Indian Journal of Science and Technology, *Vol 8(S7), 674-678, April 2015*

ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645
DOI: 10.17485/ijst/2015/v8iS7/70471

A study on Data Integrity and Consistency Guarantee in Cloud Storage for Collaboration

Jae-Young, Lee*

Department of Liberal Education, Semyung University, Korea; klitie@semyung.ac.kr

Abstract

Cloud Storage for Collaboration is the storage for specific users who participate in their jobs. Data stored on the common storage, which shares with the co-workers, is more important than personal data stored on the storage of one person in matters of data integrity and consistency guarantee. This study suggests how to ensure logical consistency of data, user authentication and how to sustain data integrity when various users try to use the common data simultaneously in Cloud Storage for collaboration.

Keywords: Cloud Storage, Component, Concurrency Control, Collaboration Data Consistency, Integrity, User Authentication

1. Introduction

Cloud Computing provides services combining diffuse computing resources with virtualization technology by using network. One of the services is IaaS (Infrastructure as a Service) that offers hardware like computing power, storage or database and others. Cloud storage, one of IaaS, is 'online virtual network storage space'. That's why people can work in any place where internet is available and with any device they use, if people save data in Cloud storage.

Cloud storage, which makes the participants to work together by saving and sharing data, is suitable for current trends that the number of people who work together at a long distance is rising steadily because of the spread of wireless internet and various mobile devices.

Cloud storage for Collaboration, which shares the storages with many people, needs a method to secure the data and there has to be the other method to protect the data in a totally different way.

This study proposes how to sustain data integrity protecting data from unauthorized users whose malicious hackings approach the data, and how to maintain logical data consistency by using Concurrency Control when many people try to access the one data at the same time. Moreover, I'll make a suggestion about how to certify the users of Cloud storage.

The summary of this study is as follows. In the chapter 2, I will explain a research about the suggested method, and I will offer how to ensure Data integrity, logical data consistency and user authentication in the chapter 3. In the chapter 4, I'll analyze stability of the suggested method. Finally, in the chapter 5, there will be conclusion and future researches.

2. The Related Research

2.1 Cloud storage

Cloud Computing is a service which combines the dispersed computing resources with virtualization technology by using networks. 'Cloud' is virtual worlds, the resources are combined, and the users get the service through 'Cloud'. 'Cloud' provides three virtual services. First of all, it is Software as a Service (SaaS), gives us application software that the users need. Second, it is Platform as a Service (PaaS), which offers platform, development environment, tools and others that are worked by the application. Third, it is Infrastructure as a Service (IaaS)

*Author for correspondence

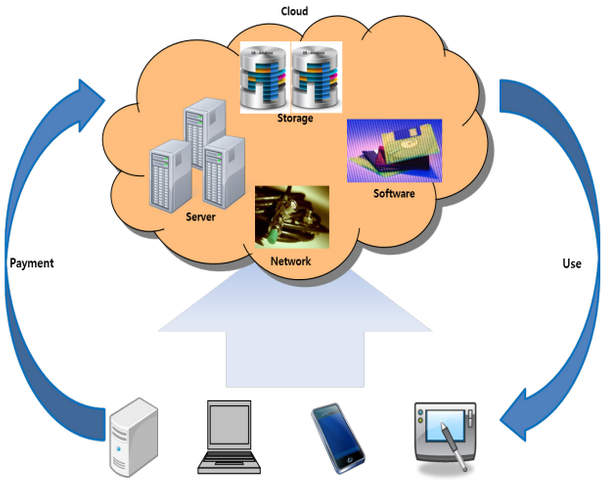

Figure 1. Cloud Computing.

that provides hardware like computing ability, storage or data base, not the vague service which is served by PaaS[1–3].

One of IaaS 'Cloud storage', is provided by Cloud, is virtual network storage space. That's why people can work in any place where internet is available and with any device they use, if people save data in Cloud storage.

2.2 Public-key Encryption System

Public-key encryption system is the asymmetrical encryption method that is composed of one key for code and the other key for decoding. These two keys are mathematically connected each other but they have a completely different value. One of the keys is owned by the key's user and the other key, so-called public key, is kept in trusted public certification agencies.

