# Data Sharing with Sensitive Information Hiding for Secure Cloud Storage

**Vipul Bornare[1] Kiran Nikam[2] Dhiraj Khedkar[3] Sagar Hole[4]**
[1,2,3,4]Department of Information Technology
[1,2,3,4]Sanjivani College of Engineering, Kopargaon, India

*Abstract*— On the cloud storage services like Google drive users can store their data on the cloud remotely and also do data sharing with others. Auditing of data integrity is made to guarantee the integrity of the data stored in the cloud is fulfilled properly. In the cloud storage systems, the cloud file might contain some confidential information. The secret information should not be viewed to others when the cloud is shared. Encryption of the whole file which is shared can tell that the sensitive information is hidden, but will be resulted in that file is not exposed to others. How can we come to know that data sharing with sensitive information hiding is been explain in this research paper? Now here we are addressing this problem, by providing a solution for sharing of data with sensitive information kept hidden on remotely stored data at cloud while keeping it integrated in this project.

***Keywords:*** Cloud Storage, Data Integrity Auditing, Encryption, Sensitive Information Hiding, Sanitization, Key Generation

## I. INTRODUCTION

Cloud computing is an attracting technology in the field of computer science. It is proven that cloud will bring changes to the IT industry. The cloud is changing our life by providing users with new types of services. Users get service from a cloud without paying attention to the details. Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. More and more people pay attention to cloud computing. Cloud computing is efficient and scalable but maintaining the stability of processing so many jobs in the cloud computing environment is a very complex problem with load balancing receiving much attention for researchers. In these the following four architectural patterns are distinguished:

1) Replication of applications allows to receive multiple results from one operation performed in distinct clouds and to compare them within the own premise. This enables the user to get an evidence on the integrity of the result.
2) Partition of application System into tiers allows separating the logic from the data. This gives additional protection against data leakage due to flaws in the application logic.
3) Partition of application logic into fragments allows distributing the application logic to distinct clouds. This has two benefits. First, no cloud provider learns the complete application logic. Second, no cloud provider learns the overall calculated result of the application. Thus, this leads to data and application confidentiality.
4) Partition of application data into fragments allows distributing fine-grained fragments of the data to distinct clouds. None of the involved cloud providers gains access to all the data, which safeguards the date's confidentiality.

## II. LITERATURE SURVEY

1) Data Sharing with Sensitive Information Hiding for Secure Cloud Storage.
   Wenting Shen, Jing Qin, Jia Yu, Rong Hao, and Jiankun Hu, Enabling Identity-Based Integrity Auditing and Data Sharing with Sensitive Information Hiding for Secure Cloud Storage IEEE Transactions on Information Forensics and Security.vol. 14, no. 2, February. 2019.
   The sensitive information accessible to third party can become threat and is against privacy of individual. With sensitive information hiding we can protect data.
2) Key-Exposure for Secure Cloud Storage.
   J. Yu, K. Ren, C. Wang, and V. Varadharajan, Enabling Cloud Storage Auditing with Key-Exposure Resistance, IEEE Transactions on Information Forensics and Security. vol. 10, no. 6, pp. 1167-1179, Jun. 2015.
   Key exposure cannot affect the security of authenticators generated before the key-exposure time period. The security of authenticators generated later than the key-exposure time period is still unable to preserve
3) Security challenges for the public cloud.
   K. Ren, C. Wang, and Q. Wang, IEEE Internet Comput., vol. 16, no. 1, pp. 6973, Jan. 2012.
   Cloud computing represents today's most exciting computing paradigm shift in information technology. However, security and privacy are perceived as primary obstacles to its wide adoption. Here, the author's outline several critical security challenges and motivate further investigation of security solutions for a trustworthy public cloud environment.
4) Secure and efficient privacy preserving public auditing scheme for cloud storage.
   S. G. Worku, C. Xu, J. Zhao, and X. He, Comput. Electr. Engg., vol. 40, no. 5, pp. 17031713, 2018.
   A secure cloud storage system supporting privacy-preserving public auditing. We further extend our result to enable the TPA (third party authority) to perform audits for multiple users simultaneously and efficiently. Extensive security and performance analysis show the proposed schemes are provably secure and highly efficient.
5) Light-weight and privacy-preserving secure cloud auditing scheme for group users via the third-party medium.

 

W. Shen, J. Yu, H. Xia, H. Zhang, X. Lu, and R. Hao, J. Netw. Comput. Appl., vol. 82, pp. 5664, Mar. 2017.

A Third-Party Medium (TPM) to perform time-consuming operations on behalf of users. The TPM is in charge of generating authenticators for users and verifying data integrity on behalf of users. In order to protect the data privacy against the TPM, we blind data using simple operations in the phase of data uploading and data auditing.

