WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Implementing Client-Side Encryption for Cloud Data Security

Vishal G K[1], Priyadharshini V[2]

[1]UG Student, Department of Computer Science, Sri Krishna Adithya College of Arts and Science, Coimbatore.
[2]Assistant Professor, Department of Computer Science, Sri Krishna Adithya College of Arts and Science, Coimbatore.

ABSTRACT:

This project is dedicated to building a robust, secure file storage and retrieval mechanism that prioritizes data privacy through end-to-end encryption. The system enables users to upload files securely, with each file being encrypted before it is sent to cloud storage providers such as Amazon Web Services (AWS) or Google Cloud Storage (GCS). In this setup, file encryption occurs on the user's side to ensure that sensitive data is protected both during transit and while at rest in the cloud. Using the Advanced Encryption Standard (AES) algorithm, each file is converted into an encrypted form that can only be decrypted by providing the correct password, ensuring that unauthorized access is prevented. When users need to retrieve their files, they enter the decryption password, which is validated by the backend system to securely decrypt and serve the file. The front end is developed using HTML and CSS, offering a user-friendly interface where users can seamlessly upload and access their encrypted files. The interface is designed for simplicity and efficiency, making it easy for users to interact with the system without any technical expertise. The backend is powered by Python, which handles the critical functions of interacting with cloud storage services, managing encryption and decryption processes, and verifying decryption passwords. Python libraries such as `boto3` for AWS and `Google-cloud-storage` for GCS provide the functionality to securely upload and retrieve files, while cryptographic libraries like `pycryptodome` or `cryptography` ensure strong data encryption and decryption. Through this end-to-end encrypted storage solution, the project offers users a secure, efficient, and user-centric system for handling sensitive files on cloud platforms, mitigating the risks associated with cloud storage and ensuring that files remain protected from unauthorized access at all times.

Keywords: AES, Encryption, Decryption, Cloud Storage.

2. Introduction:

The increasing reliance on cloud-based storage solutions for data management has necessitated robust security measures to protect sensitive files. While traditional cloud storage providers offer basic encryption, they often retain control over encryption keys and the encryption/decryption processes, potentially compromising data security.

Client-side encryption (CSE) offers a more secure approach by encrypting data locally before it is uploaded to the cloud. This ensures that the data remains encrypted throughout its lifecycle, even while residing on the cloud provider's servers. The user's personal password serves as the decryption key, granting exclusive access to the encrypted files.

By employing CSE, organizations and individuals can significantly enhance the security of their cloud-stored data. It empowers users to maintain complete control over their sensitive information, mitigating the risks associated with potential data breaches or unauthorized access. This proactive security measure provides peace of mind, knowing that their valuable data is protected from prying eyes and malicious actors.

CSE offers several key advantages:

- Enhanced Security: Data is encrypted before being sent to the cloud, making it inaccessible to unauthorized individuals, even if the cloud provider's systems are compromised.
- User Control: Users retain full control over their encryption keys, ensuring that only they can decrypt and access their data.
- Compliance: CSE can help organizations comply with data privacy regulations by providing an additional layer of security and control over sensitive information.
- Flexibility: CSE can be implemented with various cloud storage providers, offering flexibility in choosing the most suitable solution

While CSE offers significant security benefits, it's essential to consider the following:

- . Performance Overhead: Encryption and decryption processes can add overhead to data transfer and access times.
- Complexity: Implementing CSE requires careful consideration and technical expertise to ensure proper configuration and security.

By understanding the benefits and limitations of CSE, organizations and individuals can make informed decisions about how to protect their sensitive data in the cloud.

What is an Encryption?

