IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 16, Issue 6, Ver. VI (Nov – Dec. 2014), PP 68-72
www.iosrjournals.org

Advanced Client Repudiation Diverge Auditor in Public Cloud

[1]Ms.Nita R. Mhaske,[2]Prof. S.M.Rokade
[1]*student , Master of Engineering, Dept. of Computer Engineering Sir Visvesvaraya Institute of Technology, Chincholi, Sinner*
[2]*Head Of Department of Computer Engineering, Sir Visvesvaraya Institute of Technology, Chincholi, Sinner*

Abstract: *With data storage and sharing services provided by cloud , people work together by sharing data as a group . After creating group and shared data in the cloud user in the group is able to access and modified and also share the latest updated data with the rest of the 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 repudiated from the group, the blocks which were previously signed by this repudiated user must be re-signed by an existing user. The straight forward method, which allows an existing user to download the corresponding part of shared data and re-sign it during user repudiation, is inefficient due to the large size of shared data in the cloud. In this paper, we propose a novel diverge auditing mechanism for the integrity of shared data with efficient user repudiation . By utilizing the idea of proxy re-signatures, we allow the cloud to re-sign blocks on behalf of existing users during user repudiation, 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.*

Keywords: *Diverge auditing, shared data, user repudiation, cloud computing.*

I. Introduction

With data storage and sharing services provided by cloud ,people work together by sharing data as a group .after creating group and shared data in the cloud user in the group is able to access and modified and also share the latest updated data with the rest of the group.As cloud providers promised a more secure and reliable environment to the users, still there may be problem with integrity of the data in the cloud due to existence of hardware/software failures and human errors [3]. To protect the integrity , number of mechanisms have been proposed. We discuss a mechanisms which include , a signature is attached to each block in data and the integrity of data relies on the correctness of all the signatures. As shown in figure 3 we public verifier to efficiently check data integrity in the cloud without downloading the entire data, referred to as diverge auditing[3].This public verifier could be a client who would like to utilize cloud data for particular purposes or a third party auditor (TPA) who is able to provide verification services on data integrity to users. Most of the previous mechanism proposed not consider the efficiency of user repudiation when auditing the correctness of shared data in the cloud. With shared data , once user modified a block he need to compute a new signature again for a modified block ,due to that different blocks are signed by different users. For security reasons, when a user leaves the group or misbehaves, this user must be repudiated from the group. As a result, thisrepudiated user should no longer be able to access and modify shared data, and the signatures generated by this repudiated user are no longer valid to the group . As a result, the integrity of the entire data can still be verified with the public keys of existing users only. Since shared data is outsourced to the cloud and users no longer store it on local devices, a straightforward method to re-compute these signatures during user repudiation (as shown in Figure1) is to ask an existing user (i.e., Alice) to first download the blocks previously signed by the repudiated user (i.e., Bob), verify the correctness of these blocks, then re-sign these blocks, and finally 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 membershipof the group is frequently changing.

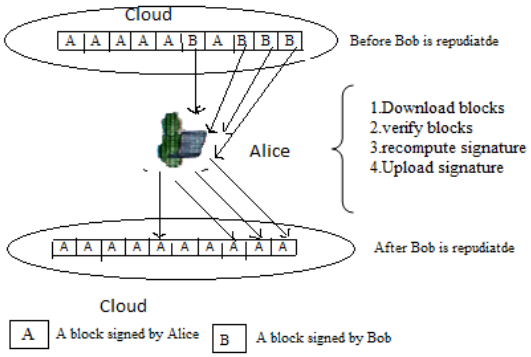

Figure1. Alice and Bob share data in the cloud. When Bob is repudiated, Alice re-signs the blocks that were previously signed by Bob with his private key.

Clearly, if the cloud could possess each user's private key, it can easily finish the re-signing task for existing users without asking them to download and re-sign blocks. However, since the cloud is not in the same trusted domain with each user in the group, outsourcing every user's private key to the cloud would introduce significant security issues. Another important problem we need to consider is that the re-computation of any signature during user repudiation should not affect the most attractive property of diverge auditingauditing data integrity publicly without retrieving the entire data. Therefore, how to efficiently reduce the significant burden to existing users introduced by user repudiation, and still allow a public verifier to check the integrity of shared data without downloading the entire data from the cloud, is a challenging task. a novel diverge auditing mechanism for the integrity of shared data with efficient user repudiation in the cloud. In this, by utilizing the idea of proxy re-signatures [4], once a user in the group is repudiated, the cloud is able to resign the blocks, which were signed by the repudiated user, with a re-signing key (as presented in Figure2). As a result, the efficiency of user repudiation can be significantly improved, and computation and communication resources of existing users can be easily saved. Meanwhile, the cloud, who is not in the same trusted domain with each user, is only able to convert a signature of the repudiated user into a signature of an existing user on the same block, but it cannot sign arbitrary blocks on behalf of either the repudiated user or an existing user.

