Efficient Provable Data Possession for Hybrid Clouds

Yan Zhu[1,2], Huaixi Wang[3], Zexing Hu[1], Gail-Joon Ahn[4], Hongxin Hu[4], Stephen S. Yau[4]
[1]Institute of Computer Science and Technology, Peking University, Beijing 100871, China
[2]Key Laboratory of Network and Software Security Assurance (Peking University), Ministry of Education
[3]School of Mathematical Sciences, Peking University, Beijing 100871, China
[4]Arizona State University, Tempe, AZ 85287, USA
{yan.zhu,wanghx,huzx}@pku.edu.cn, {gahn,hxhu,yau}@asu.edu

ABSTRACT

Provable data possession is a technique for ensuring the integrity of data in outsourcing storage service. In this paper, we propose a cooperative provable data possession scheme in hybrid clouds to support 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. Our experiments show that the verification of our scheme requires a small, constant amount of overhead, which minimizes communication complexity.

Categories and Subject Descriptors

H.3.2 [**Information Storage and Retrieval**]: Information Storage; E.3 [**Data**]: Data Encryption

General Terms

Design, Performance, Security

Keywords

Storage Security, Provable Data Possession, Hybrid Clouds

1. INTRODUCTION

In cloud computing, one of the core design principles is dynamic scalability, which guarantees cloud storage service to handle growing amounts of application data in a flexible manner or to be readily enlarged. By integrating multiple private and public cloud services, hybrid clouds can effectively provide dynamic scalability of service and data migration. For example, a client might integrate the data from multiple private or public providers into a backup or archive file (see Figure 1), or a service might capture the data from other services from private clouds, but the intermediate data and results are stored in hybrid clouds [3].

Although Provable Data Possession (PDP) schemes evolved around public clouds offer a publicly accessible remote interface to check and manage the tremendous amount of data, the majority of existing PDP schemes are incapable of satisfying such an inherent requirement of hybrid clouds in terms of bandwidth and time. In order to address this problem, we consider a hybrid cloud storage service involving three different entities, as illustrated in Figure 1: the cloud client who stores or uses data in the cloud; the cloud service provider (CSP) which has significant storage space and computation resources to manage and provide storage services; and the trusted third party (TTP) who stores the clients' audit data and offers the query services for their data.

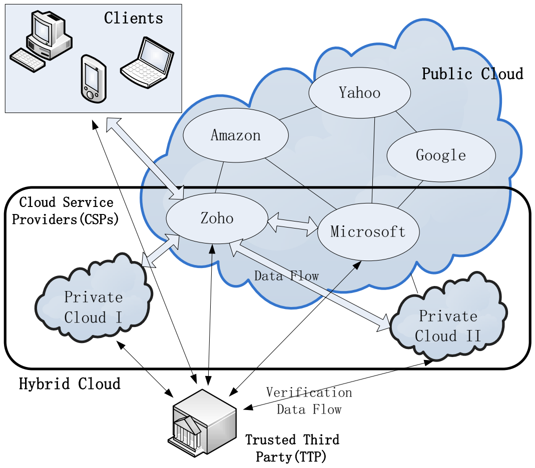


Figure 1: Cloud data storage architecture for hybrid clouds

In this architecture, we consider the existence of multiple CSPs to cooperatively store and maintain the clients' data, and a publicly verifiable PDP is used to verify the integrity and availability of their stored data in CSPs. The clients are allowed to dynamically access and update their data for various applications, and the verification process of PDP is seamlessly performed for the clients in hybrid clouds. Hence, it is a challenging problem to design a PDP scheme for supporting dynamic scalability.

In this work, we focus on the construction of PDP scheme for hybrid clouds, supporting privacy protection and dynamic scalability. We first provide an effective construction of Cooperative Provable Data Possession (CPDP) using Homomorphic Verifiable Responses (HVR) and Hash Index Hierarchy (HIH). This construction uses homomorphic property, such that the responses of the client's challenge computed from multiple CSPs can be combined into a single response as the final result of hybrid clouds. By using this mechanism, the clients can be convinced of data possession without knowing what machines or in which geographical locations their files reside. More importantly, a new hash index hierarchy is proposed for the clients to seamlessly store and manage the resources in hybrid clouds. Our experimental results also validate the effectiveness of our construction.