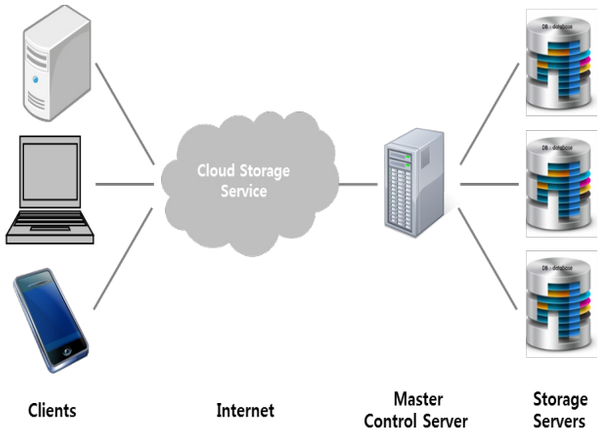

Figure 2. Cloud Storage.

If user 1 sends a secret massage to user2, user1 will use the public key of user 2 and then send the encrypted message to user 2. The message which is encrypted by public key can be deciphered only by the paired private key. This is a way to keep confidentiality about the message.

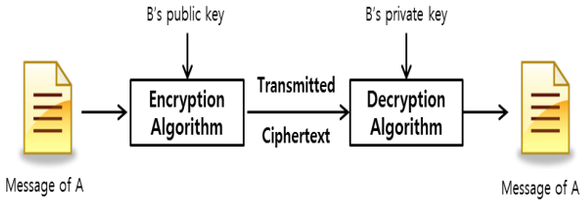

Figure 3. Send message to B.

If user 2 does digital signature to certify the message which is sent by user 1, user 2 will send the encrypted message through his private key and user 2 confirm about the message, which is deciphered by the public key of user1. If deciphered message is certified, user1, who have the used public key and the paired private key, can ascertain whether it is the transmitted message.

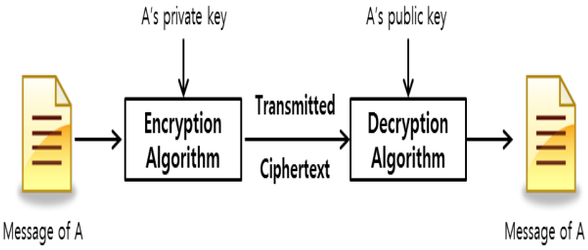

Figure 4. Signed statement of A.

Public-key encryption system uses the private key and the public key, which is mathematically connected, has no problem, unlike symmetric cipher method which uses only one key for encryption and decoding[6,7].

2.3 Concurrency Control

Concurrency Control handles potential problems, when over two users try to access one data. Concurrency Control has a feature which demands data consistency and quick reaction time.

When people use one of the storages as a public storage without Concurrency Control, there could be some problems. First of all, for such a case which two users renew one data at the same time, user 2 traces the contents

of data which user1 renewed and the renewal of user1 will be nullified. Second, while user1 is doing an updated task using the data, if user 2 reads the data before user 1 works on the data and user 3 reads the data after user1 works on the data, there will be in discord with the value of same data. At last, when user1 updates the task after reading the data and fails the task so user1 restores the original value of data, during the time that user 2 reads the unfinished value of data, the original value of data and the new value of data will be different.

Concurrency Control is divided into pessimistic and optimistic. Pessimistic Concurrency Control, so-called exclusive lock, is a fundamental way to prohibit people who approach one data at the same time. Data user protects the data from others who try to access the data using Lock Operation and prevents the simultaneous access to public data. Pessimistic Concurrency Control sustains data consistency to be limited to one about the simultaneous access of public data[8].

Optimistic Concurrency Control always allows to access public data without numerical order. However, if many users request for changing the data at the same time, the user who has the late timestamp will be canceled considering an order of Timestamp and optimistic concurrency control maintains logical consistency of data to re-do the work[9,10].

3. Proposed Method

All users who use the storage for collaboration create the private key and the public key, are utilized in Public-Key Encryption system.

To save message M in the storage, first of all, user gets hash value H of the message M using hash function like Figure 5. Second, the hash value H which is made from the message M is encrypted by using the user's private key. So, it is S (H) which is come through this procedure and this is a sign of the user about the message M.