Server-side encryption is the most commonly used type of encryption, while client-sided encryption does not. In server-side encryption, the provider will handle both the user data and the encryption key. These methods simplify the user or data owner's tasks. However, as the frequency of attacks from insider's increases, confidence in data administrators is on the decline. Scientists suggest the implementation of client-side encryption. Client side encryption, as the name suggests, enables users to control their own key and encrypt their data prior to saving it in the database. The security of data is ensured by determining who has the key and the authorization to access and alter the data. The database administrator, for instance, has limited access to view the data. They are only able to see a small portion of the information related to the stored data. The research primarily focuses on client-side encryption, which is carried out through the implementation of AES. We are eager to delve deeper into the intricacies of how changes occur during the process of storing and retrieving data in an encrypted database. We aim to explore the user's role and abilities, as well as the pivotal function of the proxy as the central component in AES. The fact that this type of encryption system is a valuable approach for data security means that additional research and implementation is required in real-life scenarios.

What is the use of Cloud Storage?

Cloud storage presents multiple standout features:- Immediate accessibility, affordability, and ease- Reliability, flexibility, and a diverse array of leasing options Moreover, cloud computing relies on key features such as security, scalability, cost-effectiveness, accessibility, data recovery capabilities, and efficient resource utilization. Trust becomes a critical issue when considering the transfer of user data to cloud environments, posing a substantial challenge in the interaction between users and cloud service providers. Users of cloud storage services require clear visibility and assurance concerning the security and integrity of their data stored in the cloud. This is particularly crucial due to the lack of comprehensive monitoring capabilities. To cater to this essential need and encourage widespread user approval, various data and resource protection approaches have been implemented and incorporated into the realm of cloud security, utilizing modern cryptographic algorithms.

3. Methodology:

3.1.1 Existing Methodologies

In the realm of cloud storage, several established encryption methodologies are commonly employed to safeguard data. These methodologies primarily include server-side encryption provided by cloud storage services, as well as various client-side encryption tools.

AWS S3 Encryption

Amazon Web Services (AWS) Simple Storage Service (S3) provides robust server-side encryption options for securing files at rest. With AWS S3, users can choose from multiple encryption options, including server-side encryption with Amazon S3 managed keys (SSE-S3), server-side encryption with AWS Key Management Service keys (SSE-KMS), and server-side encryption with customer-provided keys (SSE-C). These options allow flexibility for users, as they can select either AWS-managed keys or manage their own encryption keys to maintain tighter control over data security.

Google Cloud Storage (GCS) Encryption

Google Cloud Storage offers comparable encryption services to AWS, enabling users to encrypt files stored within the cloud. Users can select from three encryption key options: Google-managed keys, customer-managed keys through Google Cloud's Key Management Service (KMS), or customer-supplied keys. By allowing users to manage encryption keys, Google Cloud provides an added layer of control over data security, making it possible for organizations to align their cloud storage practices with internal security policies.

Client-Side Encryption Tools

Beyond built-in encryption options from cloud providers, client-side encryption tools, such as GnuPG and VeraCrypt, offer additional methods for securing data. These tools allow users to encrypt files on their devices before uploading them to the cloud, ensuring that the files remain protected even if the cloud service itself is compromised. However, client-side encryption tools generally require manual handling and configuration, making them less integrated with cloud storage solutions and often more challenging for non-technical users to operate.

3.1.2 Drawbacks of Existing Methodologies

While the existing methodologies provide fundamental layers of security, they also exhibit certain limitations that affect data confidentiality, integrity, and user experience.

Dependence on Cloud Provider Security

Data security in cloud storage is significantly influenced by the security policies and infrastructure of the cloud provider. Cloud service providers, despite rigorous security measures, are vulnerable to threats such as data breaches, insider threats, and denial-of-service (DoS) attacks. Users relying on server-side encryption are ultimately dependent on the cloud provider‟s security framework, which could potentially expose them to risks beyond their control

Vulnerability to Phishing Attacks

Even with encrypted cloud storage, users remain susceptible to phishing attacks that seek to compromise their credentials. Attackers may deploy phishing tactics to steal user credentials, potentially decrypting stored files and accessing sensitive information. This highlights a potential vulnerability

where users could inadvertently compromise their own data security by responding to deceptive requests or providing sensitive information to unauthorized parties.

