Cooperative Secure Multi – Cloud Verification System

Mr. B.Shankernayak
Associate Professor
Dept. Of Information Technology
Swami Vivekananda Institute Of Technology, Sec-Bad

Msb.Deekshitha
Btech Final Year Student
Dept. Of Information Technology Line Swami Vivekananda Institute Of Technology,Sec-Bad

Mr J. Karthik Reddy
Btech Final Year Student
Dept. Of Information Technology Line Swami Vivekananda Institute Of Technology,Sec-Bad

Msb.Swathi
Btech Final Year Student
Dept. Of Information Technology Line Swami Vivekananda Institute Of Technology,Sec-Bad

Abstract:

Provable data possession (PDP) is a technique for ensuring the integrity of data in storage outsourcing. In this paper, we address the construction of an efficient PDP scheme for distributed cloud storage to support the scalability of service and data migration, in which we consider the existence of multiple cloud service providers to cooperatively store and maintain the clients data. We present a cooperative PDP(CPDP) scheme based on homomorphic verifiable response and hash index hierarchy. We prove the security of our scheme based on multi-prover zero-knowledge proof system, which can satisfy completeness, knowledge soundness, and, zero-knowledge properties. In addition, we articulateperformance optimization mechanisms for our scheme, and in particular present an efficient method for selecting optimal parameter values to minimize the computation costs of clients and storage service providers. Our experiments show that our solution introduces lower computation overheads in comparison with non-cooperative approaches.

2. Introduction

Become a faster profit growth point by providing a comparably low-cost, scalable, position-independent platform for clients' data. Since cloud computing environment is constructed based on open architectures and interfaces, it has the capability to incorporate multiple internal and/or external cloud services together to provide high interoperability. We call such a distributed cloud environment as a *multi-Cloud* (or *hybrid cloud*). Often, by using virtual infrastructure management (VIM), a multi-cloud allows clients to easily access his/her resources remotely through interfaces such as Web services provided by Amazon EC2. There exist various tools and technologies for multicloud, such as Platform VM Orchestrator, VMware vSphere, and Ovirt. These tools help cloud providers construct a distributed cloud storage platform (DCSP) for managing clients' data. However, if such an important platform is vulnerable to security attacks, it would bring irretrievable losses to the clients. For example, the confidential data in an enterprise may be illegally accessed through a remote interface provided by a multi-cloud, or relevant data and archives may be lost or tampered with when they are stored into an uncertain storage pool outside the enterprise. Therefore, it is indispensable for cloud service providers (CSPs) to provide security techniques for managing their storage services. Provable data possession (PDP)(or proofs of retrievability (POR)) is such a probabilistic proof technique for a storage provider to prove the integrity and ownership of clients' data without downloading data. The proof-checking without downloading makes it especially important for large-size files and folders (typically including many clients' files) to check whether these data have been tampered with or deleted without downloading the latest version of data. Thus, it is able to replace traditional hash and signature functions in storage outsourcing. Various PDP schemes have been recently proposed, such as Sclable PDP and Dynamic PDP. However, these schemes mainly focus on PDP issues at untrusted servers in a *single* cloud storage provider and are not suitable for a multi-cloud environment (see the comparison of POR/PDP schemes in Table 1) motivation.

To provide a low-cost, scalable, location independent platform for managing clients' data, current cloud storage systems adopt several new distributed file systems, for example, Apache Hadoop Distribution File System (HDFS), Google File System (GFS), Amazon S3 File System, Cloud Store etc. These file systems share some similar features: a single metadata server provides centralized management by a global namespace; files are split into blocks or chunks and stored on block servers; and the systems are comprised of interconnected clusters of block servers. Those features enable cloud service providers to store and process large amounts of data.

