REVIEW ON ANDA: AUDING FOR SHARED DATA IN CLOUD WITH EFFICIENT USER REVOCATION

Gade Supriya, Harshal Mahajan
Siddhant College of Engineering, Sudumbare, Pune, India

Abstract- Now days, many organizations outsource data storage to the cloud such that a member of an organization (data owner) can easily share data with other members (users). With data storage and sharing services in the cloud, users can easily modify and share data as a group. 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. For security reasons, once a user is revoked from the group, the blocks which were previously signed by this revoked user must be re-signed by an existing user. However, previous methods either unnecessarily reveal the identity of a data owner to the entrusted cloud or any public verifiers, or introduce significant overheads on verification metadata for preserving anonymity. 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 proxy re-signatures, we allow the cloud to re-sign blocks on behalf of existing users during user revocation, so that existing users do not need to download and re-sign blocks by themselves. In addition, a public verifier is always able to audit the integrity of shared data without retrieving the entire data from the cloud, even if some part of shared data has been re-signed by the cloud. Moreover, our mechanism is able to support batch auditing by verifying multiple auditing tasks simultaneously

***Index Terms*- Public auditing, shared data, user revocation, cloud computing.**

I. INTRODUCTION

Many organizations outsource their large-scale data storage to the cloud for saving the cost in maintaining in-house storage. With cloud storage service, the members of an organization can share data with other members easily by uploading their data to the cloud. With data storage and sharing services (such as Drop box and Google Drive) provided by the cloud, people can easily work together as a group by sharing data with each other. More specifically, once a user creates shared data in the cloud, every user in the group is able to not only access and modify shared data, but also share the latest version of the shared data with the rest of the group. Although cloud providers promise a more secure and reliable environment to the users, the integrity of data in the cloud may still be compromised, due to the existence of hardware/software failures and human errors. However, the general method for protecting data integrity will conflict with another significant concern of data owners — identity privacy, or anonymity. Specifically, if digital signatures are used to serve as verification metadata, they can only be verified with a data owner's public key. To protect the integrity of data in the cloud, a number of mechanisms have been proposed. In these mechanisms, a signature is attached to each block in data, and the integrity of data relies on the correctness of all the signatures. One of the most significant and common features of these mechanisms is to allow a public verifier to efficiently check data integrity in the cloud without downloading the entire data, referred to as public auditing (or denoted as Provable Data Possession). 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.

With shared data, once a user modifies a block, she also needs to compute a new signature for the modified block. Due to the modifications from different users, different blocks are signed by different users. For security reasons, when a user leaves the group or misbehaves, this user must be revoked from the group. As a result, this revoked user should no longer be able to access and modify shared data, and the signatures generated by this revoked user are no longer valid to the group. Therefore, although the content of shared data is not changed during user revocation, the blocks, which were previously signed by the revoked user, still need to be re-signed by an existing user in the group. As a result, the integrity of the entire data can still be

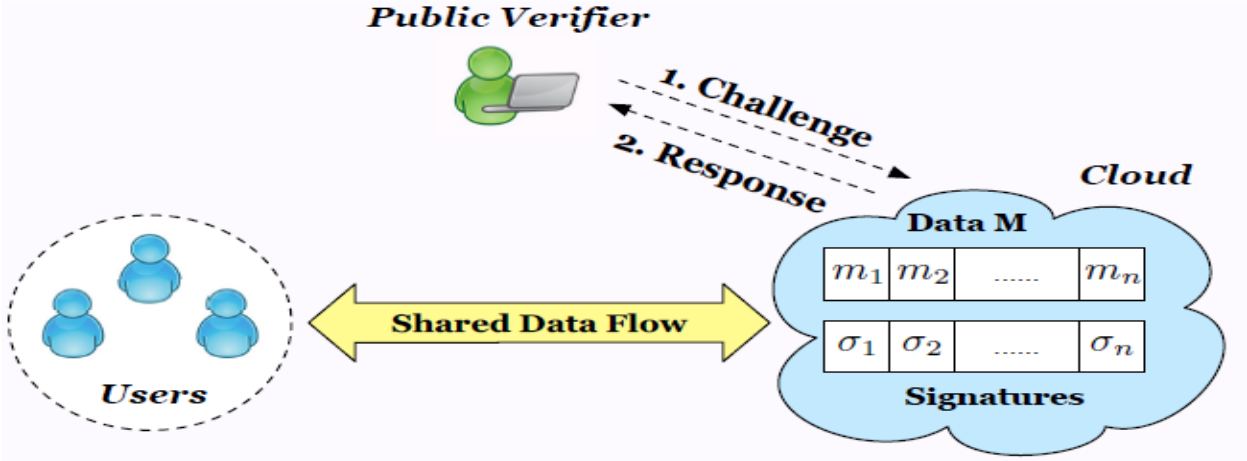

Fig. 1. The system model includes the cloud, the public verifier, and users.

verified with the public keys of existing users only.

