International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Two Stage Public Auditing for Shared Data With Efficient User Revocation in the Cloud

Ajmy Abbas[1]

Computer Science and Engineering, ICET, Muvattupuzha, India

Abstract: ***With cloud computing, users can remotely store their data into the cloud and use on-demand high-quality applications by using a shared pool of configurable computing resources by using data outsourcing users are relieved from the burden of data storage and maintenance. When users put their data (of large size) on the cloud, the data integrity protection is challenging. To ensure shared data integrity can be verified publicly, users in the group need to compute signatures on all the blocks in shared data. Different blocks in shared data are generally signed by different users due to data modifications performed by different users. The straightforward method, which allows an existing user to download the corresponding part of shared data and re-sign it during user revocation, is inefficient due to the large size of shared data in the cloud. In this paper, we propose a novel public auditing mechanism for the integrity of shared data with efficient user revocation in mind. By utilizing the idea of collusion resistant multi- proxy re-signatures, we allow the cloud to resign blocks on behalf of existing users during user revocation. Collusion-resistant proxy re-signature schemes generally have two levels of signatures. In addition, a public verifier is always able to audit the integrity of shared data without retrieving the entire data from the cloud. Moreover, our mechanism is able to support batch auditing by verifying multiple auditing tasks simultaneously.***

Keywords: Public Auditing, Third Party Auditing (TPA), Shared Data, User Revocation, Proxy re-Signature.

1. Introduction

With knowledge storage and sharing services (such as Dropbox and Google Drive) provided by the cloud, folks will simply work along as a bunch by sharing knowledge with one another. a lot of specifically, once a user creates shared knowledge within the cloud, each user within the group is ready to not only access and modify shared knowledge, but conjointly share the newest version of the shared knowledge with the rest of the cluster. Though cloud suppliers promise a safer and reliable atmosphere to the users, the integrity of knowledge within the cloud should be compromised, due to the existence of hardware/software failures and human errors.

To defend the integrity of knowledge in the cloud, a number of mechanisms are planned. In these mechanisms, a signature is attached to each block in data, and also the integrity of knowledge depends on the correctness of all the signatures. One amongst the foremost significant and common options of these mechanisms is to permit a public verifier to efficiently check data integrity within the cloud while not downloading the complete knowledge, remarked as public auditing. This public verifier could be a client who would like to utilize cloud data for particular purposes (e.g., search, computation, data mining, etc.) or a third party auditor (TPA) who is able to provide verification services on data integrity to users. Most of the previous works focus on auditing the integrity of personal data. Different from these works, several recent works focus on how to preserve identity privacy from public verifiers when auditing the integrity of shared data. Unfortunately, none of the above mechanisms considers the efficiency of user revocation when auditing the correctness of shared data in the cloud.

Since shared knowledge is outsourced to the cloud and users no longer store it on native devices, an easy method to re-compute these signatures during user revocation [Figure:1] is to raise associate degree existing user (i.e., Alice) to initial transfer the blocks antecedently signed by the revoked user (i.e., Bob), verify the correctness of these blocks, then re-sign these blocks, and at last upload the new signatures to the cloud. However, this straightforward method may cost the existing user a huge amount of communication and computation resources by downloading and verifying blocks, and by re-computing and uploading signatures, especially when the number of re-signed blocks is quite large or the membership of the group is frequently changing. To make this matter even worse, existing users might access their information sharing services provided by the cloud with resource limited devices, like mobile phones that are more prevents existing users from maintaining the correctness of shared information expeditiously throughout user revocation.

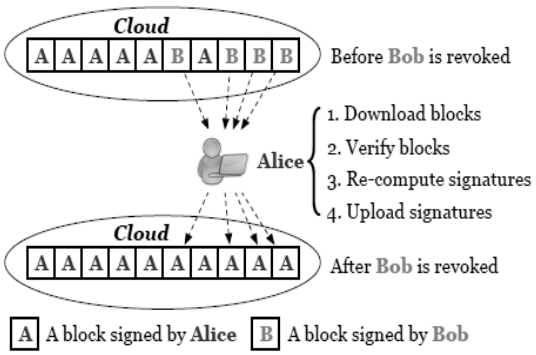

Figure 1: Alice and Bob share data in the cloud. When Bob is revoked, Alice re-signs the blocks that were previously signed by Bob with her private key.

Clearly, if the cloud might possess every user's personal key, it will simply end the re-signing task for existing users while not asking them to transfer and re-sign blocks. However, since the cloud isn't within the same trusted domain with every user within the cluster, outsourcing every user's

personal key to the cloud would introduce significant security problems. Another vital downside we need to contemplate is that the re-computation of any signature throughout user revocation ought to not have an effect on the most attractive property of public auditing — auditing data integrity in public while not retrieving the entire data.

