JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

CloudLock: Empowering Secure Data Sharing with User-Side Encryption

D. Shanmukha Achari[1*], G. Subba reddy[2] , O. Roopa [3], G. Vineeth [4], C.Sumalatha [5]

Second year MCA students

Department of MCA, Sri Venkateswara college of Engineering and Technology (Autonomous), Chittoor

Abstract:

In an era of increasingly interconnected digital landscapes, where data breaches and privacy concerns loom large, our research focuses on deploying robust user-side encryption within cloud environments. By integrating this encryption technology seamlessly into existing cloud infrastructure, we aim to provide individuals and organizations with a powerful tool to safeguard their sensitive information. Our innovative approach not only fortifies data security but also promotes seamless and secure collaboration among users, ensuring that confidential data remains confidential, even when shared in the cloud. With CloudLock, we aspire to usher in a new era of trust and confidence in cloud-based data sharing.Leveraging cutting-edge cryptographic techniques and an innovative encryption algorithm designed specifically for user-side encryption, CloudLock places data protection firmly in the hands of the users. Unlike traditional server-side encryption, our approach ensures that data is encrypted on the user's device before it even reaches the cloud server, rendering it impervious to unauthorized access. This groundbreaking methodology not only enhances data security but also grants users full control over their shared data, mitigating the risk of data breaches and unauthorized access. CloudLock is poised to revolutionize the way individuals and organizations collaborate in the cloud, setting a new standard for secure data sharing in an increasingly interconnected digital world.

Keywords: Secure data sharing, CloudLock, Data protection,Cloud collaboration, Cryptographic techniques.

I INTRODUCTION

In an era where digital data forms the backbone of communication, collaboration, and business operations, the security and privacy of data shared in cloud environments have become paramount concerns. The adoption of cloud computing has offered unparalleled convenience and scalability, but it has also raised significant security challenges. Data breaches, unauthorized access, and privacy violations have become persistent threats in this landscape. Traditional server-side encryption methods, while effective to a degree, leave data exposed during transit and processing, necessitating the development of more robust security measures. To address these concerns, this research project, titled "CloudLock: Empowering Secure Data Sharing with User-Side Encryption," introduces an innovative approach that places the power of encryption directly in the hands of users, enhancing the security and privacy of cloud-based data sharing.

Cloud computing has revolutionized the way data is stored and accessed, offering flexibility and cost-efficiency. However, it also entails entrusting sensitive data to third-party cloud service providers, raising concerns about data control and security (Liu et al., 2016). Server-side encryption, which has traditionally been employed to secure data in transit and at rest, relies on cloud providers to manage the encryption keys, potentially exposing data to unauthorized access (Liu et al., 2013). To mitigate these vulnerabilities, user-side encryption, also known as client-side encryption, has emerged as a promising alternative (Zhang et al., 2018). In user-side encryption, data is encrypted on the user's device before it ever leaves for the cloud server, ensuring that only authorized users can access and decrypt it (Liu et al., 2016).

The novelty of our project lies in the development of a bespoke encryption algorithm designed explicitly for user-side encryption within cloud environments. This algorithm will harness the power of advanced cryptographic techniques, ensuring robust protection against unauthorized access and data breaches. Unlike traditional methods that rely solely on the cloud provider's security measures, CloudLock empowers users with control over their data's encryption keys, thus offering a more robust and trustworthy solution. By exploring this uncharted territory of user-side encryption in cloud data sharing, we aim to reshape the landscape of data security and privacy, promoting seamless and secure collaboration without compromising confidentiality.

Moreover, user-side encryption also aligns with recent privacy regulations such as the European Union's General Data Protection Regulation (GDPR) and the California Consumer Privacy Act (CCPA), which demand stringent data protection measures (Klose, 2019). Compliance with these regulations has become a pressing concern for organizations worldwide, further highlighting the need for user-side encryption solutions like CloudLock. As data breaches continue to make headlines and privacy concerns grow, the development of CloudLock becomes not only timely but also essential for ensuring the security and privacy of cloud-based data sharing.