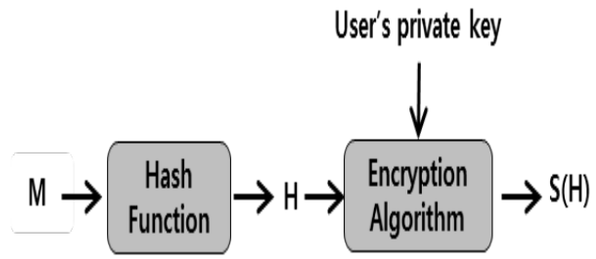

Figure 5. Digital Signature value S(H).

Data D which is saved in storage consists of message M, ID of the last user, signature S(H) of user about the message M, the date that the message M was created, the date that the message M was last updated and time-stamp TS. The first value of TS is 1 and the value of TS is

Figure 6. Data D.

increased by one whenever the message M gets uploaded. The figure of saved data D is same as Figure 6.

When the user reads the message M of data D that is saved in the storage, the user has to go through formalities like Figure 7.

- The user reads TS of the data D, which the user tries to access, and saves it.
- The user gets the hash value H decoding S (H) of the data D by using the key of specific user who lastly saved the data.
- The user gets hash value H' of the message M after reading the message M and compares H with H'.
- If H=H', the user reads the message M and uses it. If H is not same as H', the message M has to be discarded.

When the user renews the message M, that the user read, the user has to go through formalities like Figure 8.

- The user re-reads TS of the data D and saves TS'. The user compares it with TS which the user read in number one of Figure 7.
- If TS=TS', the user renews the message M. After increasing the value of TS by one, the user reframes the Data D.
- If TS is not same as TS', the user re-reads from TS of one in Figure 7.

When the user saves the message M which the user renews,

- The user makes H', is the hash value of message M, using the hash function.
- The user creates digital signature S (H'), which is used by the personal key of the user, in the hash value H'.
- The user constructs the data D'. (the last user ID', the last updated', TS')

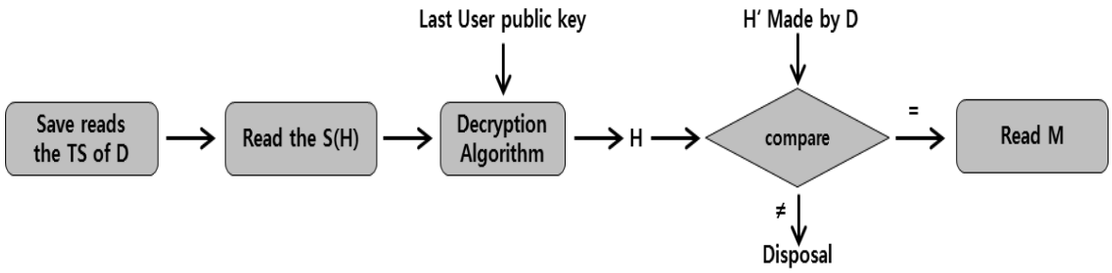

Figure 7. Read Messge M.

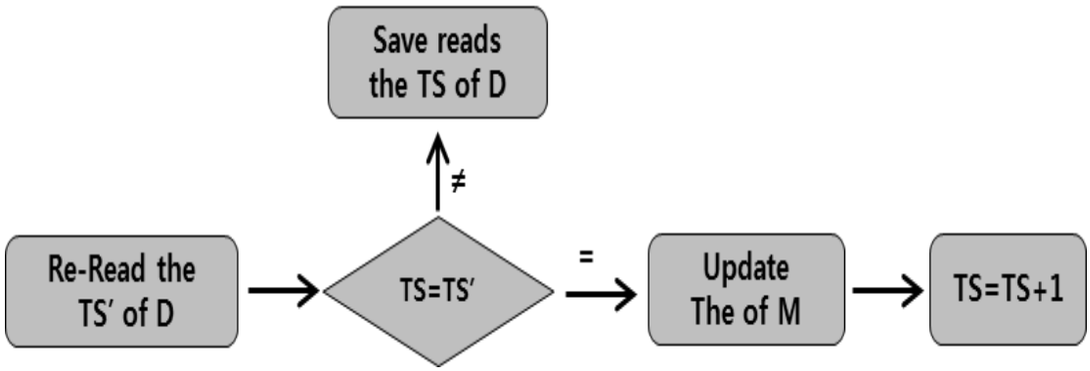

Figure 8. Read Messge M.