However, it is crucial to offer an efficient verification on the integrity and automatic recovery. Moreover, this verification is necessary to provide reliability by automatically maintaining multiple copies of data and automatically redeploying processing logic in the event of failures. Although existing schemes can make a false or true decision for data possession without downloading data at untrusted stores, they are not suitable for a distributed cloud storage environment since they were not originally constructed on interactive proof system. For example, the schemes based on Merkle Hash tree (MHT), such as DPDP-I, DPDP-II and SPDP in Table 1, use an authenticated skip list to check the integrity of file blocks adjacently in space. unfortunately, they did not provide any algorithms for constructing distributed Merkle trees that are necessary for efficient verification in a multi-cloud environment. In addition, when a client asks for a file block, the server needs to send the file block along with a proof for the intactness of the block. However, this process incurs significantcommunication overhead in a multi-cloud environment, since the server in one cloud typically needs to generate such a proof with the help of other cloud storage services, where the adjacent blocks are stored. The other schemes, such as PDP, CPOR-I, and CPOR-II in Table 1, are constructed on homomorphic verification tags, by which the server can generate tags for multiple file blocks in terms of a single response value. However, that doesn't mean the responses from multiple clouds can be also combined into a single value on the client side. For lack of homomorphic responses, clients must invoke the PDP protocol repeatedly to check the integrity of file blocks stored in multiple cloud servers.

Also, clients need to know the exact position of each file block in a multi-cloud environment. In addition, the verification process in such a case will lead to high communication overheads and computation costs at client sides as well. Therefore, it is of utmost necessary to design a cooperative PDP model to reduce the storage and network overheads and enhance the transparency of verification activities in cluster-based cloud storage systems. Moreover, such a cooperative PDP scheme should provide features for timely detecting abnormality and renewing multiple copies of data. Even though existing PDP schemes have addressed various security properties, such as public verifiability, dynamics, scalability, and privacy preservation, we still need a careful consideration of some potential attacks, including two major categories: *Data Leakage Attack* by which an adversary can easily obtain the stored data through verification process after running or wiretapping sufficient verification communications (see Attacks 1 and 3 in Appendix A), and *Tag Forgery Attack* by which a dishonest CSP can deceive the clients (see Attacks 2 and 4 in Appendix A). Although various security models have been proposed for existing PDP schemes, these models still cannot cover all security requirements, especially for provable secure privacy preservation and ownership authentication. To establish a highly effective security model, it is necessary to analyze the PDP scheme within the framework of zero-knowledge proof system (ZKPS) due to the reason that PDP system is essentially an interactive proof system (IPS), which has been well studied in the cryptography community. In summary, a verification scheme for data integrity in distributed storage environments should have the following features: · Usability aspect: A client should utilize the integrity check in the way of collaboration services. The scheme should conceal the details of the storage to reduce the burden on clients; · Security aspect: The scheme should provide adequate security features to resist some existing attacks, such as data leakage attack and tag forgery attack; · Performance aspect: The scheme should have the lower communication and computation overheads than non-cooperative solution.

3.SYSTEM ANALYSIS

EXISTING SYSTEM

There exist various tools and technology for multi-cloud, such as platform VM Orchestrator, VMwarevSphere, and Ovirt. These tools help cloud provides construct a distributed cloud storage platform for managingclients data. However, if such an important platform is vulnerable to security attacks, it would bring irretrievable losses to the clients. For example, the confidential data in an enterprise may be illegally accessed through a remote interface provided by a multi-cloud, or relevant data and archives may be lost or tampered with when they are stored into an uncertain storage pool outside the enterprise. Therefore, it is indispensible for cloud service providers to provide security techniques for managing their storage service.

PROPOSED SYSTEM

To check the availability and integrity of outsourced data in cloud storages, researches have proposed two basic approaches called Provable Data Possession and proofs ofRetrievability. Ateniese et al. First proposed the PDP model for ensuring possession of files on untrusted storages and provided an RSA-based scheme for static case that achieves the communication cost. They also proposed a publicity verifiable version, which allows anyone, not just the owner, to challenge the server for data possession. They proposed a lightweight PDP scheme based on cryptographic hash function and symmetric key encryption, but the servers can deceive the owners by using previous metadata or response due to the lack of randomness in challenges. The number of updates and challenges are limited and fixed in advance and users cannot perform block insertions anywhere.

