International Journal of Current Research and Modern Education (IJCRME)
ISSN (Online): 2455 - 5428
(www.rdmodernresearch.com) Volume I, Issue I, 2016

SECURE VALIDITY FOR CHANGING SOCIAL NETWORK POLICY

N. Pandidurai* & A. Anitha**

* PG Scholar, Department of Master of Computer Applications, Dhanalakshmi Srinivasan Engineering College, Perambalur, Tamilnadu

** Assistant Professor, Department of Master of Computer Applications, Dhanalakshmi Srinivasan Engineering College, Perambalur, Tamilnadu

Abstract:

In this project Dynamic Proofs of Retrievability for Coded Cloud Storage Systems. We are introducing the concept of dynamic proof when user give more request to the server the server will overloaded. So we are introducing the concept of dynamic proof like captcha or OTP. Whenever users doing some operation in cloud like upload, file request or download a dynamic proof will be generated for this process. Then we are implementing content matching when ever user modifies the data. The landowner will match the content of original file and modified file. If the content is meaningful data owner will update the content, otherwise data owner will delete the modify file. The projects consist of three roles admin, audit and user. The user may be data owner or data-user and admin will control whole website. The data-owner will upload the data by encrypting with key. The main purpose of the audit is to verify the file of data owner on uploading the file on server. So we are using key generation algorithm for encrypting. To avoid data modification we are implementing dynamic.

Index Terms: File Modifying & File Upload

1. Introduction:

The main objective is used to reduce the load on server we are implementing dynamic proof and to recover the data when the user modifies the data in cloud. For recovering the data we are implementing the content matching to recover the file from the server. The main scope of project dynamic proof of retrievability scheme for coded cloud storage systems generating dynamic proof whenever users doing some operation in cloud like upload, file-request or download a dynamic proof will be generated for this process. Then we are implementing content matching on file recovery. Then we are implementing content matching when ever user modifies the data. The landowner will match the content of original file and modified file. If the content is meaningful data owner will update the content, otherwise data owner will delete the modify file. The project consists of three roles admin, audit and user. The user may be data owner or data-user and admin will control whole website. The data-owner will upload the data by encrypting with key. The main purpose of the audit is to verify the file of data owner on uploading the file on server. So we are using key generation algorithm for encrypting. To avoid data modification we are implementing dynamic proof. The reminder of this paper is organized as follows. Section II, describes the Related Works. Section III, describes the Proposed Work. Section IV, describes the Experimental Evaluation and Results. Section V summarizes the Conclusion and Future Enhancement.

2. Related Works:

The .NET Framework consists of the common language runtime and the .NET Framework class library. The common language runtime is the foundation of the .NET Framework. You can think of the runtime as an agent that manages code at execution time, providing core services such as memory management, thread management, and

remoting, while also enforcing strict type safety and other forms of code accuracy that promote security and robustness. In fact, the concept of code management is a fundamental principle of the runtime. Code that targets the runtime is known as managed code, while code that does not target the runtime is known as unmanaged code. The class library is a comprehensive, object-oriented collection of reusable types that you can use to develop applications ranging from traditional command-line or graphical user interface (GUI) applications to applications based on the latest innovations provided by ASP.NET, such as Web Forms and XML Web services.

The .NET Framework can be hosted by unmanaged components that load the common language runtime into their processes and initiate the execution of managed code, thereby creating a software environment that can exploit both managed and unmanaged features. The .NET Framework not only provides several runtime hosts, but also supports the development of third-party runtime hosts. For example, ASP.NET hosts the runtime to provide a scalable, server-side environment for managed code. ASP.NET works directly with the runtime to enable ASP.NET applications and XML Web services, both of which are discussed later in this topic.

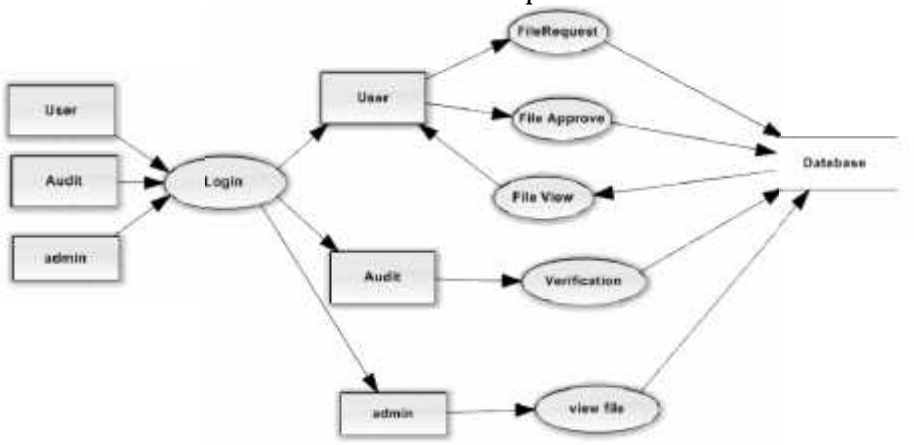

Figure 1: System Flow Diagram

3. Proposed Work:

We are introducing the concept of dynamic proof when user give more request to the server the server will overloaded. So we are introducing the concept of dynamic proof like captcha or OTP. Whenever users doing some operation in cloud like upload, file request or download a dynamic proof will be generated for this process. Then we are implementing content matching when ever user modifies the data. The landowner will match the content of original file and modified file. If the content is meaningful data owner will update the content, otherwise data owner will delete the modify file. The project consists of three roles admin, audit and user. The user may be data owner or data-user and admin will control whole website. The data-owner will upload the data by encrypting with key. The main purpose of the audit is to verify the file of data owner on uploading the file on server. So we are using key generation algorithm for encrypting. To avoid data modification we are implementing dynamic proof

File Upload: In this module, Data owner will upload file in server. The file is encrypted and splited and stored in server by using two key public and private key. Public key is given to third party for verification and key is with data owner for sharing purpose. The file will be encrypted and stored in cloud server.