Complexity and Lack of User-Friendliness

The technical complexity associated with encryption and decryption can deter non-technical users from adopting secure storage practices. Existing encryption methodologies may require specialized knowledge or multiple steps to implement, presenting a usability barrier for some users. A user-friendly interface, as well as simplified encryption workflows, is essential for widespread adoption, ensuring users of all technical backgrounds can securely store and access their data.

3.2.1 Proposed Methodologies

The proposed methodology aims to address the limitations of existing approaches by integrating encryption and decryption directly into the file upload and download workflow within the cloud. This approach provides an additional layer of security and enhances user control over encryption practices.

Integrated Client-Side Encryption Process

Under the proposed methodology, files are encrypted on the client side before they are uploaded to the cloud, incorporating password protection as an added measure of security. This ensures that files remain encrypted throughout their storage period and can only be decrypted when downloaded and provided with the correct password. Users maintain exclusive control over the encryption keys, thus preserving data confidentiality even if cloud provider security is compromised.

Controlled Decryption Process

To access the encrypted files, users must supply the correct password during download, enabling decryption only upon successful authentication. This user-centric approach to decryption provides the highest level of security, as files remain encrypted in storage and are inaccessible without the appropriate credentials. By embedding encryption and decryption into the upload and download process, the proposed methodology ensures that users have complete control over the security of their data.

3.2.2 Benefits of Proposed Methodologies

The proposed methodology offers significant advantages over traditional encryption approaches by enhancing security, control, and ease of use.

Enhanced Security through Client-Side Encryption

Client-side encryption mitigates reliance on cloud provider security by ensuring that files are encrypted before they are uploaded. This reduces the potential for unauthorized access, as data remains encrypted throughout its lifecycle in the cloud.

Complete Control over Encryption Keys

Users retain full control over the encryption keys, empowering them to manage their data independently of cloud provider security policies. This user-driven control over encryption keys minimizes the risk of data breaches associated with compromised cloud providers or third-party attacks.

Simplified and User-Friendly Workflow

The integration of encryption and decryption directly into the upload and download workflow simplifies the process of securely managing files in the cloud. A streamlined workflow makes it accessible to users of all technical levels, fostering greater adoption of secure storage practices.

Reduced Risk of Data Breaches

By encrypting files on the client side, the proposed methodology reduces the risk of data breaches stemming from vulnerabilities in cloud provider infrastructure. As a result, users can confidently store sensitive data in the cloud without exposing it to additional security risks associated with third-party management of encryption

AES (Advanced Encryption Standard)

In 2000, the NIST intentionally selected Rijndael as the advanced encryption standard due to its outstanding qualities in terms of security, performance, and elegance. As per NIST guidelines, the symmetric encryption method AES has a block size of 128 bits. A key feature is that AES can vary the number of encryption rounds according to the size of the encryption key. More specifically, For a 128-bit key, the AES uses 10 rounds of encryption; for 192-bit and 256-bit keys, it uses 12 rounds and 14 rounds, respectively [15]. The fundamental building blocks of each encryption round in AES encompass SubBytes, ShiftRows, MixColumns, and AddRoundKey operations. Within these functions, the AddRoundKey operation stands out, as it performs a crucial exclusive OR (XOR) operation between the input state matrix and the cryptographic key. It is noteworthy that in the traditional AES framework, each round key is generated by means of a predetermined key expansion process. The choice of Rijndael as the advanced encryption standard, together with its block length, variable number of encryption rounds, and essential components of AES rounds, such as the crucial AddRoundKey operation, all work together to enhance the strength and efficiency of this highly utilized encryption algorithm.

Objective:

1. **To develop a secure cloud storage solution** that integrates client-side encryption and decryption, ensuring data remains encrypted both during upload and while stored.

2. **To provide users with full control over encryption keys** and the decryption process, minimizing reliance on cloud provider security policies.
3. **To create a streamlined, user-friendly interface** that simplifies secure file uploads and downloads,.
4. **To enhance data security against potential breaches** by implementing an independent encryption system that reduces risks associated with compromised cloud providers.