CCS'10, October 4–8, 2010, Chicago, Illinois, USA.
ACM 978-1-4503-0244-9/10/10.

2. COOPERATIVE PROVABLE DATA POSSESSION

In this section, we introduce the principles of our cooperative provable data possession for hybrid clouds, including the main technique, model, fragment structure, index hierarchy, and the architecture to support our scheme.

2.1 Homomorphic Verifiable Response

A homomorphism is a map $f : \mathbb{P} \rightarrow \mathbb{Q}$ between two groups such that $f(g_1 \oplus g_2) = f(g_1) \otimes f(g_2)$ for all $g_1, g_2 \in \mathbb{P}$, where $\oplus$ denotes the operation in $\mathbb{P}$ and $\otimes$ denotes the operation in $\mathbb{Q}$. This notation was used to define a Homomorphic Verifiable Tags (HVTs) in [2]: Given two values σ_i and σ_j for two message m_i and m_j, anyone can combine them into a value σ' corresponding to the sum of the message $m_i + m_j$.

When provable data possession is considered as a challenge-response protocol, we also extend this notation to introduce the concept of a Homomorphic Verifiable Responses (HVRs), which is used to integrate multiple responses from the different CSPs in cooperative PDP scheme, as follows:

Definition 1 (homomorphic verifiable response). *A response is called homomorphic verifiable response in PDP protocol, if given two responses θ_i and θ_j for two challenges Q_i and Q_j from two CSPs, there exists an efficient algorithm to combine them into a response θ corresponding to the sum of the challenges $Q_i \bigcup Q_j$.*

2.2 Definition of CPDP Model

In order to prove the integrity of data stored in hybrid clouds, we define a framework for Cooperative Provable Data Possession (CPDP) as follows:

Definition 2 (Cooperative-PDP). *A cooperative provable data possession scheme $\mathcal{S}'$ is a collection of two algorithms and an interactive proof system, $\mathcal{S}' = (\mathcal{K}, \mathcal{T}, \mathcal{P})$:*

$\mathcal{K}eyGen(1^\kappa)$: *takes a security parameter κ as input, and returns a secret key sk or a public-secret keypair (pk, sk);*

$\mathcal{T}agGen(sk, F, \mathcal{P})$: *takes a secret key sk, a file F, and a set of CSPs $\mathcal{P} = \{P_k\}$, and returns the triples (ζ, ψ, σ), where ζ is the secret of tags, $\psi = (u, \mathcal{H})$ is a set of verification parameters u and an index hierarchy $\mathcal{H}$ for F, $\sigma = \{\sigma^{(k)}\}_{P_k \in \mathcal{P}}$ denotes a set of all tags, where $\sigma^{(k)}$ is the tags of the fraction $F^{(k)}$ of F in P_k; and*

$\mathcal{P}roof(\mathcal{P}, V)$: *is a protocol for a proof of data possession between CSPs $\mathcal{P} = \{P_k\}$ and a verifier V. At the end of the protocol, V returns a bit $\{0|1\}$ denoting a binary decision for either false or true. It includes two cases:*

- $\langle \sum_{P_k \in \mathcal{P}} P_k(F^{(k)}, \sigma^{(k)}), V(sk, \zeta) \rangle$ *is a private proof, where each P_k takes a fraction of file $F^{(k)}$ and a set of all tags $\sigma^{(k)}$, and V takes a secret key sk and the secret of tags ζ; and*
- $\langle \sum_{P_k \in \mathcal{P}} P_k(F^{(k)}, \sigma^{(k)}), V \rangle (pk, \psi)$ *is a public proof, where each P_k takes a file $F^{(k)}$ and a set of all tags $\sigma^{(k)}$, and a public key pk and a set of public parameters ψ are the common input between P and V.*

For both cases, $P(x)$ denotes the subject P holds the secret x, and $\langle P, V \rangle (x)$ denotes both parties P and V share a common data x in a protocol. $\sum_{P_k \in \mathcal{P}}$ denotes the cooperative computing in $P_k \in \mathcal{P}$.

To realize the CPDP, a trivial way is to check the data stored in each cloud one by one. However, it would cause significant cost growth in terms of communication and computation overheads. It is obviously unreasonable to adopt such a primitive approach that diminishes the advantages of cloud storage: scaling arbitrarily up and down on-demand [1].