II. PROBLEM STATEMENT

In this section, we describe the system and security model, and illustrate the design objectives of our proposed mechanism.

II-A. System and Security Model:

As illustrated in Fig. 1, the system model in this paper includes three entities: the cloud, the public verifier, and users (who share data as a group). The cloud offers data storage and sharing services to the group. The public verifier, such as 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 can provide verification services on data integrity aims to check the integrity of shared data via a challenge-and response protocol with the cloud. In the group, there is one original user and a number of group users. The original user is the original owner of data. This original user creates and shares data with other users in the group through the cloud. Both the original user and group users are able to access, download and modify shared data. Shared data is divided into a number of blocks.

A user in the group can modify a block in shared data by performing an insert, delete or update operation on the block. In this paper, we assume the cloud itself is semi-trusted, which means it follows protocols and does not pollute data integrity actively as a malicious adversary, but it may lie to verifiers about the incorrectness of shared data in order to save the reputation of its data services and avoid losing money on its data services. In addition, we also assume there is no collusion between the cloud and any user during the design of our mechanism. Generally, the incorrectness of share data under the above semi trusted model can be introduced by hardware/software failures or human errors happened in the cloud. Considering these factors, users do not fully trust the cloud with the integrity of shared data. To protect the integrity of shared data, each block in shared data is attached with a signature, which is computed by one of the users in the group. Specifically, when shared data is initially created by the original user in the cloud, all the signatures on shared data are computed by the original user. After

that, once a user modifies a block, this user also needs to sign the modified block with his/her own private key. By sharing data among a group of users, different blocks may be signed by different users due to modifications from different users. When a user in the group leaves or misbehaves, the group needs to revoke this user. Generally, as the creator of shared data, the original user acts as the group manager and is able to revoke users on behalf of the group. Once a user is revoked, the signatures computed by this revoked user become invalid to the group, and the blocks that were previously signed by this revoked user should be re-signed by an existing user's private key, so that the correctness of the entire data can still be verified with the public keys of existing users only.

II-B. Design Objectives:

Our proposed mechanism should achieve the following properties: (1) Correctness: The public verifier is able to correctly check the integrity of shared data. (2) Efficient and Secure User Revocation: On one hand, once a user is revoked from the group, the blocks signed by the revoked user can be efficiently re-signed. On the other hand, only existing users in the group can generate valid signatures on shared data, and the revoked user can no longer compute valid signatures on shared data. (3) Public Auditing: The public verifier can audit the integrity of shared data without retrieving the entire data from the cloud, even if some blocks in shared data have been re-signed by the cloud. (4) Scalability: Cloud data can be efficiently shared among a large number of users, and the public verifier is able to handle a large number of auditing tasks simultaneously and efficiently.

III. PRELIMINARIES

III-A. Bilinear maps :

Let G1, G2 and G*T* be three multiplicative cyclic groups of prime order *p*, *g*1 and *g*2 be the generators of G1 and G2. ψ is a computable isomorphism from *G*2 to *G*1, with $\psi(g2) = g1$. The map e : G1 × G2 → G*T* is said to be an admissible bilinear pairing if the following conditions hold true.

(1) e is bilinear, i.e. $e(g1^a, g2^b) = e(g1, g2)^{ab}$ For all $a, b \in Zp$.

(2) e is non-degenerate, i.e. $e(\mathrm{g1}, \mathrm{g2}) \neq 1G_T$

(3) e is efficiently computable.

III-B. Blind Signatures:

Blind signatures, first proposed by Chaum, form a special type of signatures where the message owner and the signer are different parties. More specifically, the message owners choose a blinding factor to blind the content of her message and send the blinded message to the signer. After received the blinded message, the signer generates a signature on the blinded message and returns it to the message owner. The message owner is able to recover and output a regular signature on the original message based on the result returned by the signer and the blinding factor. The *blindness* properties require that the signer cannot learn the content of the original message during the generation of a signature. For *unlink ability*, it requires that the signer cannot link a blinded message/signature to its corresponding unblended form.

III-C. Shamir Secret Sharing:

A (w, t)-Shamir secret sharing scheme, where w = (2t − 1), is able to divide a secret s into w pieces in such a way that this secret s can be easily recovered from any t pieces, while the knowledge of any (t − 1) pieces reveals absolutely no information about this secret s. The essential idea of a (w, t)-Shamir secret sharing scheme is that, a number of t points define a polynomial of degree (t−1). Suppose we want to share the secret s 2 Zp. We set a0 = s and define the following polynomial f(x) = at−1xt−1 + ▪ ▪ ▪ + a1x + a0, (1) by picking at−1, ▪ ▪ ▪, a1 uniformly at random from Zp. Each piece of the share is actually a point of polynomial f(x), for example, (xi, f (xi)). The secret s can be recovered by at least a number of t points of polynomial f(x) with Lagrange polynomial interpolation. Shamir secret sharing is generally used in key management schemes and secures multi computation.

