Secure Cloud Storage: A Protocol for Remote Data Integrity Checking

Naganandhini.S,
Assistant Professor,
Department of CSE,
PSNA College of Engg & Tech,
Dindigul,Tamilnadu, India.
nandhu.be2010@gmail.com

Shenbagavalli.S.T.,
Assistant Professor,
Department of CSE,
PSNA College of Engg & Tech,
Dindigul,Tamilnadu, India.
stshenbagavalli@gmail.com

Shanmugapriya.M.,
Assistant Professor,
Department of CSE,
PSNA College of Engg & Tech,
Dindigul,Tamilnadu, India.
spriya1828@gmail.com

***Abstract*—The integrity of files stored on servers is crucial for many applications running on the internet. Recently, many works provide data dynamics and/or public verifiability to this type of protocols. Remote data possession checking protocols permit checking that, a remote server can access an uncorrupted file in such a way that a verifier does not need to know beforehand the entire file. In addition, both data dynamics operations and public verifiability are supported. Token pre-computation algorithm is used to generate tokens. Challenge-Response protocol is used for communication between the challenger and the responder. Through a formal analysis, the correctness and security of the protocol is shown. After that, through theoretical analysis and experimental results, we demonstrate that the proposed protocol has a good performance.**

***Index Terms*—Data integrity, data dynamics, public verifiability, privacy.**

I. INTRODUCTION

Cloud computing is the one, where users can remotely store their data into the cloud so as to enjoy the on-demand high quality applications and services from a shared pool of configurable computing resources.Storing data in the cloud has become a trend . An increasingnumber of clients store their important data in remote servers inthe cloud, without leaving a copy in their local computers.

Sometimes the data stored in the cloud is so important that theclients must ensure it is not lost or corrupted. While it is easy tocheck data integrity after completely downloading the data to bechecked, downloading large amounts of data just for checking dataintegrity is a waste of communication bandwidth. Hence, a lot ofworks have been done on designingremote data integrity checking protocols, which allow dataintegrity to be checked without completely downloading the data.Remote data integrity checking proposes RSA-based methods for solving thisproblem.

Recently, many works focus on providing three advanced features for remote dataintegrity checking protocols: data dynamics, publicverifiability , and privacy against verifiers.

Someprotocols support data dynamics at the blocklevel, including block insertion, block modification, and blockdeletion. Some protocol supports data append operation. On the other hand, some protocols support public verifiability, by which anyone (notjust the client) can perform the integrity checking operation. Some support privacy against third-partyverifiers.

In this paper, we have the following main contributions:

- We propose a remote data integrity checking protocol forcloud storage, which can be viewed as an adaptation ofSebe´ et al.'s protocol. The proposed protocol inherits thesupport of data dynamics, and supports publicverifiability and privacy against third-party verifiers, whileat the same time it doesn't need to use a third-party auditor.
- We give a security analysis of the proposed protocol,which shows that it is secure against the untrusted serverand private against third-party verifiers.
- We have theoretically analyzed and experimentally testedthe efficiency of the protocol. Both theoretical and experimentalresults demonstrate that our protocol is efficient.

II. PROBLEM DEFINITION

Consider a client who stores a file in a remote server which is located in some geographic location without even having a local copy in the system. Later the client may want to check whether the integrity of the file is maintained by the cloud or not. In order to avoid the disputes between the client and the server, the client asks for public verifiability. During the public verification, the third-party auditor

may leak any private information. So a privacy preserving remote data integrity checking protocol must be framed.

In added to that, the proposed protocol must support data dynamics including data insertion, data modification and data deletion. Also it must allow for public verifiability by which anyone can perform the data integrity checking operation. Here third-party auditor must be avoided because it leaks any private information. Whenever the data is modified, the corresponding tags must be updated. Data level dynamics is achieved using block level dynamics. The communication and computation cost must be reduced.

III. Challenges And Opportunities

We consider a cloud storage system in which there are a client andan un-trusted server.

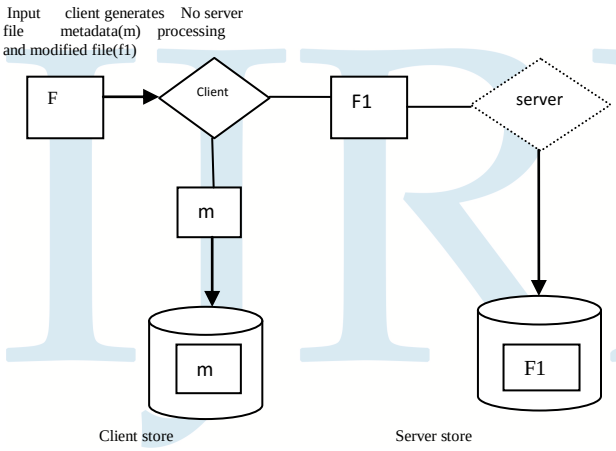

Fig. 1 Pre-process and Store

The client stores her data in the server without keeping a local copy. Hence, it is of critical importance that the client should be able to verify the integrity of the data stored in the remote untrusted server.

