C

Cryptic Algorithms:
Hiding Sensitive Information in Cloud Computing

Shivlal Mewada
https://orcid.org/0000-0001-5543-8622
Government Holkar Science College, India

INTRODUCTION

An Overview

In today's digital world, there is an immense growth of data which is difficult for the users to store and share it locally. Due to this, a greater number of users switch to cloud storing facilities. But somehow, the data stored in the cloud might be manipulated or lost due to the unavoidable software bugs, hardware flaws and human inaccuracy in the cloud. In order to check whether the data is stored accurately in the cloud storage, many remote data integrity auditing schemes have been implemented.

The data stored in the cloud storages is frequently shared across multiple users in other cloud storage applications, such as Google Drive, Dropbox and iCloud. Data sharing in cloud storage allows a number of users to share their data with others. However, shared data in the cloud platforms might contain some tactful or sensitive information. For example, the Health Records, Bank Transactions, Confidential information etc. stored and shared in the cloud usually contains tactful or sensitive information. If these documents are directly uploaded to the cloud to be shared for research purposes, the sensitive information of data owner will be inevitably released to the cloud and the researchers. Thus, it is important to achieve remote data integrity auditing on the terms that the sensitive information of shared data is protected. A typical method of solving this problem is to encrypt the whole shared file before sending it to the cloud storages, and then verify the integrity of this encrypted file. This method provides the sensitive information hiding since only the data owner can decrypt this file. But, the whole shared file will be will not be available others. For example, encrypting the Health Records of infectious disease to patients can protect the privacy of patient and hospital, but this encrypted Health Record cannot be efficiently used by researchers for further findings any more. Distributing the decryption method to the researchers seems to be a possible solution to the above problem. However, it is impractical to accept this method in real scenarios due to the following reasons.

Firstly, distributing decryption method needs secure sources of communication, which is difficult to be implemented for each researcher in some instances. Further, it seems very hard for a user to know which researchers will use his/her Health Records in the near future when he/she uploads the Health Records to the cloud. As a conclusion, it is impractical to hide sensitive information by encrypting the whole shared file. Thus, how to execute data storing and sharing simultaneously with sensitive information hiding in remote data integrity auditing is very dominant and valuable.

DOI: 10.4018/978-1-7998-9220-5.ch044

Related Work

In process to check the integrity of the data blocks stored in the cloud storage, many remote data integrity auditing schemes have been proposed. To reduce the computation burden on user, a Third-Party Auditor (TPA) is introduced to ensure the integrity of the data stored in cloud for user. (K. Ren, 2012, G. Ateniese, 2007) here author, proposed a notation of Provable Data Possession (PDP) to ensure the data possession on the unreliable cloud storage. In their scheme, homomorphic authenticators and random sampling strategies are used to achieve block less verification and reduce hardware costs. In order to protect the data privacy, Wang et al. (C. Wang, 2013) proposed a privacy-preserving remote data integrity auditing scheme with the employment of a random masking technique. Solomon et al. (S. G. Worku, 2014) utilized a different random masking technique to further construct a remote data integrity auditing scheme supporting data privacy protection. Wang et al. (Y. Zhang, 2011) proposed another remote data integrity auditing scheme supporting full data dynamics by utilizing Merkle Hash Tree. In (B. Wang, 2012) authors designed a privacy-preserving shared data integrity auditing scheme by modifying the ring signature for secure cloud storage. With the employment of the Shamir secret sharing technique, Luo et al. (Y. Luo, 2015) constructed a shared data integrity auditing scheme supporting user revocation. The aforementioned schemes all rely on Public Key Infrastructure (PKI), which incurs the considerable overheads from the complicated certificate management. To simplify certificate management, Yu et al. (H. Wang, 2015) constructed a remote data integrity auditing scheme with perfect data privacy preserving in identity-based cryptosystems. Wang et al. (H. Wang, 2016) proposed an identity-based data integrity auditing scheme satisfying unconditional anonymity and incentive. Zhang et al. (Y. Zhang, 2018) proposed an identity-based remote data integrity auditing scheme for shared data supporting real efficient user revocation. Other aspects, such as privacy-preserving authenticators (W. Shen, 2017) and data deduplication (J. Li, 2016, S. Gonth, 2020, S. Mewada, 2013, Vivek R., 2013, Mewada, 2011, 2015, 2016, 2020, 2021) in remote data integrity auditing have also been explored. However, all of existing remote data integrity auditing schemes cannot support data sharing with sensitive information hiding. In this paper, we explore how to achieve data sharing with sensitive information hiding in identity-based integrity auditing for secure cloud Storage.

SYSTEM METHODOLOGY

A) Process

STEP 1: Firstly, data owner generates a private key using his/her unique contact number.

STEP 2: Then the owner himself blinds the original file i.e., hides sensitive/personal information and passes the blinded file to the sanitizer module in the system.

STEP 3: Sanitizer module sanitizes the blinded file i.e., per-forms the 2nd step of hiding sensitive/personal information of the data owner. After the sanitization process, sanitized file is stored into the cloud storage.

STEP 4: At any point if the data owner needs a confirmation that the file stored in the cloud is completely sanitized then he delegates a Third-Party Auditor (TPA) Module to check the file stored in cloud.

STEP 5: Third Party Auditor (TPA) Module sends a auditing challenge to the cloud and in return cloud system responds with an auditing proof.