Therefore, a way to expeditiously scale back the numerous burdens to existing users introduced by user revocation, and still enable a public champion to see the integrity of shared information while not downloading the complete information from the cloud, is a difficult task.

In this paper, we tend to propose a unique public auditing mechanism for the integrity of shared knowledge with efficient user revocation within the cloud. In our mechanism, by utilizing the thought of proxy re-signatures, once a user within the cluster is revoked, the cloud is in a position to resign the blocks, that were signed by the revoked user, with a re-signing key [Figure:2].As a result, the potency of user revocation may be considerably improved, and computation and communication resources of existing users may be simply saved. Meanwhile, the cloud, who isn't within the same sure domain with every user, is merely ready to convert a signature of the revoked user into a signature of associate in nursing existing user on identical block, however it cannot sign quirky blocks on behalf of either the revoked user or associate in nursing existing user. By planning new proxy re-signature themes with nice properties, that ancient proxy resignatures do not have, our mechanism is often ready to check the integrity of shared knowledge while not retrieving the entire knowledge from the cloud.

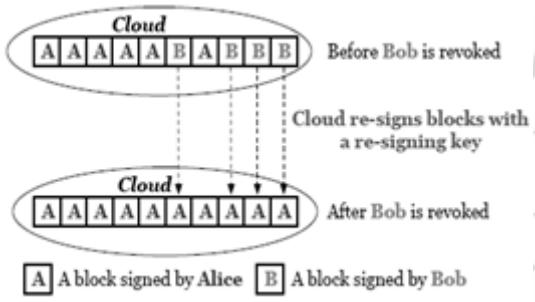

Figure 2: When Bob is revoked, the cloud re-signs the blocks that were previously signed by Bob with a resigning key.

If a revoked user is able to collude with the cloud, who possesses are-signing key, then the cloud and revoked user together are able to easily reveal the private key of a existing user. How to design such type of collusion-resistant proxy re-signature schemes while also supporting public auditing (i.e., blockless verifiability and non malleability) remains to be seen. Essentially, since collusion-resistant proxy re-signature schemes generally have two levels of signatures where the two levels of signatures are in different forms and need to be verified differently, achieving blockless verifiability on both of the two levels of signatures and verifying them together in a public auditing mechanism is challenging. Extend our mechanism into the multi-proxy model to reduce the prospect of the misuse on re-signing keys within the cloud and improve the responsibility of the whole mechanism.

1.1 Proxy Re-Signatures

It includes six phases: KeyGen, ReKey, Sign,ReSign, ProofGen, ProofVerify. In KeyGen, every user in the group generates his/her public key and private key. In ReKey, the cloud computes are-signing key for each pair of users in the group. When the original user creates shared data in the cloud, he/she computes a signature on each block as in Sign. After that, if a user in the group modifies a block in shared data, the signature on the modified block is also computed as in Sign. In ReSign, a user is revoked from the group, and the cloud re-signs the blocks, which were previously signed by this revoked user, with a re-signing key. The verification on data integrity is performed via a challenge-and-response protocol between the cloud and a public verifier. More specifically, the cloud is able to generate a proof of possession of shared data in ProofGen under the challenge of a public verifier. In Proof-Verify, a public verifier is able to check the correctness of a proof responded by the cloud.In ReSign, without loss of generality, we assume that the cloud always converts signatures of a revoked user into signatures of the original user. The reason is that the original user acts as the group manager, and we assume he/she is secure in our mechanism. Another way to decide which re-signing key should be used when a user is revoked from the group, is to ask the original user to create a priority list (PL). Every existing user's id is in the PL and listed in the order of re-signing priority. When the cloud needs to decide which existing user the signatures should be converted into, the first user shown in the PL is selected.

Let G_1 and G_2 be two groups of order p,g be a generator of G_1 , e:$G_1 \times G_2 \rightarrow G_2$ be a bilinear map, ω be another generator of G_1. The global parameters are$(e,p,G_1,G_2,g, \omega,H)$,where H is a hash function with $H:\{0,1\}^* \rightarrow G_1$.The total number of blocks in shared data is n,and shared data is described as $M=(m_1,\ldots,m_n)$.The total number of users in the group is d.

KeyGen: User u_i generates a random $\pi_i \in Z^*_p$,and outputs public key $pk_i=g^{\pi}{}_i$ and private key $sk_i = \pi_i$.Without loss of generality, we assume user u_1 is the original user, who is creator of shared data. The original user also creates a user list, which contains ids of all the users in the group.

