International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 13
September 2016

Data inspection and security in the duplication of the BigCloud data

***G Kamala, **E Devender Rao, ***M V Ramana Murthy**

*Department of Mathematics & computer Science, Osmania University, Hyderabad.

**Associate Professor, Aurora's PG College, Ramanthapur, Hyderabad

***Department of Mathematics & computer Science, Osmania University, Hyderabad

ABSTRACT

In this work, we study the problem of integrity inspection and secure duplication on cloud data. Specifically, aiming at Achieving both data integrity and duplication in cloud, wepropose two secure systems, namely BigCloud. BigCloud introduces an inspection entity with a maintenance of a MapReduce cloud, which helps clients generate data tags before uploading as well as audit the integrity of data having been stored in cloud. Compared with previous work, the computation by user in BigCloud is greatly reduced during the file uploading and inspection phases. BigCloud+ is designed motivated by the fact that customers always want to encrypt their data before uploading, and enables integrity inspection and secure duplication on encrypted data.

I. INTRODUCTION

Cloud storage is a model of networked enterprise storage where data is stored in virtualized pools of storage which are generally hosted by third parties. Cloud storage provides customers with benefits, ranging from cost saving and simplified convenience, to mobility opportunities and scalable service. These great features attract more and more customers to utilize and storage their personal data to the cloud storage .
Even though cloud storage system has been widely adopted, it fails to accommodate some important emerging needs such as the
abilities of inspection integrity of cloud files by cloud clients and detecting duplicated files by cloud servers. We illustrate both problems below.

The first problem is integrity auditing. The cloud server is able to relieve clients from the heavy burden of storage management and maintenance. The most difference of cloud storage from traditional in-house storage is that the data is transferred via Internet and stored in an uncertain domain, not under control of the clients at all, which inevitably raises clients great concerns on the integrity of their data. These concerns originate from the fact that the cloud storage is susceptible to security threats from both outside and inside of the cloud .

The second problem is secure duplication. The rapid adoption of cloud services is accompanied by increasing volumes of data stored at remote cloud servers. Among these remote stored files, most of them are duplicated
.

II. *MOTIVATION*

Motivated by the fact that customers always want to encrypt their data before uploading, for reasons ranging from personal privacy to corporate policy, we introduce a key server into BigCloud as with and propose the BigCloud schema. Besides supporting integrity inspectionand secure deduplication, BigCloud+ enables the guarantee of file confidentiality. we propose a method of directly inspectionintegrity on encrypted data. The challenge of deduplication on encrypted is the prevention of dictionary attack.

III. ARCHITECTURE

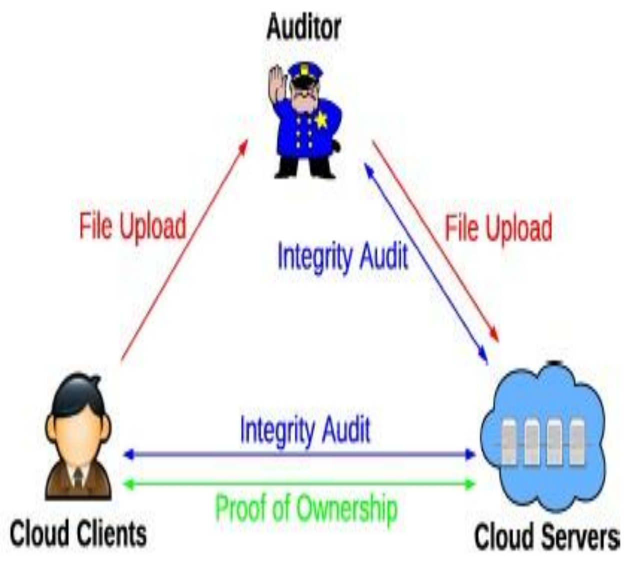

Figure : Architecture of BigCloud System

In the BigCloud system, we have three entities:

• **Cloud Clients** have large data files to be stored and rely on the cloud for data maintenance and computation. They can be either individual consumers or commercial organizations;

• **Cloud Servers** virtualize the resources according to the requirements of clients and expose them as storage pools. Typically, the cloud clients may buy or lease storage capacity from cloud servers, and store their individual data in these bought or rented spaces for future utilization;

• **Auditor** which helps clients upload and audit their outsourced data maintains MapReduce cloud and acts like a certificate authority. This assumption presumes that the auditor is associated with a pair of public and private keys. Its public key is made available to the other entities in the system .