If the server modifies any part of the client's data, the client should be able to detect it; furthermore, any third-party verifier should also be able to detect it. In case a third-party verifier verifies the integrity of the client's data, the data should be kept private against the third-party verifier. Above we present a formal statement of the problem.

Denote by m the file that will be stored in the un-trusted server, which is divided into blocks of equal lengths.

We need to design a remote data integrity checking protocol that includes the following five functions: SetUp, TagGen, Challenge, GenProof, and CheckProof.

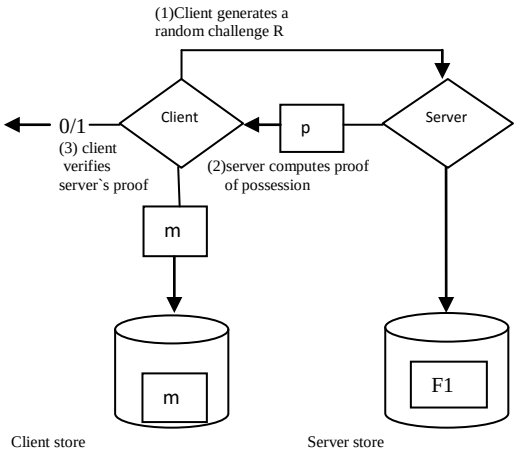

Fig. 2 Verify Server Possession

1) SetUp: Given a security parameter k, this function generates a public key which is known to everyone and a private key which is kept secret.

2) TagGen: Given the public key, private key and a file, this function computes a tag called verification tag and makes it publicly known to everyone. This tag will be used for public verification of data integrity.

3) Challenge: Using this function, the verifier generates a challenge to request for the integrity proof of m. The verifier sends request to the server.

4) GenProof: Using this function, the server computes a response to the request. The server sends response back to the verifier.

*5) CheckProof:*The verifier checksthe validity of the response. If it is valid, the function outputs "Success", otherwise the function outputs "Failure". The secret key is not needed in the *CheckProof* function.

A. Data Dynamics at Block Level

Data dynamics means after clients store their data at the remote server, they can dynamically update their data at later times. At the block level, the main operations are block insertion, block modification and block deletion. Moreover, when data is updated, the verification metadata also needs to be updated. The updating overhead should be made as small as possible.

B. Security Requirements

There are two security requirements forthe remote data integrity checking protocol: security against theserver with public verifiability, and

privacy against third-partyverifiers. We first give the definition of security against the serverwith public verifiability. In this definition, we have two entities: achallenger that stands for either the client or any third-partyverifier, and an adversary that stands for the untrusted server.

C. *Security against the Server with Public Verifiability*

When the verifier is not the client herself, the protocol mustensure that no private information about the client's data is leakedto the third-party verifier.

D. *Privacy against Third-Party verifiers*

For the remotedata integrity checking protocol, if there exists a PPT simulator S then the protocol ensures privacy against third-party verifiers.

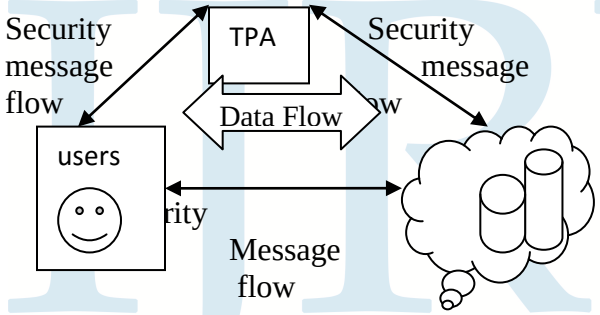

Fig. 3 Data storage architecture in cloud

E. *Homomorphic Verifiable Tags*

In our construction, we use aRSA-based homomorphic verifiable tags (HVT) . Whenever a piece of data is modified then the tag is updated. From this we can check the integrity of our file. Because of the homomorphic property, tags computed for multiple file blocks can be combined into a single value. The client pre-computes tags for each block of a file and then stores the file and its tags with a server. At a later time, the client can verify that the server possesses the file by generating a random challenge against a randomly selected set of file blocks. Using the queried blocks and their corresponding tags, the server generates a proof of possession. The client is thus convinced of data possession, without actually having to retrieve file blocks.The HVT has the following two features:

1) *Blockless verification:* By using HVTs, the server can construct a proof of possession of a certain file blocks, while the client needs not have access to these file blocks.

2) *Homomorphic property:*For any two messages *mi* and *mj* , the tag for *mi+mj* can be generated by combining *Di* and *Dj* .

IV. The Proposed Remote Data Integrity Checking Protocol

In this section, we describe the proposed remote data integritychecking protocol. we make all the blocksdistinct by adding random numbers in blocks with the same value.If the server still tries to save its storage space, then the only way isby breaking the prime factorization of N. The hardness of breaking large numberfactorization makes the proposed protocol secure against theun-trusted server.