4. System Design

OOAD OF THE SYSTEM:

An object oriented analysis and design language from the object management group. Many design

methodologies for describing object-oriented systems were developed in the late 1980s.UML standardizes several diagramming methods, including Grady Booch's work at Rational Software .Rum Baugh's Object Modeling Technique and Ivan Jacobson's wok on use cases.

UML:

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

TYPES

The following are the types of UML diagrams followed in this Project:

- Use Case Diagram.
- Activity Diagram.
- Class Diagram.
- Sequence Diagram.
- Collaboration Diagram.

DETAILED DESIGN
USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.Use cases in the system are

Use cases in our system

1. Registration.
2. Login.
3. File Upload.
4. View File.
5. Download.
6. Cloud Status.
7. File Alert.

Actors in our system

1. Tpa.
2. User.

Active classes in the system are:-

1. cloud
2. user.
3. tpa (third party auditor).

SEQUENCE DIAGRAM:

A Sequence diagram is an interaction diagram .It represents sequence of messages flowing from one object to another. It is used to visualize the sequence of calls in the system to perform a specific functionality.

Objects in the sequence diagram are :-*

1 User.
2 Login.
3 Upload.
4 Tpa.
5 Cloud.

5. Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

5.1 Test Cases

1.Check whether file is uploaded or not.
2.Verify the uploaded file according to the parameters.
3. notify unauthorized access to file.
4. View uploaded file.
4. Download the uploaded file by the user.

Conclusion

An efficient PDP scheme for distributed cloud storage has been developed. Based on homomorphic verifiable response and hash index hierarchy, we have proposed a cooperative PDP scheme to support dynamic scalability on multiple storage servers. We also showed that our scheme provided all security properties required by zero knowledge interactive proof system, so that it can resist various attacks even if it is deployed as a public audit service in clouds. Furthermore, we optimized the probabilistic query and periodic verification to improve the audit performance. Our experiments clearly demonstrated that our approaches only introduce a small amount of computation and communication overheads. Therefore, our solution can be treated as a new candidate for data integrity verification in outsourcing data storage systems. As part of future work, we would extend our work to explore more effective CPDP constructions. First, from our experiments we found that the performance of CPDP scheme, especially for large files, is affected by the bilinear mapping operations due to its high complexity. To solve this problem, RSA based constructions may be a better choice, but this is still a challenging task because the existing RSA based schemes have too many restrictions on the performance and security. Next, from a practical point of view, we still need to address some issues about integrating our CPDP scheme smoothly with existing systems, for example, how to match indexhash hierarchy with HDFS's two-layer name space, how to match index structure with cluster-network model, and how to dynamically update the CPDP parameters according to HDFS' specific requirements. Finally, it is still a challenging problem for the generation of tags with the length irrelevant to the size of data blocks. We would explore such a issue to provide the support of variable-length block verification.

REFERENCES

[1] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. T. Foster, "Virtual infrastructure management in private and hybrid clouds," IEEE Internet Computing, vol. 13, no. 5, pp. 14–22, 2009.

[2] G. Ateniese, R. C. Burns, R. Curtmola, J. Herring, L. Kissner, Z. N. J. Peterson, and D. X. Song, "Provable data possession at untrusted stores," in ACM Conference on Computer and Communications Security, P. Ning, S. D. C. di Vimercati, and P. F. Syverson, Eds. ACM, 2007, pp. 598–609.

[3] A. Juels and B. S. K. Jr., "Pors: proofs of retrievability for large files," in ACMConference on Computer and Communications Security, P. Ning, S. D. C. di Vimercati, and P. F. Syverson, Eds. ACM, 2007, pp. 584–597.