IV. EXISTING SYSTEM

- BigCloud introduces an with a maintenance of a Map Reduce cloud, which helps clients generate data tags before uploading as well as audit the integrity of data having been stored in cloud.
- In addition, SecCoud also enables secure deduplication. Notice that the ―security‖ considered in SecCoud is the prevention of leakage of side channel information. In order to prevent the leakage of such side channel information, we follow the tradition of and design a proof of ownership protocol between clients and cloud servers, which allows clients to prove to cloud servers that they exactly own the target data.
-

V. PROPOSED SYSTEM

we describe our proposed BigCloud system. Specifically, we begin with giving the system model of Sec- Cloud as well as introducing the design goals for BigCloud.

™ ***System Model***

we propose the BigCloud system. In the BigCloud system, we have three entities:

• Cloud Clients have large data files to be stored and rely on the cloud for data maintenance and computation

• Cloud Servers virtualize the resources according to the requirements of clients and expose them as storage pools.

• Auditor which helps clients upload and audit their outsourced data maintains a Map Reduce cloud and acts like a certificate authority. This assumption presumes that the auditor is associated with a pair of public and private keys .

The BigCloud system supporting file-level deduplicationincludes the following three Protocols:-

1) File Uploading Protocol: This protocol aims at allowing clients to upload files via the auditor. Specifically, the file uploading protocol includes three phases:

• ***Phase 1* (cloud client → cloud server)**: client performs the duplicate check with the cloud server to confirm if such a file is stored in cloud storage or not before uploading a file. If there is a duplicate, another protocol called Proof of Ownership will be run between the client and the cloud storage server. Otherwise, the following protocols (including *phase*

2 and *phase* 3) are run between these two entities.

• ***Phase 2* (cloud client → auditor):** client uploads files to the auditor, and receives a receipt from auditor.

• ***Phase 3* (auditor → cloud server):** auditor helps generate a set of tags for the uploading file, and send them along with this file to cloud server.

2) Integrity InspectionProtocol: It is an interactive protocol for integrity verification and allowed to be initialized by any entity except the cloud server. In this protocol, the cloud server plays the role of prover, while the auditor or client works as the verifier. This protocol includes two phases:

• ***Phase 1* (cloud client/auditor → cloud server**): verifier (i.e., client or auditor) generates a set of challenges and sends them to the prover (i.e., cloud server).

• ***Phase 2* (cloud server → cloud client/auditor):** based on the stored files and file tags, prover (i.e., cloud server) tries to prove that it exactly owns the target file by sending the proof back to verifier (i.e., cloud client or auditor). At the end of this protocol, verifier outputs true if the integrity verification is passed.

3) Proof of Ownership Protocol: It is an interactive protocol initialized at the cloud server for verifying that the client exactly owns a claimed file. This protocol is typically triggered along with file uploading protocol to prevent the leakage of side channel information. On the contrast to integrity inspection protocol, in PoW the cloud server works as verifier, while the client plays the role of prover. This protocol also includes two phases:

• ***Phase 1* (cloud server → client)**: cloud server generates a set of challenges and sends them to the client.

• ***Phase 2* (client → cloud server):** the client responds with the proof for file ownership, and cloud server finally verifies the validity of proof.

REFERENCE:

1. Oberheide, J., Veeraraghavan, K., Cooke, E., Flinn, J., & Jahanian, F. (2008, June). Virtualized in-cloud security services for mobile devices. In Proceedings of the First Workshop on Virtualization in Mobile Computing (pp. 31-35). ACM.
2. Schoo, P., Fusenig, V., Souza, V., Melo, M., Murray, P., Debar, H., ... & Zeghlache, D. (2011). Challenges for cloud networking security. In Mobile Networks and Management (pp. 298-313). Springer Berlin Heidelberg.
3. Bitar, N., Gringeri, S., & Xia, T. J. (2013). Technologies and protocols for data center and cloud networking. Communications Magazine, IEEE, 51(9), 24-31.
4. Houidi, I., Mechtri, M., Louati, W., & Zeghlache, D. (2011, July). Cloud service delivery across multiple cloud platforms. In Services Computing (SCC), 2011 IEEE International Conference on (pp. 741-742). IEEE.
5. Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., & Zagorodnov, D. (2009, May). The eucalyptus open-source cloud-computing system. In Cluster Computing and the Grid, 2009. CCGRID'09. 9th IEEE/ACM International Symposium on (pp. 124-131). IEEE.
6. Wodczak, M. (2011, November). Resilience aspects of autonomic cooperative communications in context of cloud networking. In Network Cloud Computing and Applications (NCCA), 2011 First International Symposium on (pp. 107-113). IEEE.
7. Bitar, N., Gringeri, S., & Xia, T. J. (2013). Technologies and protocols for data center and cloud networking.
8. Communications Magazine, IEEE, 51(9), 24-31.
9. Bechler, M., Hof, H. J., Kraft, D., Pahlke, F., & Wolf, L. (2004, March). A cluster-based security architecture for ad hoc networks. In INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies (Vol. 4, pp. 2393-2403). IEEE.
10. Covington, M. J., Fogla, P., Zhan, Z., & Ahamad, M. (2002). A context-aware security architecture for emerging applications. In Computer Security Applications Conference, 2002. Proceedings. 18th Annual (pp. 249-258). IEEE.
11. Zhou, L., & Chao, H. C. (2011). Multimedia traffic security architecture for the internet of things. Network, IEEE, 25(3), 35-40.
12. Pensak, D. A., Cristy, J. J., & Singles, S. J. (2001). U.S. Patent No. 6,289,450. Washington, DC: U.S. Patent and Trademark Office.
13. Nagaratnam, N., Janson, P., Dayka, J., Nadalin, A., Siebenlist, F., Welch, V., ... & Tuecke, S. (2002). The security architecture for open grid services. Open Grid Service Architecture Security Working Group (OGSA-SEC-WG), 1-31.
14. Wood, D. L., Pratt, T., Dilger, M. B., Norton, D., & Nadiadi, Y. (2004). U.S. Patent No. 6,691,232. Washington, DC: U.S. Patent and Trademark Office.
15. Okuhara, M., Shiozaki, T., & Suzuki, T. (2010). Security architecture for cloud computing. Fujitsu Sci. Tech. J, 46(4), 397-402.
16. Subashini, S., & Kavitha, V. (2011). A survey on security issues in service

delivery models of cloud computing. Journal of network and computer applications, 34(1), 1-11.
17. Ramgovind, S., Eloff, M. M., & Smith, E. (2010, August). The management of security in cloud computing. In Information Security for South Africa (ISSA), 2010 (pp. 1-7). IEEE.
18. Zissis, D., & Lekkas, D. (2012). Addressing cloud computing security issues. Future Generation computer systems, 28(3), 583-592.
19. Ramgovind, S., Eloff, M. M., & Smith, E. (2010, August). The management of security in cloud computing. In Information Security for South Africa (ISSA), 2010 (pp. 1-7). IEEE.
20. Zissis, D., & Lekkas, D. (2012). Addressing cloud computing security issues. Future Generation computer systems, 28(3), 583-592.
21. Subashini, S., & Kavitha, V. (2011). A survey on security issues in service delivery models of cloud computing. Journal of network and computer applications, 34(1), 1-11.
22. Wang, Q., Wang, C., Li, J., Ren, K., & Lou, W. (2009). Enabling public verifiability and data dynamics for storage security in cloud computing. In Computer Security–ESORICS 2009 (pp. 355-370). Springer Berlin Heidelberg.
23. Rimal, B. P., Choi, E., & Lumb, I. (2009, August). A taxonomy and survey of cloud computing systems. In INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on (pp. 44-51). Ieee.
24. Briand, L., & Labiche, Y. (2001). A UML-based approach to system testing. In ≪ UML≫ 2001—The Unified Modeling Language. Modeling Languages, Concepts, and Tools (pp. 194-208). Springer Berlin Heidelberg.
25. Reuys, A., Kamsties, E., Pohl, K., & Reis, S. (2005, January). Model-based system testing of software product families. In Advanced Information Systems Engineering (pp. 519-534). Springer Berlin Heidelberg.
26. Campione, M. (2001). *The Java tutorial: a short course on the basics* (Vol. 1). Addison-Wesley Professional. P(1-10)
27. Dean, J., & Dean, R. (2007). Introduction to programming with Java: a problem solving approach. McGraw-Hill, Inc..p(15-30)