4. Results :

The process of generating blocks for the purpose of adding them to the encryption comprises the following stages. It commences with the retrieval of the dynamic encryption key used to encrypt the file, which is generated as part of the key generation process. Simultaneously, the contents of the latest block in the encryption are read. Subsequently, the block counter is incremented by one, and the current time and date on the client's device are recorded. The next step involves constructing the block, which will contain the encryption key as data and the hash of the previous block. Following this, the user encrypts the entire block using their private encryption key, and the new block, along with the encrypted file, is transmitted to the server. Additionally a local copy of the block may be retained for the purpose of expedited retrieval when necessary. Every client is required to possess a pair of AES keys, comprising both public and private keys. The public key serves the dual purpose of being disseminated to the general public and stored alongside the client's unique identifier on the server when sharing with other clients is necessitated. Conversely, the private key is retained by the client to facilitate the decryption of blocks and to obtain the access key for each individual file exclusively. The client's public key plays a pivotal role in encrypting block data on the client side prior to transmission to the server, thereby rendering the block data impervious to inspection by the server's administrators.

Decryption and File Downloading

In the process of decryption and file retrieval, the client initiates a request for the file they wish to decrypt. Consequently, the server transmits the file along with its associated block. Using the private key of the AES algorithm, the client decrypts the block to access its specific data, which includes the encryption key specific to the file. The key not being present in a specific format on the server allows the service provider to generate numerous copies of the user's encryption and distribute them in a decentralized way. Only individuals who are authorized and possess the AES decryption key can access the necessary keys for a specific file. Moreover, users will be able to effortlessly handle numerous files with unique keys by utilizing just one key securely stored on their device.

Number of character	CBS [21] algorithm (execution time)	Proposed algorithm (execution time)
1000	1135ms	930ms
2000	2120ms	1780ms
3000	3218ms	2800ms

1. User Download Request

The process begins when the user initiates a request to download a previously uploaded file. This request is typically made through the user interface (UI) on the front end, where the user selects the desired file for retrieval. The system receives this request and prepares to fetch the encrypted file from cloud storage.

2. Fetching the Encrypted File from Cloud Storage

Upon receiving the download request, the system communicates with the cloud storage service (e.g., AWS S3 or Google Cloud Storage) to retrieve the encrypted file. The file remains in its encrypted form during this process, ensuring that it is protected against unauthorized access while in transit. The encrypted data is securely transferred back to the backend server, ready for decryption.

3. User Authentication and Password Entry

To proceed with decryption, the system prompts the user to enter the decryption password. This password is essential, as it serves as the encryption key that was used to initially secure the file before it was uploaded. The system verifies that the entered password matches the required decryption key. This verification step is crucial to prevent unauthorized access to the data. If the password is incorrect, access to the decrypted content is denied, maintaining the integrity and security of the system.

4. Decryption of the Encrypted File

Once the password is verified, the system uses it to decrypt the file using the Advanced Encryption Standard (AES) algorithm. This algorithm transforms the encrypted data back into its original, readable form. The decryption process takes place on the client side, ensuring that sensitive data remains private and is not exposed to the backend or any external systems. This client-side decryption adds an extra layer of security by preventing unauthorized decryption at the server level.

5. Providing Access to the Decrypted File

After successful decryption, the file is made accessible to the user. The decrypted content is now available in its original format, and the user can view, download, or otherwise interact with it as needed. This final step completes the secure file download process, ensuring that data privacy and integrity have been maintained from storage through to user access.