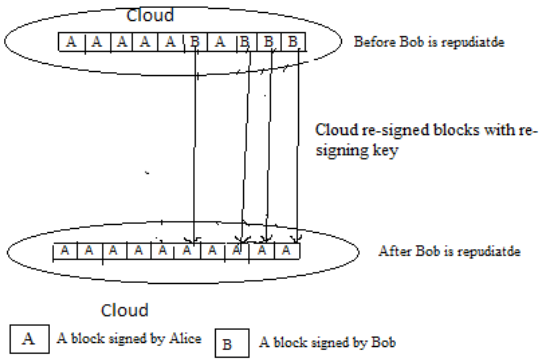

Figure 2. When Bob is repudiated, the cloud re-signs the blocks with Alice which was previously signed by Bob with a resigning key.

II. Literature Survey

G.Ateniese,R.Burns,R.curtmola[3] proposed mechanism ,that allowed the client who stored the data at an untrusted server to verify that the server possesses the original data without retrieving it. They allow to verify data possession without having access to the actual data file. C.Wang,Q.Wang,K.Ren[5] proposed mechanism focusing on cloud data storage security which effect on quality of service, supports secure and efficient dynamics operations on data blocks including data update,delete and append. H.Shacham & B.Waters [4]proposed two proofs for retrievability schemes for full proofs of security against arbitrary adversaries in the strongest scheme which is Juels and Kaliski and secondly scheme with private verifiability. H.Wang [7] proposed system for proxy provable data possession for the purpose when the client cannot perform the remote data possession checking. System was based on bilinear pairing technique. B.Wang,B.Li, H.Li,F.Li [10] proposed first certificateless diverge auditing mechanism for verifying data integrity in the untrusted cloud is based on CDH assumption and DL assumption.C.Wang,Q.Wang,K.Ren,W.Lou[6] proposed mechanism which

consist public key based homomorphic authenticator and uniquely integrate it with random mask technique to achieve a privacy preserving diverge auditing system for cloud data storage security. B.Wang,B.Li,H.Li [8] proposed mechanism a privacy preserving mechanism that allows diverge auditing on shared data stored in the cloud by identifying the singer on each block in shared data is kept private from a Third Party Auditor (TPA).

Existing System:

Since shared data is outsourced to the cloud and users no longer store it on local devices, a straightforward method to re-compute these signatures during user repudiation (as shown in Figure. 1) is to ask an existing user (i.e., Alice) to first download the blocks previously signed by the repudiated user (i.e., Bob), verify the correctness of these blocks, then re-sign these blocks, and finally 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 may access their data sharing services provided by the cloud with resource limited devices, such as mobile phones, which further prevents existing users from maintaining the correctness of shared data efficiently during user repudiation. Clearly, if the cloud could possess each user's private key, it can easily finish the re-signing task for existing users without asking them to download and re-sign blocks. However, since the cloud is not in the same trusted domain with each user in the group, outsourcing every user's private key to the cloud would introduce significant security issues. Another important problem we need to consider is that the re-computation of any signature during user repudiation should not affect the most attractive property of diverge auditing — auditing data integrity publicly without retrieving the entire data. Therefore, how to efficiently reduce the significant burden to existing users introduced by user repudiation, and still allow a public verifier to check the integrity of shared data without downloading the entire data from the cloud, is a challenging task.

Proposed System:

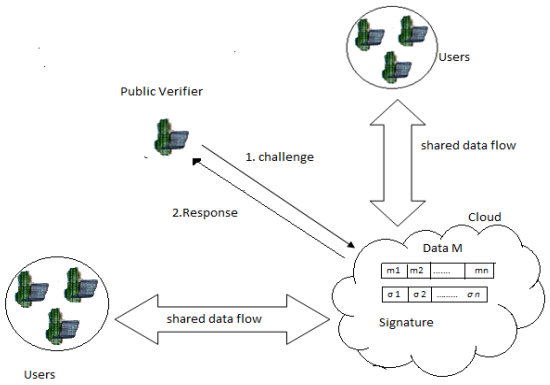

Figure 3: block diagram of proposed system

Block diagram includes three entities: the cloud, the public verifier, and two or many 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 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.