IV. CONSTRUCTION OF ANDA

Anda includes six algorithms: **KeyGen**, **ReKey**, **Sign**, **ReSign**, **ProofGen**, **and ProofVerify**.

- In **KeyGen**, every user in the group generates his/her public key and private key
- In **ReKey**, the cloud computes a re-signing key for each pair of users in the group

- In **Sign,** When the original user creates shared data in the cloud, he/she computes a signature on each block
- 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 resigning key
- in **ProofGen** under the challenge of a public verifier the cloud is able to generate a proof of possession of shared data
- In **ProofVerify**, a public verifier is able to check the correctness of a proof responded by the cloud

V. EXTENSION OF ANDA

In this section, we will utilize several different methods to extend our mechanism in terms of detection probability, scalability and reliability.

V-A. Detection probability:

As presented in our mechanism, a verifier selects a number of random blocks instead of choosing all the blocks in shared data, which can improve the efficiency of auditing.

V-B. Scalability:

To improve the scalability of our proposed mechanism by reducing the total number of resigning keys in the cloud and enabling batch auditing for verifying multiple auditing tasks simultaneously.

- Reduce the Number of Re-signing Keys: As described in Anda, the cloud needs to establish and maintain a re-signing key for each pair of two users in the group. Since the number of users in the group is denoted as d, the total number of re-signing keys for the group is $d(d-1)/2$. Clearly, if the cloud data is shared by a very large number of users, e.g. $d = 200$, then the total number of re-signing keys that the cloud has to securely store and manage is 19, 900, which significantly increases the complexity of key management in cloud.
- Batch Auditing for Multiple Auditing Tasks: to improve the scalability of our public auditing mechanism in such cases, we can further extend Aanda to support batch auditing by utilizing the properties of bilinear maps. With batch auditing, a public verifier can perform multiple auditing tasks simultaneously. Compared to the batch auditing in,where the verification metadata (i.e, signatures) in each auditing task are generated by a single user, our batch auditing method needs to perform on multiple auditing tasks where the verification metadata in each auditing.

V-C. Reliability of Aanda:

In our mechanism, it is very important for the cloud to securely store and manage the re-signing keys of the group, so that the cloud can correctly and successfully convert signatures from a revoked user to an existing user when it is necessary. However, due to the existence of internal attacks, simply storing these re-signing keys in the cloud with a single re-signing proxy may sometimes allow inside attackers to disclose these re-signing keys and arbitrarily convert signatures on shared data, even no user is revoking from the group. Obviously, the arbitrary misuse of re-signing keys will change the ownership of corresponding blocks in shared data without users' permission, and affect the integrity of shared data in the cloud. To prevent the arbitrary use of re-signing keys and enhance the reliability of our mechanism, we propose an extended version of our mechanism, denoted as Anda, in the multi-proxy model.

VI. CONCLUSION

In this paper, we introduce what we believe is the right approach to achieve anonymity in storing data to the cloud with publicly-verifiable data-integrity in mind. Our approach decouples the anonymous protection mechanism from the provable data possession mechanism via the use of security mediator. Our solution not only minimizes the computation and bandwidth requirement of this mediator, but also minimizes the trust placed on it in terms of data privacy and identity privacy. The efficiency of our system is also empirically demonstrated. When a user in the group is revoked, we allow the semi-trusted cloud to re-sign blocks that were signed by the revoked user with proxy re-signatures. Experimental results show that 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.

REFERENCES

[1]. B. Wang, B. Li, and H. Li, "Public Auditing for Shared Data with Efficient User Revoation in the Cloud," IEEE Transaction on service computing 2014.

[2]. B. Wang, B. Li, and H. Li, "Public Auditing for Shared Data with Efficient User Revoation in the Cloud," in the Proceedings of IEEE INFOCOM 2013, 2013, pp. 2904–2912.

[3]. 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, "A View of Cloud Computing," Communications of the ACM, vol. 53, no. 4, pp. 50–58, Apirl 2010.

[4]. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores,"in the Proceedings of ACM CCS 2007, 2007, pp. 598–610.

[5]. H. Shacham and B. Waters, "Compact Proofs of Retrievability," in the Proceedings of ASIACRYPT 2008. Springer-Verlag,2008,pp. 90–107.

[6]. C. Wang, Q. Wang, K. Ren, and W. Lou, "Ensuring Data Storage Security in Cloud Computing," in the Proceedings of ACM/IEEE IWQoS 2009, 2009, pp. 1–9.

[7]. Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling Public Verifiability and Data Dynamic for Storage Security in Cloud Computing," in the Proceedings of ESORICS 2009. Springer-Verlag, 2009, pp. 355–370.

[8]. C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing," in the Proceedings of IEEE INFOCOM 2010, 2010, pp. 525–533.