2.3 Fragment Structure of CPDP

We propose a fragment structure of CPDP scheme based on the above-mentioned model as shown in Figure 2, which has following characters: 1) a file is split into $n \times s$ sectors and each block (s sectors) corresponds to a tag, so that the storage of signature tags can be reduced with the order of s; 2) the verifier can check the integrity of a file by random sampling approach, which is a matter of the utmost importance for large or huge files; and 3) this structure relies on homomorphic properties to aggregate the data and tags into a constant size response, which minimizes network communication overheads.

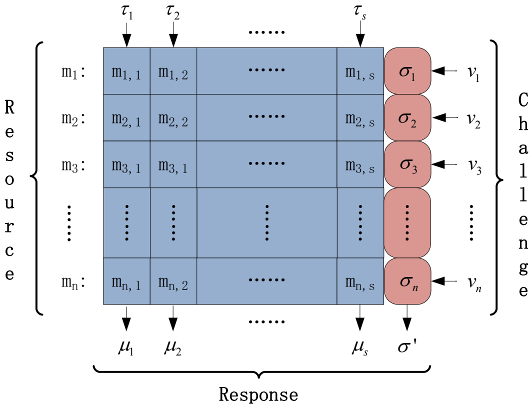


Figure 2: The fragment structure of CPDP model.

The above structure, considered as a common representation for some existing schemes [2, 4], can be converted to MAC-based, ECC or RSA schemes. By using BLS signatures and random oracle model, it is easy to design a practical CPDP scheme with the shortest homomorphic verifiable responses for public verifiability. This structure also creates favorable conditions for the architecture of CSPs.

2.4 Hash Index Hierarchy

An architecture for data storage in hybrid clouds is illustrated in Figure 3. This architecture is based on a hierarchical structure with three layers to represent the relationship among all blocks for stored resources. Three layers can be described as follows:

- First-Layer (*Express Layer*): offers an abstract representation of the stored resources;
- Second-Layer (*Service Layer*): promptly offers and manages cloud storage services; and
- Third-Layer (*Storage Layer*): directly realizes data storage on many physical devices.

This architecture naturally accommodates the hierarchical representation of file systems. We make use of a simple hierarchy to organize multiple CSP services, which involve private clouds or public clouds, by shading the differences between these clouds. In Figure 3, the resources in the Express Layer are split and stored into three CSPs in the Service Layer. In turn, each CSP fragments and stores the assigned data into the storage servers in the Storage Layer. We distinguish different CSPs by different colors, and the denotation of the Storage Layer is the same as in Figure 2. Moreover, we follow the logical order of the data blocks to organize the Storage Layer. This architecture could provide some special functions for data storage and management, e.g., there may exist an overlap among data blocks (as shown in dashed line) and discontinuous blocks (as shown on a noncontinuous color).

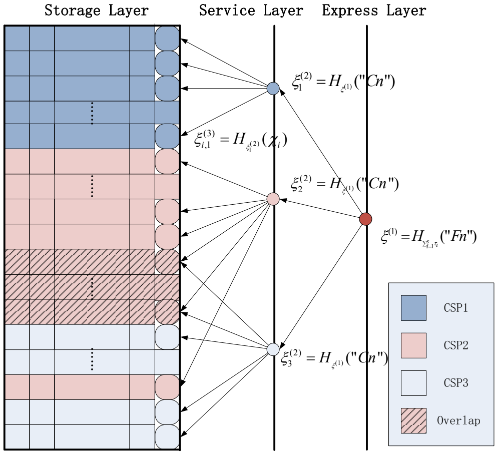

Storage Layer
Service Layer
Express Layer
$\xi_1^{(2)} = H_{\xi^{(1)}}(\text{"}Cn\text{"})$
$\xi_{i,1}^{(3)} = H_{\xi_1^{(2)}}(\chi_i)$
$\xi_2^{(2)} = H_{\xi^{(1)}}(\text{"}Cn\text{"})$
$\xi^{(1)} = H_{\sum_{i=1}^{s}\tau_i}(\text{"}Fn\text{"})$
$\xi_3^{(2)} = H_{\xi^{(1)}}(\text{"}Cn\text{"})$
CSP1
CSP2
CSP3
Overlap

Figure 3: The architecture of CPDP model.