[4] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proceedings of the 4th international conference on Security and privacy in communication netowrks, SecureComm, 2008, pp. 1–10.

[5] C. C. Erway, A. K¨upc¸ ¨u, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in ACM Conference on Computer and Communications Security, E. Al-Shaer, S. Jha, and A. D. Keromytis, Eds. ACM, 2009, pp. 213–222.

[6] H. Shacham and B. Waters, "Compact proofs of retrievability," in ASIACRYPT, ser. Lecture Notes in Computer Science, J. Pieprzyk, Ed., vol. 5350. Springer, 2008, pp. 90–107.

[7] Q. Wang, C.Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in ESORICS, ser. Lecture Notes in Computer Science, M. Backes and P. Ning, Eds., vol. 5789. Springer, 2009, pp. 355–370.

[8] Y. Zhu, H. Wang, Z. Hu, G.-J.Ahn, H. Hu, and S. S. Yau, "Dynamic audit services for integrity verification of outsourced storages in clouds," in SAC, W. C. Chu, W. E. Wong, M. J. Palakal, and C.-C. Hung, Eds. ACM, 2011, pp. 1550–1557.

[9] K. D. Bowers, A. Juels, and A. Oprea, "Hail: a high-availability and integrity layer for cloud storage," in ACM Conference on Computer and Communications Security, E. Al-Shaer, S. Jha, and A. D. Keromytis, Eds. ACM, 2009, pp. 187–198.

[10] Y. Dodis, S. P. Vadhan, and D. Wichs, "Proofs of retrievability via hardness amplification," in TCC, ser. Lecture Notes in Computer Science, O. Reingold, Ed., vol. 5444. Springer, 2009, pp. 109–127.

[11] L. Fortnow, J. Rompel, and M. Sipser, "On the power of multiprover interactive protocols," in Theoretical Computer Science, 988, pp. 156–161.

[12] Y. Zhu, H. Hu, G.-J.Ahn, Y. Han, and S. Chen, "Collaborative integrity verification in hybrid clouds," in IEEE Conference on the 7th International Conference on Collaborative Computing: Networking, Applications and Worksharing, CollaborateCom, Orlando, Florida, USA, October 15-18, 2011, pp. 197–206.

[13] 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," EECS Department, University of California, Berkeley, Tech. Rep., Feb 2009.

[14] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in Advances in Cryptology (CRYPTO'2001), vol. 2139 of LNCS, 2001, pp. 213–229.

[15] O. Goldreich, Foundations of Cryptography: Basic Tools. Cambridge
University Press, 2001.

[16] P. S. L. M. Barreto, S. D. Galbraith, C. O'Eigeartaigh, and M. Scott, "Efficient pairing computation on supersingularabelian varieties," Des. Codes Cryptography, vol. 42, no. 3, pp. 239–271, 2007.

[17] J.-L. Beuchat, N. Brisebarre, J. Detrey, and E. Okamoto, "Arithmetic operators for pairing-based cryptography," in CHES, ser. Lecture Notes in Computer Science, P. Paillier and I. Verbauwhede, Eds., vol. 4727. Springer, 2007, pp. 239–255.

[18] H. Hu, L. Hu, and D. Feng, "On a class of pseudorandom sequences from elliptic curves over finite fields," IEEE Transactions on Information Theory, vol. 53, no. 7, pp. 2598–2605, 2007.

[19] A. Bialecki, M. Cafarella, D. Cutting, and O. O'Malley, "Hadoop: A framework for running applications on large clusters built of commodity hardware," Tech. Rep., 2005. [Online]. Available: http://lucene.apache.org/hadoop/

20] E. Al-Shaer, S. Jha, and A. D. Keromytis, Eds., Proceedings of the 2009 ACM Conference on Computer and Communications Security, CCS 2009, Chicago, Illinois, USA, November 9-13, 2009. ACM, 2009.