Rekey: The cloud generates a re-signing key $rk_{i\rightarrow j}$ as follows: The cloud generates a random $r \in Z_p$ and sends it to user u_i. user u_i sends r/π_i to user u_j , where $sk_i = \pi_i$. user u_j sends $r\,\pi_j / \pi_i$ to cloud, where $sk_j = \pi_j$.The cloud recovers $rk_{i\rightarrow j} = \pi_j / \pi_i \in Z^*_p$

Sign: Given private key $sk_i = \pi_i$,bock $m_k \in Z_p$ and its block identifier id_k ,where $k \in [1,n]$,user u_i outputs the signature on block m_k as:

$\sigma_k = (H(id_k)\ \omega^{mk})^{\pi i} \in G_1$

Resign : Given re-signing key $rk_{i\rightarrow j}$,public key pk_i,signature σ_k, block m_k block identifier id_k ,the cloud first checks that $e(\sigma_k,g) = e(H(id_k)\ \omega^{mk}\ , pk_i)$.If the verification resut 0,the cloud outputs null;otherwise it outputs

$\sigma'_k = \sigma_k^{rki\rightarrow j} = (H(id_k)\ \omega^{m}{}_k)^{\pi}{}_i{}^{\pi}{}_j{}^{/\pi}{}_i = (H(id_k)\ \omega^{m}{}_k)^{\pi}{}_j$

After the re-signing,the original user removes user u_i's id from UL and signs the new UL.By leveraging Shamir Secret Sharing s multiple proxies is used, each re-signing key is divided into s pieces and each piece is distributed to one proxy. These multiple proxies belong to the same cloud, but store and manage each piece of a re-signing key independently. since collusion-resistant proxy re-signature schemes generally have two levels of signatures (i.e., the first level is signed by a user and the second level is re-signed by the proxy), where the two levels of signatures are in different forms and need to be verified differently, achieving blockless verifiability on both of the two levels of signatures and verifying them together in a public auditing mechanism is challenging.

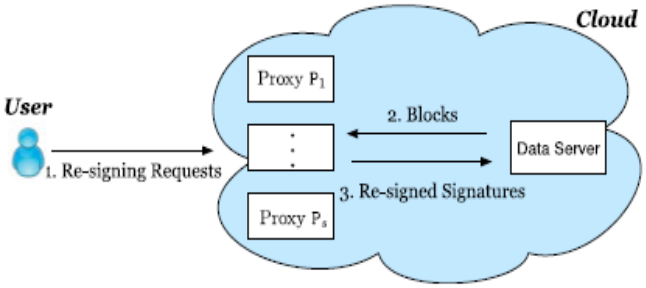

Figure 3: Multiple re-signing proxies in the cloud

2. Related Work

2.1 Provable Data Possession at Untrusted Stores

The model for provable data possession (PDP) that allows a client that has stored data at an untrusted server to verify that the server possesses the original data without retrieving it. The model generates probabilistic proofs of possession by sampling random sets of blocks from the server, which drastically reduces I/O costs. The client maintains a constant amount of metadata to verify the proof. The challenge/response protocol transmits a small, constant amount of data, which minimizes network communication. Thus, the PDP model for remote data checking supports large data sets in widely-distributed storage systems. It present two provably-secure PDP schemes that are more efficient than previous solutions, even when compared with schemes that achieve weaker guarantees. In particular, the overhead at the server is low (or even constant), as opposed to linear in the size of the data. Experiments using our implementation verify the practicality of PDP and reveal that the performance of PDP is bounded by disk I/O and not by cryptographic computation.

2.2 Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing

Using cloud storage, users can remotely store their data and enjoy the on-demand high-quality applications and services from a shared pool of configurable computing resources, without the burden of local data storage and maintenance. However, the fact that users no longer have physical possession of the outsourced data makes the data integrity protection in cloud computing a formidable task, especially for users with constrained computing resources. Moreover, users should be able to just use the cloud storage as if it is local, without worrying about the need to verify its integrity. Thus, enabling public auditability for cloud storage is of critical importance so that users can resort to a third-party auditor (TPA) to check the integrity of outsourced data and be worry free. To securely introduce an effective TPA, the auditing process should bring in no new vulnerabilities toward user data privacy, and introduce no additional online burden to user. It deals with a secure cloud storage system supporting privacy preserving public auditing. It further extend our result to enable the TPA to perform audits for multiple users simultaneously and efficiently.

2.3 Compact Proofs of Retrievability

In a proof-of-retrievability system, a data storage center must prove to a verifier that he is actually storing all of a clients data. The central challenge is to build systems that are both efficient and provably secure that is, it should be possible to extract the clients data from any prover that passes a verification check. In this paper, we give the first proof-of-retrievability schemes with full proofs of security against arbitrary adversaries in the strongest model. Our first scheme, built from BLS signatures and secure in the random oracle model, features a proof-of-retrievability protocol in which the clients query and servers response are both extremely short. This scheme allows public verifiability: anyone can act as a verifier, not just the file owner. Our second scheme, which builds on pseudorandom functions (PRFs) and is secure in the standard model, allows only private verification. It features a proof-of-retrievability protocol with an even shorter servers response than our first scheme, but the clients query is long. Both schemes rely on homomorphic properties to aggregate a proof into one small authenticator value.

