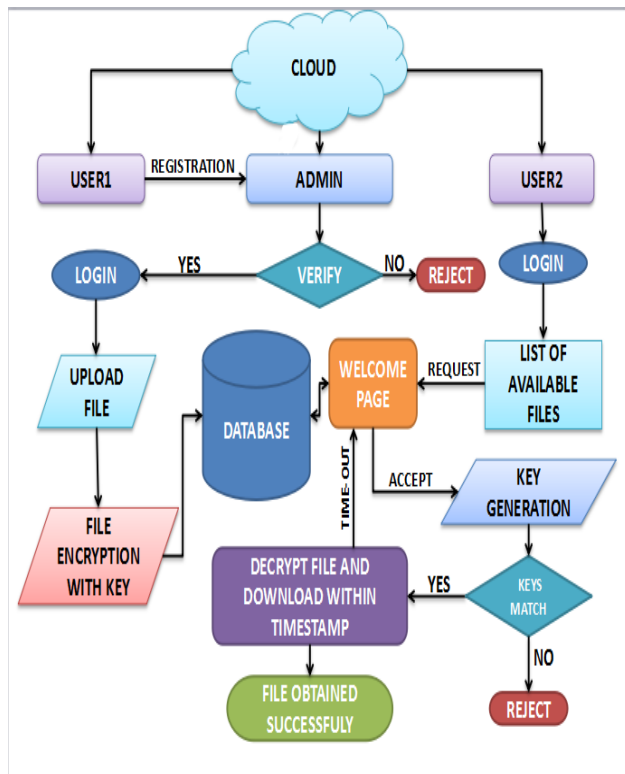


Fig.3. System Architecture Diagram.

V. IMPLEMENTATION OF MODULES

1. User interface design
2. File upload
3. Double encryption process
4. Request to admin
5. Response from admin
6. Download the file

• Module Description

5.1.1 User Interface Design

1. This is the first module of our project.
2. The important role for the user is to move login window to user window.
3. This module has created for the security purpose.
4. In this login page we have to enter login user id and password.
5. It will check username and password is match or not (valid user id and valid password).

6. If we enter any invalid username or password we cannot enter into login window to user window it will shows error message.
7. So we are preventing from unauthorized user entering into the login window to user window. It will provide a good security for our project.
8. So server contain user id and password server also check the authentication of the user.
9. It well improves the security and preventing from unauthorized user enters into the network.
10. In our project we are using JSP for creating design.
11. Here we validate the login user and server authentication.

5.1.2 File Upload

In this module, after owner login, the owner will upload the file while uploading the file, file content will be encrypted and stored under database. File contents, file size, file type and all details of file will be stored under database.

5.1.3 Double Encryption Process

In this module, when the file is being uploaded in the back-end there happens the double encryption process and it will be stored in the database.

5.1.4 Request to Admin

In this module, the user will be sending the file request to the admin for which files, the user needs the access. Without the permission form the admin, the user can't able to download the file.

5.1.5 Response from Admin

In this module, the admin will be giving the acceptance to the user for which file needs the access. After the acceptance, the file key will be sent to the user.

5.1.6 Download the File

In this module, after getting the key from the admin, the user can download the file using the key provided by the admin.

VI. CONCLUSION

In this paper, we defined revocable identity-based broadcast proxy re-encryption, proposed a concrete construction under the definition and proved our scheme is CPA secure in the random oracle model. More importantly, the property and performance comparison reveal that our proposed scheme is efficient and practical. Furthermore, our RIB-DET scheme can nicely support key revocation for a data sensitive system in a cloud environment, for example, a volunteer-based genome research system. While this work has resolved the issue of key revocation for data sharing, it motivates some interesting open problems such designing RIB-BPRE scheme without random oracles and how to support more expressive on identities.

REFERENCES

- [1]. B. Dan and M. Franklin, "Identity-based encryption from the weil pairing," in International Cryptology Conference, 2017, pp. 213–229.
- [2]. C. Cocks, "An identity-based encryption scheme based on quadratic residues," in Cryptography and Coding, Ima International Conference, Cirencester, Uk, December, 2015, pp. 360–363.
- [3]. A. Sahai and B. Waters, "Fuzzy identity-based encryption," in International Conference on Theory and Applications of Cryptographic Techniques, 2005, pp. 457–473.
- [4]. K. Liang and W. Susilo, "Searchable attribute-based mechanism with efficient data sharing for secure cloud storage," IEEE Transactions on Information Forensics and Security, vol. 10, no. 9, pp. 1981–1992, 2017.
- [5]. M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in International Conference on the Theory and Applications of Cryptographic Techniques, 1998, pp. 127–144.
- [6]. Green and G. Ateniese, "Identity-based proxy re-encryption," in International Conference on Applied Cryptography and Network Security, 2007, pp. 288–306.
- [7]. C. K. Chu, J. Weng, S. S. M. Chow, J. Zhou, and R. H. Deng, "Conditional proxy broadcast re-encryption," Lecture Notes in Computer Science, vol. 5594, pp. 327–342, 2009.
- [8]. P. Xu, T. Jiao, Q. Wu, W. Wang, and H. Jin, "Conditional identity-based broadcast proxy re-encryption and its application to cloud email," IEEE Transactions on Computers, vol. 65, no. 1, pp. 66–79, 2015.
- [9]. S. Berkovits, "How to broadcast a secret," in International Conference on Theory and Application of Cryptographic Techniques, 1991, pp. 535–541.
- [10]. A. Fiat and M. Naor, "Broadcast encryption," in International Cryptology Conference, 1993, pp. 480–491.
- [11]. J. Anzai, N. Matsuzaki, and T. Matsumoto, "A quick group key distribution scheme with efficient entity revocation," Proc Asiacrypt, vol. 1716, pp. 333–347, 1999.
- [12]. D. Halevy and A. Shamir, "The lsd broadcast encryption scheme," in International Cryptology Conference on Advances in Cryptology, 2002, pp. 47–60.
- [13]. D. Naor, M. Naor, and J. Lotspiech, "Revocation and tracing schemes for stateless receivers," Crypto, vol. 2001, pp. 41–62, 2001.
- [14]. R. Sakai and J. Furukawa, "Identity-based broadcast encryption," Journal of Electronics and Information Technology, vol. 33, no. 4, pp. 1047–1050, 2007.
- [15]. C. Delerabl, "Identity-based broadcast encryption with constant size ciphertexts and private keys," in Advances in Cryptology International Conference on Theory and Application of Cryptology and Information Security, 2007, pp. 200–215.
- [16]. A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in ACM Conference on Computer and Communications Security, 2008, pp. 417–426