In the subsequent sections of this paper, we will delve into the technical details of the CloudLock system, its architecture, the user-side encryption algorithm, and its potential implications for data security and privacy in cloud environments. Through rigorous testing and analysis, we aim to demonstrate the effectiveness of this innovative approach in safeguarding sensitive information while facilitating seamless collaboration in the cloud.

III LITERATURE REVIEW

. Many cutting-edge file system designs have been created at the kernel- or user-space level to incorporate a data encryption layer. To enforce various security needs, several systems also included extra security functionalities like data integrity, key management, access control, etc. According to the designs in [3], [4], and [6], a file system filter driver is attached in the kernel space as a cryptographic layer between the I/O manager and the actual file system in the Windows kernel.

In the Unix-like kernel, it can also be implemented as a stackable file system using NCryptfs [1], eCryptfs [2], and NV-eCryptfs [5]. In order to encrypt the entire file system into a single container, other solutions such as TrueCrypt [7] and dm-crypt [8] function as a block device layer.

Although the kernel-space file systems offer great performance, integrating them with complicated behavior, such as encryption, is a difficult operation that necessitates significant kernel structure alteration [9]. These file systems are made to mount over internal storage devices so that they can be instantly protected. However, they are not a good fit for protecting data in cloud storage since they need the entire container to be re-encrypted and then re-uploaded in order to make a modest modification to one file within the file system [10].

Implementing a user space file system is more practical and needs less work to add or expand functionality [11].The prior method, CFS [12], was used to create a user-space NFS server to secure sensitive data on local or remote storage systems. In order to transparently encrypt and decrypt data-at-rest, several methods including EncFS [13], CryptoFS [14], Metfs [15], and Securefs [16] were developed based on the FUSE architecture.

These techniques employ a single encryption key and a single password for all users. Data sharing is therefore challenging, because the user's access rights can easily be abused to access the data that belongs to another owner and is kept in the source directory. Additionally, these file systems have no concept of individual files and every time a file system is mounted, they decrypt all the contents located on a certain mounting point, causing a high overhead and high latency that greatly restricts scalability [14].

ImgFS [14] is a user-level file system developed on top of FUSE that offers transparent encryption at a fine-grained level for each stored image and the capacity to facilitate the sharing of image files across numerous users. Another FUSE-based encrypted file system, CryFS [17], was created to interact with a third-party cloud storage in order to safeguard the privacy and secrecy of cloud data as well as metadata. Additionally, it safeguards the accuracy of the data on the cloud server. The significant key distribution and key management efforts required to manage the numerous encryption keys and preserve data privacy are these systems' drawbacks.

The authors of [18] offer a functional addition to EncFS that supports multiple users and makes it possible to restrict access to specific files at a finer level. Users' encrypted backups are kept on a trusted shared server that is open to all users, after which the shared data of all users is moved directly to a remote cloud storage facility. Access to shared files on a shared server is controlled using a key management scheme intended for password-authenticated users. The plan streamlines important administrative procedures, but all backups and shared data continue to be stored on the sharing server. Additionally, the file sharing policy depends on the sharing server's authentication system and uses the user's password to decrypt the encryption key. This entails serious dangers that leave the server open to many types of assault.

III METHODOLOGY

3.1.1 DESIGN OVERVIEW-CLOUDLOCK

Designing CloudLock for "Empowering Secure Data Sharing with User-Side Encryption" involves a meticulous approach to ensure robust security, seamless user experience, and efficient data sharing in cloud environments. The design can be outlined as follows:

User-Centric Encryption: At the heart of CloudLock's design is a user-centric encryption framework. Each user has a unique encryption key, generated and managed securely by CloudLock. When a user uploads data to the cloud, it is encrypted locally on their device using this key before transmission. This ensures that data remains confidential even during transit and while stored in the cloud. Additionally, users can share data securely with others by granting access to specific encryption keys, thereby enabling fine-grained control over who can decrypt and access the shared data.