We employ this architecture to construct a new Hash Index Hierarchy $\mathcal{H}$, which is used to replace the hash function in original PDP schemes, as follows:

- Express layer: given s random $\{\tau_i\}_{i=1}^{s}$ and the file name Fn, establishes $\xi^{(1)} = H_{\sum_{i=1}^{s}\tau_i}(\text{“}Fn\text{”})$ and makes it public for verification but makes $\{\tau_i\}_{i=1}^{s}$ secret;

- Service layer: given the $\xi^{(1)}$ and the cloud name Cn, sets $\xi_k^{(2)} = H_{\xi^{(1)}}(\text{“}Cn\text{”})$;

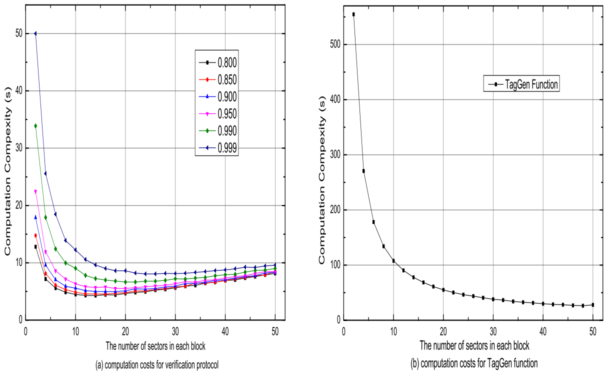
- Storage layer: given the $\xi_k^{(2)}$, a block number i, and its index record $\chi_i = \text{“}B_i||V_i||R_i\text{”}$, holds $\xi_{i,k}^{(3)} = H_{\xi_k^{(2)}}(\chi_i)$.

By using this structure, it is obvious that our CPDP scheme can also support dynamic data operations.

3. IMPLEMENTATION AND EXPERIMENTAL RESULTS

We have implemented our PDP scheme and validated the effect of dispersed secret data on private clouds and hybrid clouds. The code was written in C++ and the experiments were run on an Intel Core 2 processor with 2.16 GHz. All cryptographic operations utilize the QT and bilinear cryptographic library.

In our CPDP scheme, the client's communication overhead is not changed in contrast to common PDP scheme, and the interaction among CSPs needs $c - 1$ times constant-size communication overheads, where c is the number of CSPs in hybrid clouds. Therefore, the total of communication overheads is not significantly increased. Next, we evaluated the performance of our CPDP scheme in terms of computational overhead. For the sake of comparison, our experiments were executed in the following scenario: a fixed-size file is used to generate the tags and prove data possession under the different number of sectors s. For a 150K-Bytes file, the computational overheads of the verification protocol are shown in Figure 4(a) when the value of s is ranged from 1 to 50 and the size of sector is 20-Bytes. Moreover, there exists an optimal value of s from 15 to 25. The computational overheads of the tag generation are also shown in Figure 4(b). The results indicate that the overheads are reduced when the values of s are increased. Hence, it is necessary to select the optimal number of sectors in each block to minimize the computation costs of clients and storage service providers.



(a) computation costs for verification protocol

(b) computation costs for TagGen function

Figure 4: The experiment results of the different s for a 150K-Bytes file ($\rho = 0.01$ and $P = 0.99$).

4. CONCLUSIONS

In this paper, we addressed the construction of PDP scheme for hybrid clouds. Based on homomorphic verifiable responses and hash index hierarchy, we proposed a cooperative PDP scheme to support dynamic scalability on multiple storage servers. Our experiments showed that our schemes require a small, constant amount of overhead.

5. ACKNOWLEDGMENTS

The work of Y. Zhu, H. Wang, and Z. Hu was partially supported by the National Development and Reform Commission under project “a monitoring platform for web safe browsing”. The work of G.-J. Ahn and H. Hu was partially supported by the grants from National Science Foundation.

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. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley, Feb 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*, pages 598–609, 2007.

[3] S. Y. Ko, I. Hoque, B. Cho, and I. Gupta. On availability of intermediate data in cloud computations. In *Proc. 12th Usenix Workshop on Hot Topics in Operating Systems (HotOS XII)*, 2009.

[4] H. Shacham and B. Waters. Compact proofs of retrievability. In *ASIACRYPT*, pages 90–107, 2008.