V. Correctness And Security Analysis

In this section, we have to first show that the proposed protocol is correctin the sense that the server can pass theverification of data

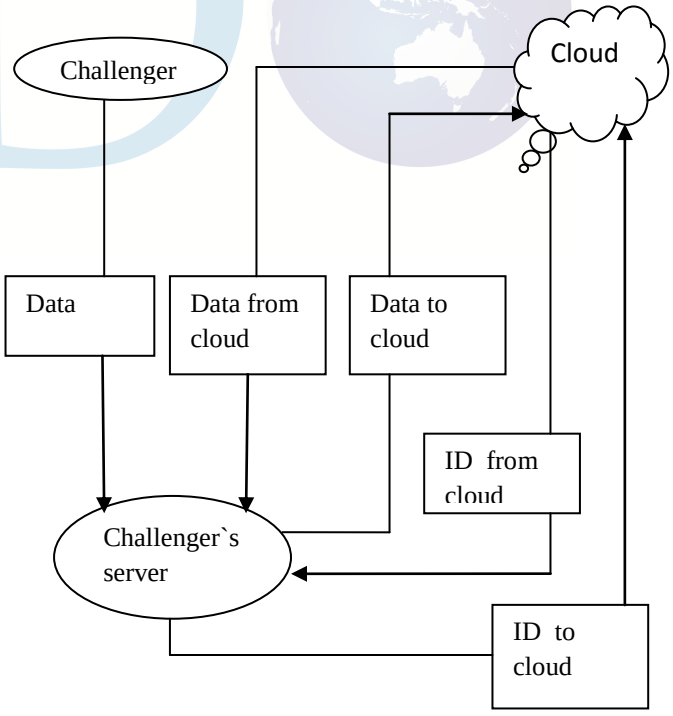

Fig. 4 Work flow of an E-RDPC protocol

integrity as long as both the client and the server are honest. Thenwe have to show that the protocol is secure against the un-trusted server.It guarantee that, assuming the client ishonest, if and only if the server has access to the complete anduncorrupted data, it

can pass the verification process successfully.Finally, we show that the proposed protocol is private againstthird-party verifiers.

- If both the client and the server are honest, then the servercan pass the verification successfully.
- Under the KEA1-r and the large integer factorizationassumptions, the proposed protocol is secure against the un-trustedserver.
- Under thesemi honest model, a third party verifier cannot get anyinformation about the client's data m from the protocol execution.Hence, the protocol is private against third-party verifiers.

VI. Data Dynamics

The proposed protocol supports data dynamics at the block level. If the client wants to make sure that thefile has really been updated, she can launch a proof requestimmediately by sending a challenge to the server. Any block that isupdated is given a novel random number, so that each blockremains unique. Therefore, the server cannot delete any blockwithout being detected.

VII. Experiments

In the experiment, we want to measure the computation costs at the verifierand the server when the file length is fixed and the block size is changed. After that, we want to measurethe computation costs when the file length changes and the blocksize is fixed. We also measure theclient's preprocessing costs.

VIII. Related Work

In this paper, we propose a new remote data integrity checkingprotocol for cloud storage. The proposed protocol is suitable forproviding integrity protection of customers' important data. Theproposed protocol supports data insertion, modification, anddeletion at the block level, and also supports public verifiability.

The proposed protocol is proved to be secure against an un-trustedserver. It is also private against third-party verifiers. Boththeoretical analysis and experimental results demonstrate thatthe proposed protocol has very good efficiency in the aspects ofcommunication, computation, and storage costs.

IX. Future Work

Work is to extend the protocol to support data level dynamics. The difficulty is that there is no clear mapping relationship between the data and the tags. In the current construction, data level dynamics can be supported by using block level dynamics. Whenever a piece of data is modified, the corresponding blocks and tags are updated. However, this can bring unnecessary computation and communication costs. We aim to achieve data level dynamics at minimal costs in our future work.

References

[1] F. Sebe, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte, and J.-J.Quisquater, "Efficient Remote Data Possession Checking in CriticalInformation Infrastructures," IEEE Trans. Knowledge and Data Eng.,vol. 20, no. 8, pp. 1034-1038, Aug. 2008.

[2] C. Wang, Q. Wang, K. Ren, and W. Lou, "Ensuring Data Storage Security inCloud Computing," Proc. 17th Int'l Workshop Quality of Service (IWQoS '09),pp. 1-9, July 2009.

[3] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving PublicAuditing for Data Storage Security in Cloud Computing," Proc. IEEEINFOCOM, Mar. 2010.

[4] C. Wang, S.S.-M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-PreservingPublic Auditing for Secure Cloud Storage," Cryptology ePrint Archive,Report 2009/579, http://eprint.iacr.org/, 2009.

[5] D.L.G. Filho and P.S.L.M. Barreto, "Demonstrating Data Possession andUncheatable Data Transfer." Cryptology ePrint Archive, Report2006/150,http://eprint.iacr.org/, 2006.

[6] Z. Hao, S. Zhong, and N. Yu, "A Privacy-Preserving Remote Data IntegrityChecking Protocol with Data Dynamics and Public Verifiability," TechnicalReport 2010-11, SUNY Buffalo CSE Dept., http://www.cse.buffalo.edu/