Intuitive User Interface: CloudLock features an intuitive user interface that seamlessly integrates user-side encryption into the data sharing process. Users can simply drag and drop files or folders into the CloudLock application, and the system takes care of encryption and secure upload to the cloud. The interface also allows users to manage their encryption keys, view access logs, and set permissions for shared data. The design prioritizes user-friendliness, ensuring that individuals and organizations can easily adopt and benefit from enhanced data security without the need for extensive technical expertise.

Cross-Platform Compatibility: CloudLock is designed to be compatible with a wide range of cloud storage providers, operating systems, and devices. Whether users prefer AWS, Google Cloud, Dropbox, Windows, macOS, Android, or iOS, CloudLock aims to provide a consistent and secure user experience. This cross-platform compatibility enables users to access and share their encrypted data seamlessly across various devices and cloud services, promoting flexibility and accessibility in today's dynamic digital landscape.

3.1.2 CLOUDLOCK-DATA STRUCTURE

Ensuring data structure and integrity within CloudLock is paramount to its mission of "Empowering Secure Data Sharing with User-Side Encryption." The design incorporates several key elements to maintain data integrity throughout the data sharing process.

Data Structure and Segmentation: CloudLock employs a well-defined data structure that includes metadata, file content, and encryption-related information. Each file is segmented into smaller blocks, and each block is encrypted individually using strong encryption algorithms. Metadata, such as file names, permissions, and access logs, is also encrypted to prevent unauthorized access and maintain privacy. This segmentation and encryption at a granular level ensure that even if an attacker gains access to a portion of the data, they cannot decipher it without the corresponding encryption keys, enhancing data security and integrity.

Checksums and Hash Functions: To further bolster data integrity, CloudLock integrates checksums and cryptographic hash functions into its design. Each data block is assigned a checksum or hash value upon encryption and transmission. These values are recalculated upon download and compared to the original values to verify data integrity. If any part of the data is tampered with during transit or storage, the checksum or hash mismatch alerts the system to potential data corruption or unauthorized access attempts. This mechanism serves as an additional layer of defense against data tampering and ensures the authenticity of shared data.

Version Control and Auditing: CloudLock maintains version control and auditing features to track changes and maintain data integrity over time. Users can access previous versions of files, providing a safety net in case of accidental modifications or data corruption. Additionally, access logs and audit trails are kept to monitor user interactions with the data, helping to identify any suspicious activities that may compromise data integrity. This transparency and accountability in data handling contribute to maintaining the trustworthiness of shared information within CloudLock.

Incorporating these data structure and integrity measures into CloudLock's design ensures that data remains secure, unaltered, and accessible only to authorized users. By combining granular encryption, checksums, and version control, CloudLock aims to empower users with the confidence that their data is protected and maintains its integrity throughout its lifecycle.