## III. PROPOSED SYSTEM

In system provides a safe cloud storage methodology which supports privacy-preserving third party auditing better than existing system. This suggests that the security can be increased if the architecture is changed from single cloud to multiple cloud environment. Security mechanisms involved during third party auditing of outsourced data is discussed. The methods are studied to perform the auditing without demanding the local copy of data and thus drastically reduce the communication and computation overhead. Four schemes are presented that can be applied in multiple cloud environment to increase the security aspects. Hiding resource usage statistics of a single resource for a single cloud provider is achieved if first method is applied. The computation and data transfer size is very low if the second method is applied. The third method provides the security such that a single provider may not be aware of the execution flow of the single application as well as the cloud provider could not know or access all the data. The fourth method provides the benefit of auditing with very low credential data to verify the file content. It is proved that the third party auditing computation time is better than existing approach.

1) Users Registration- When first time users visit the website that time they have to register and fill necessary details to create an account.
2) Login - Once account is created then users can login into their respective accounts with the help of username and password.
3) Key Generator login- User have to login with the help of username and password for key generation purpose. Once user make successfull login then they have to enter name, enter user private key and upload a text file from local storage and click on submit button.
4) Cloud- The cloud gives us enormous data storage space to the user. Through the cloud storage service, users can upload their data to the cloud and share their data with others.
5) Sanitizer- The sanitization is the process of ensuring that only the intended information can be accessed from a document. The sanitizer checks the validity of the file tag with the help of by verifying whether signature is a valid signature or not.
6) Signature- The signatures are used to guarantee the authenticity of the files and verify the integrity of the files.
7) Encryption- Once the file sanitizer checks the document then document is available for encryption. In encryption we have used fernet algorithm (symmetric encryption) to encrypt sensitive information so that information is not exposed to cloud, sanitizer and other unwanted users.
8) User Audit Request− User will send auditing request to TPA (Third Party Auditor) to check data stored on cloud is intact or not. The job of TPA is to check data integrity on behalf of the user.
9) TPA Request to Cloud- When the TPA needs to verify the integrity of the sanitized file stored on the cloud, they sends an auditing challenge to the cloud to see whether file is kept integrated or not.
10) Auditing Proof- The cloud responds to the TPA with an auditing proof of data possession. Here cloud send proof of the data integrity to the TPA. Finally, the TPA verifies the integrity of the sanitized file by checking whether this auditing proof is correct or not.
11) Download File- Now users can download the file from the cloud which they had stored on cloud storage. This is the complete process of data sharing on cloud while user can keep its sensitive information hidden from other third parties.

## IV. SYSTEM ARCHITECTURE

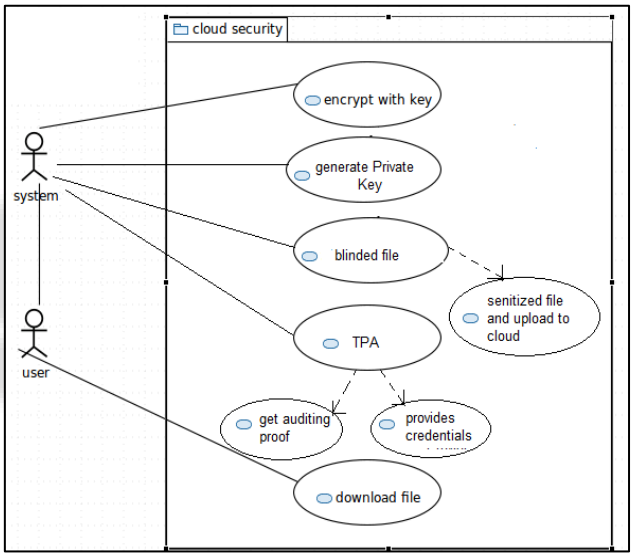


Fig. System Architecture

## IV. ALGORITHM

### *A. Algorithms*

#### *1) Fernet (symmetric encryption)*

1. Fernet guarantees that a message encrypted using it cannot be manipulated or read without the key.
2. Fernet is an implementation of symmetric (also known as “secret key”) authenticated cryptography.
3. Fernet also has support for implementing key rotation via MultiFernet.

```
>>> key = Fernet.generate_key()
>>> f = Fernet(key)
>>> token = f.encrypt(b"my deep dark secret")
>>> token b'...'
>>> f.decrypt(token) b'my deep dark secret'
```

 

*B. Parameters:*

*1) key(bytes)*

A URL-safe base64-encoded 32-byte key. This must be kept secret. Anyone with this key is able to create and read messages.