Cloud Storage: The encode is spitted in to two codes they are outer code and inner code. The outer code consists of encrypted file and inner code consists of normal file.

The files are splited into three blocks. By using inner code we can recover our file from corruption.

Audit Verification: The auditor will verify will file uploaded by data owner. The auditor will verify inner code ie, encrypted file. The auditor will verify the total number of blocks. After verification file will be uploaded in server. The file is verified based on File id and filename.

File Request: The user is login to the account on next time using the normal first factor and the second factor is modified password. Thus the password is referred from the user mobile phone. And thus the password has to be used to the next login process.

File Access Block: In this module, if users try to modify the file the alert will go to data-owner. If users try to modify a file more than three times. We can block the modified user after giving request to admin and admin will approve then only he can access the file.

4. Experimental Analysis and Results:

Implementation is the process of translating design specification in to source code. The primary goal of implementation is to write source code and internal implementation. So that conformance of code to its specification can be easily verified, So that debugging, testing and modification are eased. The source is developed with clarity, simplicity and elegance. The coding is done in a modular fashion giving such importance even to the minute detail so, when hardware and storage procedures are changed or now data is added, rewriting of application programs is not necessary. To adapt or perfect use must determine new requirements, redesign generate code and test exiting software/hardware. Traditionally such task when they are applied to an existing program has been called maintenance.

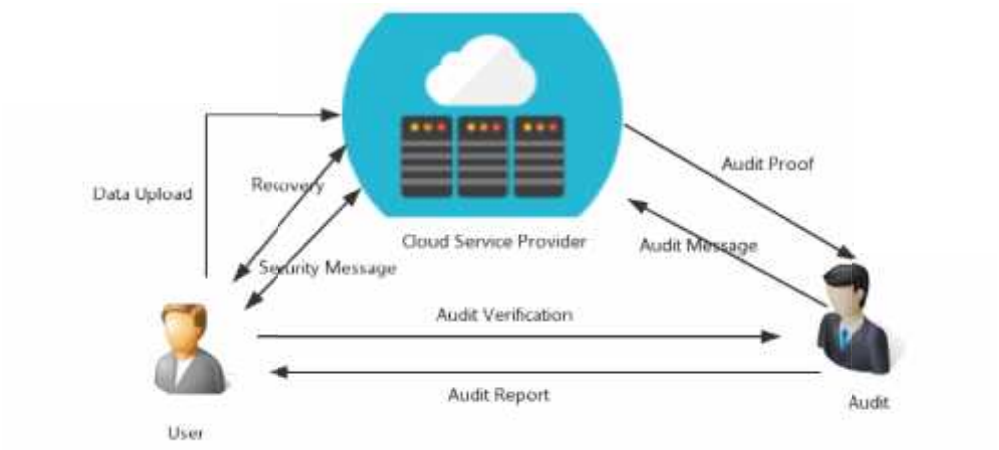

Figure 2: System Architecture

5. Conclusion and Future Enhancement:

At last we conclude a new dynamic proof of retrievability scheme for coded cloud storage systems. To reduce the overload on server by using dynamic proof and we are implementing captcha for reducing overload. We are implementing recovery process for data by content matching. At last we implement the user blocking when modifies the file. The Future scopes to save resources as well as the online burden potentially brought by the periodic auditing and accidental repairing, the data owners resort to the TPA for integrity verification and delegate the reparation to the proxy. We introduce proxy from user offline burden.

6. References:

1. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the Clouds: A Berkeley View of Cloud Computing," Commun. ACM, vol.53, no.4, pp.50-58, 2010.
2. Amazon.com, "Amazon S3 Availability Event: July 20, 2008,"http:// status. aws.amazon.com/s3-20080720.html, July 2008.
3. S. Wilson, "Appengine Outage," http://www.cioweblog.com/50226711 /appengine outage.php, June 2008.
4. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson,and D. Song, "Provable Data Possession at Untrusted Stores,"Proc. 14th ACM Conf. Computer and Comm. Security (CCS'07),pp.598-609, 2007.
5. Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling PublicV erifiability and Data Dynamic for Storage Security in Cloud Computing," Proc. 14th European Symp. Research in Computer Security (ESORICS'09), pp. 355-370, 2009.
6. A. Juels and B.S. Kaliski, "PORs: Proofs of Retrievability for Large Files," Proc. 14th ACM Conf. Computer and Comm. Security (CCS'07), pp. 584-597, 2007.
7. C, Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing," Proc. 29th Annual IEEE Int'l Conf. Computer Comm. (INFOCOM'10), pp. 525-533, 2010.
8. Y. Zhu, H. Wang, Z. Hu, G.J. Ahn, H, Hu, and S.S. Yau, "Cooperative Provable Data Possession," Report 2010/234, Cryptologye Print Archive, 2010.
9. G. Ateniese, R. Burns, R. Curtmola, J. Herring, O. Khan, L. Kissner, Z. Peterson, and D. Song, "Remote Data Checking Using Provable Data Possession," ACM Transactions on Information and System Security, vol.14, no.1, pp.12-34, 2011.