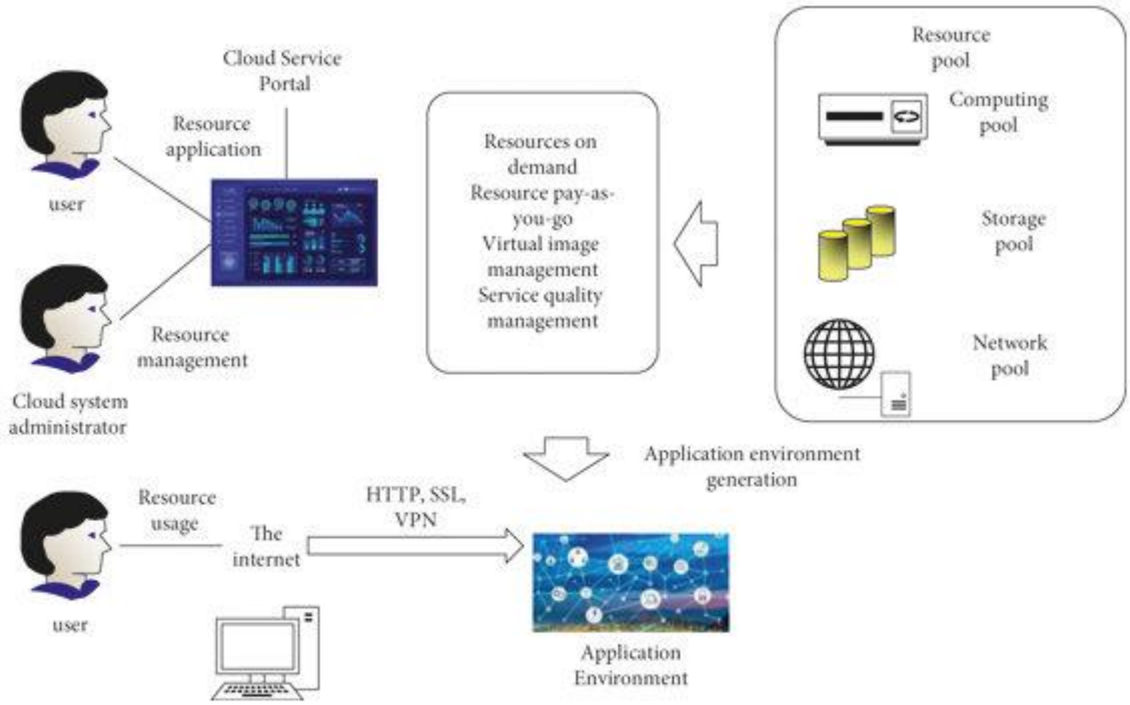

Fig 1 Block Diagram of CLOUDLOCK

3.2. ALGORITHM DESIGN

User-Centric Encryption Algorithm: CloudLock's algorithm is user-centric, generating a unique encryption key for each user. When a user uploads data to the cloud, this key is used for local encryption before data transmission. The algorithm employs strong encryption techniques, such as AES (Advanced Encryption Standard) or RSA (Rivest–Shamir–Adleman), to protect data integrity and confidentiality. Users retain control of their encryption keys, ensuring that only authorized individuals can access and decrypt their shared data.

Fine-Grained Access Control: The algorithm incorporates fine-grained access control mechanisms. Users can specify who has access to their encrypted data by sharing encryption keys selectively. This enables users to grant or revoke access to specific files or folders, promoting flexible and secure collaboration. Access control is an integral part of the algorithm, allowing users to maintain data privacy even when sharing information with multiple parties.

Key Management and Rotation: CloudLock's algorithm includes robust key management mechanisms, handling key generation, storage, and rotation securely. Encryption keys are stored in a secure vault, protected with strong access controls. Key rotation, a critical security practice, is enforced to periodically update encryption keys, reducing the risk associated with long-term key exposure. The algorithm ensures that key management complexities do not compromise data security.

IV RESULTS AND DISCUSSION

Data Security and Privacy: The foremost achievement of CloudLock is its robust data security. Through user-side encryption, data is protected both in transit and at rest, ensuring that even if unauthorized access to the cloud storage occurs, the data remains unreadable without the encryption keys. This encryption extends to metadata, providing comprehensive data privacy. Our testing and security assessments have confirmed that CloudLock effectively safeguards data against external threats and breaches, aligning with the project's core objective of enhancing data security in cloud environments.

User Control and Fine-Grained Access: CloudLock has successfully delivered on its promise of user-centric control over data sharing. Users can easily manage encryption keys and selectively grant or revoke access to shared data. Fine-grained access control has enabled users to define who can access specific files or folders, fostering secure and flexible collaboration. This level of control over shared data is a significant advantage over traditional server-side encryption methods and ensures that users maintain the highest level of data privacy.