Table 1. Safety Analysis

Classification	Security system	Context
Logical consistency	●	While the user is working on the task, the user can confirm whether there is an unauthorized attempt of the data by utilizing TimeStamp.0
Integrity	●	The user can confirm whether the data is changed by using the hash value H of the message M
User authentication	●	All users who use the data in Cloud storage for Collaboration are possible to certify user authentication because of public-key encryption system

4. Stability Assessment of Suggesting Way

In the chapter 4, I evaluate the stability of ensuring integrity, logical consistency and user authentication about the saved data in the storage which the suggested method is in Cloud Storage for Collaboration.

First of all, when the user wants to read the data which is saved in Cloud Storage for Collaboration in the suggested method, the user has to read timestamp which is saved in the data before anything else. After reading timestamp and saving it, the user has to make sure whether others access the data or not to renew the data. If the timestamp which the user saved when the user read the data at the first time and the timestamp which the user tries to update are same, it is okay to renew the data because this means that there is no another access. If they are different, the user gives up renewing the data and starts to read the

data. When the user renews the data, the user can confirm that there is another user to access the data. Therefore, it guarantees logical consistency of the data through concurrency control.

Second, the data D which can be saved in Cloud Storage for Collaboration is consisted of the saved message M, the hash value H of the message M, user ID, the digital signature S(H), which is encrypted by the hash value as the private key, the date that the data was created, the last update, TimeStamp TS. The hash value of the message provides integrity because the user can confirm whether there is an unauthorized attempt of the data.

At last, all users who use the data in Cloud storage for Collaboration are possible to certify user authentication because of Public-Key Encryption system.

5. Conclusion

The users save data in 'online virtual network storage space' which Cloud Storage for Collaboration provides and Cloud Storage makes the users to work in any place where internet is available and with any device they use. Cloud storage for Collaboration needs special security unlike normal cloud storage because Cloud Storage for Collaboration opens to many people. This study proposes how to secure public data ensuring integrity of data that is saved in storage, logical consistency and user authentication.

The methods which this study provides, first of all, when the user saves the message M, this makes the user confirm whether there is other user who access the data by putting timestamp in the message M, during the time that the user renews the data. Moreover, it is possible to guarantee logical consistency of data through Concurrency Control.

Second, when the users who use the storage save the message M, the user saves the data D connecting with the message M, the hash value H of the message M, user ID, user signature S(H) of the hash value, the date that the data was created, the last update, timestamp and others. The user who tries to read the message M can compare the hash value H that was made before reading the message M with the hash value H' that is new updated after reading the message M. That is why it makes guarantee data integrity. Furthermore, the user authentication can be guaranteed because the user digital signature S (H) of the hash value shows the people who read the message M.

In the future, there will be more studies about how to certify the user authentication in the both aspects of the users and Cloud Storage for Collaboration

6. References

1. Kim D-H. Design and implementation of authentication system for reinforced security in cloud computing environment. Graduate School of Gachon University; 2013.
2. Park S-N. A study on the security strategy of cloud computing services. Graduate School of Paichai University; 2013.
3. Lim C-S. Design and implementation of a collaborative team-based cloud storage system. Graduate School Soongsil University; 2011.
4. Lee H-C. CSware: a middleware supporting collaborative workspaces based on cloud storage. Graduate School Ulsan University; 2012.
5. Han J-S. Security threats in the mobile cloud service environment. The Journal of Digital Policy & Management. 2014; 12(5):263–9.
6. Stallings W. Cryptography & network security. Prentice Hall; 2011.
7. Rivest RL, Shamir A, Adleman L. A method of obtaining digital signature and public key cryptosystem. ACM Communicatin. 1978; 21(2):120–6.
8. Kim C-H. Concurrency control and communication protocol for collaborative virtual environments. Graduate School Korea Advanced Institute of Science and Technology; 1998.
9. Park Y-H, Lee J-H, Kim M-J. Optimistic concurrency control for satisfying temporal consistency in realtime database. Proceeding of KIISE. 2000; 27(2):116–8.
10. Available from: http://blog.naver.com/jjang001313/140187225371.