*2) classmethodgenerate_key()*

Generates a fresh fernet key. Keep this some place safe! If you lose it youll no longer be able to decrypt messages; if anyone else gains access to it, they'll be able to decrypt all of your messages, and they'll also be able forge arbitrary messages that will be authenticated and decrypted.

*3) encrypt(data)*

Encrypts data passed. The result of this encryption is known as a "Fernet token" and has strong privacy and authenticity guarantees.

a) Parameters:

data (bytes) — The message you would like to encrypt.

b) Returns bytes:

A secure message that cannot be read or altered without the key. It is URL-safe base64-encoded. This is referred to as a "Fernet token".

c) Raises:

TypeError — This exception is raised if data is not bytes.

*4) decrypt(token, ttl=None)*

Decrypts a Fernet token. If successfully decrypted you will receive the original plaintext as the result, otherwise an exception will be raised. It is safe to use this data immediately as Fernet verifies that the data has not been tampered with prior to returning it.

a) Parameters:

token (bytes) — The Fernet token. This is the result of calling encrypt().

ttl (int) — Optionally, the number of seconds old a message may be for it to be valid. If the message is older than ttl seconds (from the time it was originally created) an exception will be raised. If ttl is not provided (or is None), the age of the message is not considered.

b) Returns bytes:

The original plaintext.

c) Raises:

cryptography.fernet.InvalidToken — If the token is in any way invalid, this exception is raised. A token may be invalid for a number of reasons: it is older than the ttl, it is malformed, or it does not have a valid signature.

TypeError — This exception is raised if token is not bytes.

## ACKNOWLEDGMENT

It is with the greatest pleasure and pride that we present this report. At this moment of triumph, it would be neglect all those who helped us in the successful completion of this project. We are very much thankful to our respected project guide Prof. C. D. Bawankar and Hon. Dr. M. A. Jawale (HOD IT) for there ideas and help proved to be valuable and helpful during creation of project report and set us in the right path.

We would also like to thank all the faculties who have cleared all the major concepts that were involved in the understanding techniques behind our project. Lastly, we are thankful to our friends who shared their knowledge in this field with us.

## V. CONCLUSION

We proposed an identity-based data integrity auditing scheme for secure cloud storage, which supports data sharing with sensitive information hiding in our scheme, the file stored in the cloud can be shared and used by others on the condition that the sensitive information of the file is protected.

## REFERENCES

[1] Wenting Shen, Jing Qin, Jia Yu, Rong Hao, and Jiankun Hu, Enabling Identity-Based Integrity Auditing and Data Sharing With Sensitive Information Hiding for Secure Cloud Storage IEEE Transactions on Information Forensics and Security.vol. 14, no. 2, February. 2019.

[2] J. Yu, K. Ren, C. Wang, and V. Varadharajan, Enabling Cloud Storage Auditing with Key-Exposure Resistance, IEEE Transactions on Information Forensics and Security.vol. 10, no. 6, pp. 1167-1179, Jun. 2015.

[3] K. Ren, C. Wang, and Q. Wang, Security challenges for the public cloud, IEEE Internet Comput., vol. 16, no. 1, pp. 6973, Jan. 2012.

[4] W. Shen, G. Yang, J. Yu, H. Zhang, F. Kong, and R. Hao, Remote data possession checking with privacy-preserving authenticators for cloud storage, Future Gener. Comput. Syst., vol. 76, pp. 136145, Nov. 2017.

[5] H. Shachamand B. Waters, Compact proofs of retrievability, J. Cryptol., vol. 26, no. 3, pp. 442483, Jul. 2013.

[6] C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, Privacypreservingpublic auditing for secure cloud storage, IEEE Trans. Comput., vol. 62, no. 2, pp. 362375, Feb. 2013.

[7] S. G. Worku, C. Xu, J. Zhao, and X. He, Secure and efficient privacypreservingpublic auditing scheme for cloud storage, Comput. Electr. Eng., vol. 40, no. 5, pp. 17031713, 2014.

[8] C. Guan, K. Ren, F. Zhang, F. Kerschbaum, and J. Yu, Symmetric key based proofs of retrievability supporting public verification, in Computer Security ESORICS. Cham, Switzerland: Springer, 2015, pp. 203223.

[9] W. Shen, J. Yu, H. Xia, H. Zhang, X. Lu, and R. Hao, Light-weight and privacy preserving secure cloud auditing scheme for group users via the third-party medium, J. Netw. Comput. Appl., vol. 82, pp. 5664, Mar. 2017.

 