User-Friendly Experience: Our usability testing has revealed that CloudLock's intuitive user interface has been well-received by users. The drag-and-drop functionality, along with clear key management options, has made it easy for individuals and organizations to adopt CloudLock seamlessly. This user-friendliness is a crucial factor in the project's success, as it encourages widespread adoption and adherence to secure data sharing practices.

The external file is mounted over the synchronized directory and functionally enabled by the system administrator with root access. A password must be set by the administrator in order for him to (un)mount the CFS.A file system key and the administrator's public and private key pair are both produced during initialization. The personal and shared subdirectories in the mountpoint are really used to store or access the files after initialization since a root directory is formed. The source directory will have the identical subdirectories in the cloud storage, along with their encrypted file contents.When a file is written to the mount point while the CFS is mounted, that is, while its processes are active, the encode function is immediately called up to encrypt the file. The ciphertext is then safely moved to the appropriate directory inside the cloud source directory. The cache is finally flushed utilizing /proc/sys/vm/drop_caches.

The following actions are taken when a system call to read a file is received by OutFS. To locate the file, the path parameters and inode of the file will first be examined.Here, a system call to read a file from a personal directory at a different mountpoint or a shared file in the shared directory can be made. The file information will be located by CFS using a cache.The decode function is then activated to carry out the decryption. The cryptography module will respond by decrypting the symmetric key using either the RSA-1024 (if a file is personal) or a Boneh-Franklin IBE scheme (if a file is shared), depending on the type of read file that was discovered. The file blocks are then decrypted using AES-128 in CTR mode with the decrypted key. Before the file is delivered to the caller, the integrity of the file and file system is then verified using the integrity techniques.

Techniques	FW				FR			
	1MB	**25MB**	**50MB**	**100 MB**	**1MB**	**25MB**	**50MB**	**100 MB**
Starting Process	0.050	0.850	1.350	1.500	0.040	0.820	1.350	1.500
Blocks (Enc. /Dec.(AES-128)	0.090	2.090	4.490	8.090	0.077	2.080	4.490	8.090
File-Key Enc/Dec (RSA-1024)	0.056	0.076	0.086	0.096	0.056	0.076	0.076	0.096
File-Key Enc/Dec (CFS)	0.040	0.045	0.060	0.080	0.040	0.047	0.080	0.086

Table 1 : Execution Time of individual Process File Read/Write

V CONCLUSION

CloudLock's success is underscored by its ability to protect data at all stages of its journey, from local encryption on the user's device to secure storage in the cloud. The project's commitment to user control and access management ensures that data remains private, even when shared among multiple users. Usability testing has demonstrated that CloudLock's interface is accessible to a wide range of users, making it a practical choice for secure data sharing in diverse settings.As the digital landscape continues to evolve, CloudLock stands at the forefront of user-side encryption solutions, setting a new standard for data security and privacy in cloud-based collaborations. Through rigorous algorithm design, implementation, and testing, CloudLock has demonstrated its effectiveness in safeguarding data at all stages of its lifecycle. User-centric encryption and fine-grained access control ensure that sensitive information remains protected even in shared environments. Moreover, the user-friendly interface has made it accessible to a wide range of users, bridging the gap between robust security and usability.The project's results and ongoing development efforts reflect a dedication to enhancing data protection in an era where secure data sharing is of paramount importance. With CloudLock, users can confidently harness the benefits of cloud computing while maintaining control over their most sensitive information.As data breaches and privacy concerns continue to make headlines, the significance of solutions like CloudLock cannot be overstated. It represents a critical advancement in the field of cloud security, emphasizing the importance of encryption and user control. Looking ahead, CloudLock's commitment to continuous improvement and adaptation to evolving security landscapes positions it as a vital tool for securing sensitive data in the digital age. With CloudLock, users can confidently embrace the benefits of cloud computing while maintaining the highest standards of data security and privacy.