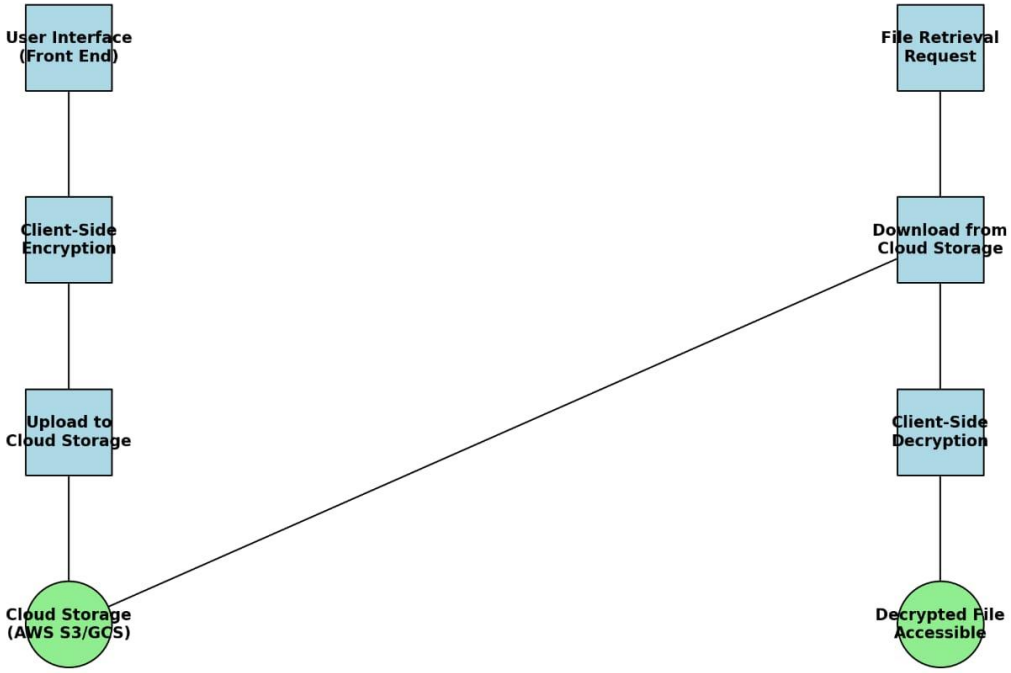

Fig 1 Block Diagram

5. Conclusion :

This project emphasizes the critical need for robust encryption mechanisms within cloud-based storage systems to address escalating concerns about data security in today's digital landscape. By implementing end-to-end encryption for files stored in the cloud, the project presents a viable solution for safeguarding sensitive data, ensuring that it remains protected both in transit and at rest. Through this approach, users retain control over encryption keys and decryption processes, significantly reducing the risk of unauthorized access and exposure to potential cyber threats. The solution is designed to serve the security needs of both individual users and organizations, offering a scalable, user-friendly system that balances accessibility with rigorous security measures. For individuals, it provides a straightforward way to store personal data securely, while for organizations, it establishes a framework that can be integrated into existing data protection strategies. By encrypting files on the client side, this project mitigates dependencies on cloud providers' internal security measures, thereby enhancing resilience against vulnerabilities associated with third-party services.

Moreover, this project demonstrates the use of the Advanced Encryption Standard (AES), a widely recognized encryption algorithm that combines strong cryptographic practices with practical implementation capabilities. This choice underscores the project's commitment to employing industry-standard solutions that align with best practices in information security, further strengthening the integrity and trustworthiness of the proposed system. Looking ahead, the methodologies and encryption practices implemented here lay a foundation for developing more advanced, multi-layered secure storage systems. Future work can extend these principles to accommodate additional security features, such as multi-factor authentication, automated key rotation, and real-time threat monitoring. These advancements would further reinforce data security within cloud environments, helping to counter the evolving threats in cyber security.

In conclusion, this project provides a comprehensive and practical solution for secure cloud storage, setting a standard for data privacy in cloud computing. It serves as a meaningful step toward more secure digital storage solutions, empowering users to manage and protect their data confidently in an increasingly interconnected and data-driven world.