2.4 Proxy Provable Data Possession in Public Clouds

Recently, cloud computing rapidly expands as an alternative to conventional computing due to it can provide a flexible, dynamic and resilient infrastructure for both academic and business environments. In public cloud environment, the client moves its data to public cloud server (PCS) and cannot control its remote data. Thus, information security is an important problem in public cloud storage, such as data confidentiality, integrity, and availability. In some cases, the client has no ability to check its remote data possession, such as the client is in prison because of committing crime,on the ocean-going vessel, in the battlefield because of the war, and so on. It has to delegate the remote data possession checking task to some proxy. It deals with study proxy provable data possession (PPDP). In public clouds, PPDP is a matter of crucial importance when the client cannot perform the remote data possession checking. The PPDP system model, the security model, and the design method is based on the bilinear pairing technique.

2.5 Oruta: Privacy-Preserving Public Auditing for Shared Data in the Cloud

With cloud storage services, it is commonplace for data to be not only stored in the cloud, but also shared across multiple

users. However, public auditing for such shared data while preserving identity privacy remains to be an open challenge. This paper deals with the first privacy-preserving mechanism that allows public auditing on shared data stored in the cloud. In particular, we exploit ring signatures to compute the verification information needed to audit the integrity of shared data. With our mechanism, the identity of the signer on each block in shared data is kept private from a third party auditor (TPA), who is still able to publicly verify the integrity of shared data without retrieving the entire file. However, the size of signatures and verification time linearly increase with the number of users in the group. It is not suitable for large groups! Besides, the identity of a signer cannot be traced (ring signatures).

3. Conclusion

Cloud computing is world's biggest innovation which uses advanced computational power and improves data sharing and data storing capabilities. It increases the ease of usage by giving access through any kind of internet connection. As every coin has two sides it also has some drawbacks. Privacy security is a main issue for cloud storage. To ensure that the risks of privacy have been mitigated a variety of techniques that may be used in order to achieve privacy. This paper shows some privacy techniques and different methods for overcoming the issues in privacy on untrusted data stores in cloud computing. a new public auditing mechanism for shared data with efficient user revocation in the cloud. When a user in the group is revoked, then allow the semi-trusted cloud to re-sign blocks that were signed by the revoked user with proxy re-signatures. The cloud can improve the efficiency of user revocation, and existing users in the group can save a significant amount of computation and communication resources during user revocation.

4. Acknowledgment

The Author would like to thank Mr.Shanavas K.A. Assistant Professor, Department of Information Technology, Ilahia College of Engineering and Technology, Muvattupuzha for his moral and technical support.

References

[1] 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 (CCS07), pp. 598-610, 2007.
[2] H. Shacham and B. Waters, "Compact Proofs of Retrievability," Proc. 14th Intl Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT08), pp. 90- 107, 2008.
[3] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing," Proc. IEEE INFOCOM, pp. 525-533, 2010.
[4] B. Wang, B. Li, and H. Li, "Knox: Privacy-Preserving Auditing for Shared Data with Large Groups in the Cloud,"Proc. 10th Intl Conf. Applied Cryptography and Network Security (ACNS12), pp. 507-525, June 2012.
[5] B.Wang, B. Li, and H. Li, "Oruta: Privacy-Preserving Public Auditing for Shared Data in the Cloud," Proc. IEEE CLOUD, pp. 295-302, 2012.
[6] C. Wang, Q. Wang, K. Ren, and W. Lou, "Towards Secure and Dependable Storage Services in Cloud Computing," IEEE Trans. Services Computing, vol. 5, no.2, pp. 220-232, Jan. 2012.
[7] B. Wang, B. Li, and H. Li, "Public Auditing for Shared Data with Efficient User Revocation in the Cloud," Proc. IEEE INFOCOM, pp. 2904-2912, 2013
[8] H. Wang, "Proxy Provable Data Possession in Public Clouds," IEEE Trans. Services Computing, vol. 6, no. 4, pp. 551-559, Oct.- Dec. 2013
[9] X. Liu, Y. Zhang, B. Wang, and J. Yan, "Mona: Secure Multi- Owner Data Sharing for Dynamic Groups in the Cloud," IEEE Trans. Parallel and Distributed Systems (TPDS13), vol. 24, no. 6, pp. 1182-1191, June 2013.

Author Profile

Ajmy Abbas received the Bachelor of Technology degree in Computer Science and Engineering from Mahatma Gandhi University, Kerala. She is currently doing Master of Technology degree in Computer Science and Engineering with Specialization in Information Systems from Mahatma Gandhi University, Kerala.