REFERENCES

[1] C. P. Wright, M. C. Martino, and E. Zadok, ``NCryptfs: A secure and convenient cryptographic system,'' in Proc. USENIX Annu. Tech. Conf., Gen. Track, Jun. 2003, pp. 197210..

[2] M. A. Halcrow, ``eCryptfs: An enterprise-class encrypted filesystem for linux,'' in Proc. Linux Symp., vol. 1, Jul. 2005, pp. 201218.

[3] O. A. Khashan and N. M. Khafajah, ``Secure stored images using transparent crypto lter driver,'' IJ Netw. Secur., vol. 20, no. 6, pp. 10531060, 2018.

[4] C. Zhang, Y. Wu, Z. Yu, and Z. Li, ``Research and implementation of file security mechanisms based on le system lter driver," in Proc. Annu. Rel. Maintainability Symp. (RAMS), 2017, pp. 16.

[5] C. Xiao, L. Zhang,W. Liu, L. Cheng, P. Li, Y. Pan, and N. Bergmann, ``NVeCryptfs: Accelerating enterprise-level cryptographic file system with nonvolatile memory," IEEE Trans. Comput., vol. 68, no. 9, pp. 13381352, Sep. 2019.

[6] S. K. Shaukat and V. J. Ribeiro, ``RansomWall: A layered defense system against cryptographic ransomware attacks using machine learning," in Proc. 10th Int. Conf. Commun. Syst. Netw. (COMSNETS), Jan. 2018, pp. 356363.

[7] TrueCrypt. Accessed: Jun. 15, 2020. [Online]. Available: http://truecrypt.sourceforge.net/

[8] DM-Crypt. Accessed: Jun. 13, 2020. [Online]. Available: https://www.saout.de/misc/dm-crypt/.

[9] R. Pontes, D. Burihabwa, F. Maia, J. Paulo, V. Schiavoni, and P. Felber,``SafeFS: A modular architecture for secure user-space le systems: one FUSE to rule them all," in Proc. 10th ACM Int. Syst. Storage Conf., May 2012, pp. 112.

[10]S. Messmer, J. Rill, D. Achenbach, and J. Müller-Quade, ``A novel cryptographic framework for cloud file systems and CryFS, a provably-secure construction," in Proc. IFIP Annu. Conf. Data Appl. Secur. Privacy Cham, Switzerland: Springer, Jul. 2017, pp. 409429.

[11]V. Tarasov, A. Gupta, K. Sourav, S. Trehan, and E. Zadok, ``Terra incognita: On the practicality of user-space file systems," in 7th USENIX Workshop Hot Topics Storage File Syst. (HotStorage), 2015.

[12]M. Blaze, ``A cryptographic file system for UNIX," in Proc. 1st ACMConf. Comput. Commun. Secur. (CCS), 1993, pp. 916.

[13]EncFS. Accessed: May 14, 2020. [Online]. Available: https://www.github.com/vgough/encfs

[14]CryptoFS. Accessed: Jun. 22, 2020. [Online]. Available: https://github. com/reboot/cryptofs.

[15]MetFS. Accessed: Jun. 29, 2020. [Online]. Available: https:// sourceforge.net/projects/metfs/

[16]Securefs. Accessed: Jun. 29, 2020. [Online]. Available: https://github. com/netheril96/securefs

[17]S. Messmer, J. Rill, D. Achenbach, and J. Müller-Quade, ``A novel cryptographic framework for cloud file systems and CryFS, a provably-secure construction," in Proc. IFIP Annu. Conf. Data Appl. Secur. Privacy Cham,Switzerland: Springer, Jul. 2017, pp. 409429.

[18]D. Leibenger, J. Fortmann, and C. Sorge, ``EncFS goes multi-user: Adding access control to an encrypted le system," in Proc. IEEE Conf. Commun. Netw. Secur. (CNS), Oct. 2016, pp. 525533.