6. REFERENCES:

1. Khaleel, Mustafa Ibrahim. "Dynamic AES Encryption and Blockchain Key Management: A Novel Solution for Cloud Data Security." qa.univsul.edu.iq. IEEEAccess, 8 Jan. 2024. Web. 10 Nov. 2024. A. Lewko and B. Waters, "Decentralizing Attribute-Based Encryption," in Proceedings of the 29th Annual International Conference on the Theory and the Applications of Cryptographic Techniques – EUROCRYPT 2010.
2. R. Anandkumar, K. Dinesh, A. J. Obaid, P. Malik, R. Sharma, A. Dumka, R. Singh, and S. Khatak, ''Securing e-health application of cloud computing using hyperchaotic image encryption framework,'' Comput. Electr. Eng., vol. 100, May 2022, Art. no. 107860.

3. Y. Alemami, A. M. Al-Ghonmein, K. G. Al-Moghrabi, and M. A. Mohamed, ‘‘Cloud data security and various cryptographic algorithms,’’ Int. J. Electr. Comput. Eng. (IJECE), vol. 13, no. 2, p. 1867, Apr. 2023
4. Sahai, B. Waters, and B. F. Wu, "Fully Secure Functional Encryption: AttributeBased Encryption and (Hierarchical) Inner Product Encryption," in Proceedings of the 2014 ACM Conference on Computer and the Communications Security – CCS 2014.
5. F. Guo, Y. Mu, and Z. Chen, “IdentityBased Encryption: How to Decrypt Multiple Ciphertexts Using a Single Decryption Key,” Proc.Pairing Based Cryptography Conf. (Pairing ’07), vol. 4575, pp. 392-406, 2007.
6. V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-Based on Encryption for Fine-Grained Access Control of Encrypted data,” in Proceedings the 13th ACM Conference on Computer and Communications Security (CCS ’06). ACM, 2006, pp. 89–98.
7. R. Canetti and S. Hohenberger, “ChosenCiphertext Secure Proxy Re-Encryption,” in Proceedings of the 14th ACM Conference on the Computer and Communications Security (CCS ’07). ACM, 2007, pp. 185–194.
8. . Liu, R. Zhang, and M. Zhao, ‘‘A robust authentication scheme with dynamic password for wireless body area networks,’’ Comput. Netw., vol. 161, pp. 220–234, Oct. 2019.
9. Manish potey, C A Dhote and Deepak H. Sharma,”Cloud Computing – Understand Risk, Threat, Vulnerability and Controls: A Survey”, International Journal of Computer Applications (0975 – 8887) Volume 67– No.3, April 2013.
10. Dr. L. Arockiam, S. Monikandan,” Data Security and Privacy in Cloud Storage using Hybrid Symmetric Encryption Algorithm”, International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 8, August 2013.
11. M.D. Dikaiakos et al., “Cloud Computing: Distributed Internet Computing for IT and Scientific Research,” IEEE Internet Computing, vol. 13, no. 5, pp. 10–13, 2009.
12. Darren Quick, Kim-Kwang Raymond Choo,” Google Drive: Forensic analysis of data Remnants”, Journal of Network and Computer Applications, Volume 40, Pages 179-193, April 2014.
13. Mayuer R. Palankar, Adrina lamnitchi, Matei Ripeanu and Simson Garfinkel ,” Amazon S3 for Science Grids: A Viable Solution”, DADC '08: Proceedings of the 2008 international workshop on Data-aware distributed computing, Pages 55–64, June 2008.
14. D Boland,” Securing Amazon Web Services (AWS) and Simple Storage Service (Amazon S3) Security”, an article regarding Amazon Simple Storage Service security, available at infosecwriters.com, June 2020.
15. D. R. Bharadwaj, A. Bhattacharya and M. Chakkaravarthy, "Cloud Threat Defense – A Threat Protection and Security Compliance Solution," 2018 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), Bangalore, India, pp. 95-99, 2018.
16. Pandian, A. Pasumpon, and S. Smys. "Effective Fragmentation Minimization by Cloud Enabled Back Up Storage." Journal of Ubiquitous Computing and Communication Technologies (UCCT) 2, no. 01 (2020): 1-9.
17. D.S.Tania Gaura, "A Secure and Efficient Client-Side Encryption Scheme in Cloud Computing," I.J. Wireless and Microwave Technologies, vol. 1, pp. 23-33, 2016.