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 computedby the original user. After that, once a user modifies a block, this user also needs to sign the modified block with users 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 repudiate this user. Generally, as the creator of shared data, the original user acts as the group manager and is able to repudiate users on behalf of the group. Once a user is repudiated, the signatures computed by this repudiated user become invalid to the group, and the blocks that were previously signed by this repudiated 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.

Flowchart:
Flowchart for the proposed system is:

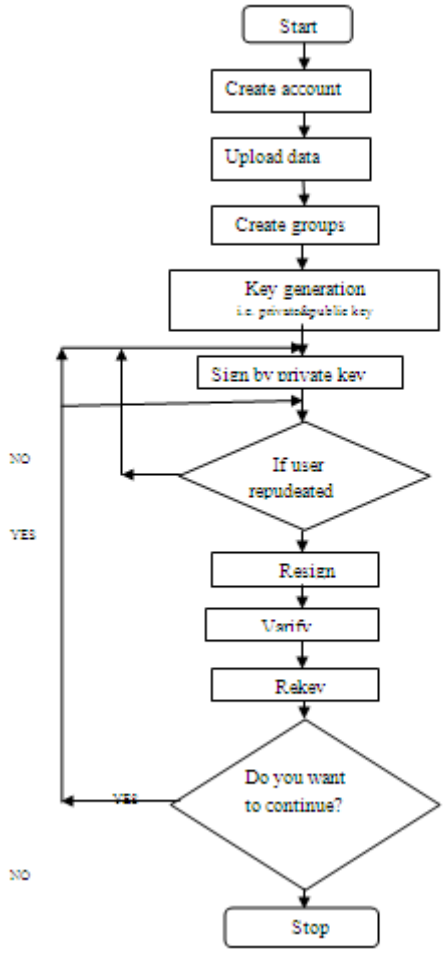

Figure 4: Flowchart for proposed system

Proposed system consist key generation, rekey generation, signing, resigning, proof of key generation, verification of rekey generation.

1. In **key generation**,every user in the group generates public key and private key.
2. In **rekey generation**, the cloud computes a re-signing key for each pair of users in the group, for the purpose of repudiation.
3. In **signing**, when the original user creates shared data in the cloud, user 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.
4. In **resigning**, a user is repudiated from the group, and the cloud re-signs the blocks, which were previously signed by this repudiated user, with a resigning key. By using resigning cloud always converts signatures of a repudiated user into signatures of the original user. This will done to secure our mechanism.
5. In **proof of key generation** ,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.
6. In **verification of rekey generation**, a public verifier is able to check the correctness of a proof responded by the cloud.

III. Conclusion

We are trying to implement a diverge auditing mechanism for shared data with efficient user repudiation in public cloud. When a user in the group is repudiated, we allow the cloud to resign the block by existing user that was signed by repudiated user with proxy resignatures. By implementing this system we can improve the efficiency of user repudiation and also saving of computation and communication resources during user repudiation.

References

[1]. B. Wang, B. Li, Member, H. Li, "Panda: Public Auditing for Shared Data with Efficient User Revocation in the Cloud " in the Proceedings of IEEE INFOCOM 2014, 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, an 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. Springe 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/IEEEIWQoS 2009, 2009, pp. 1–9.

[7]. 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.

[8]. H. Wang, "Proxy Provable Data Possession in Public Clouds," IEEE Transactions on Services Computing, accepted.

[9]. B. Wang, B. Li, and H. Li, "Oruta: Privacy-Preserving Public Auditing for Shared Data in the Cloud," in the Proceedings of IEEE Cloud 2012, 2012, pp. 295–302.

[10]. B. Wang, S. S. Chow, M. Li, and H. Li, "Storing Shared Data on the Cloud via Security-Mediator," in Proceedings of IEEE ICDCS 2013, 2013.

[11]. B.Wang, B. Li, and H. Li, "Certificateless Public Auditing for Data Integrity in the Cloud," in Proceedings of IEEE CNS 2013, 2013, pp. 276–284.