Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 44 No. 4 (2023)

Cloud Storage Data Security and Integrity Protection Using Localized and Customized Key Manipulator (LCKM) Algorithm

[1] Dr. Karthika RN, [2] Santhosh J , [3]Dr. V. Vijaya Chamundeeswari

[1] Assistant Professor, Department of IT, Saveetha Engineering College, Thandalam, Chennai 602105 Tamilnadu, India[1]
[2]Final Year Learner, Department of IT, Saveetha Engineering College, Thandalam, Chennai 602105, Tamilnadu, India[2]
[3] Professor, Department of CSE, Saveetha Engineering College, Thandalam, Chennai 602105, Tamilnadu, India[3]

E-mail: [1] karthikarn@saveetha.ac.in,[2] santhoshjai28@gmail.com, [3] vijaychamu@gmail.com

Abstract: In leveraging cloud storage services for efficient data storage while addressing data integrity concerns. Traditional data integrity auditing schemes necessitate private key usage, relying on hardware tokens and password memorization, which can be problematic if lost or forgotten. In prevailing methods data integrity checking without private key storage and key generation through third-party key distributors used, biometric data like iris scans and fingerprints, linear sketching and signatures are used to verify user data. These methods do not serve the purpose as biometrics has default drawbacks and third-party key generators can also not be trusted. To overcome these challenges, a pioneering approach called the proposed Localized key generator and storage is implemented in our proposed system. This LCKM algorithm generates customized keys for each file in a localized key generator and cloud details. Localized Key Manipulator will be provided to each user which is a password protector and the key customization will done according to the confidentiality, priority, and importance of the file. Customization of keys will be done with various combinations of digits, alphabets, symbols, and different sizes along with details of cloud storage chosen by the user. This localized key manipulator is mobile and carried as a protected file anywhere but only authenticated users can access it. Security and localized data integrity confirm the method's effectiveness, ensuring robust security and operational efficiency.

Keywords: Data Integrity, Private Key, localized key generator, customized key protection, cloud data security.

1. INTRODUCTION

Cloud storage services offer users a convenient and scalable solution for data storage. These services enable users to offload their data to the cloud, reducing the need for costly hardware and software maintenance, which comes with significant advantages. However, entrusting data to the cloud means relinquishing physical control, making it challenging to ensure the integrity of cloud-stored data in the face of hardware or software failures and human errors within the cloud infrastructure [1]. To address these concerns, numerous data integrity auditing schemes have been proposed, allowing data owners or Third Party Auditors (TPAs) to verify the integrity of data stored in the cloud. These schemes cover diverse aspects of data integrity auditing, including dynamic data operations, data and user identity privacy protection, key exposure resilience, certificate management simplification, and privacy-preserving authentication. In these schemes, users typically generate data authenticators using their private keys, necessitating secure storage and management of these keys. Users often require portable secure hardware tokens (e.g., USB tokens, smart cards) to safeguard their private keys, along with memorizing passwords to activate them, which can be cumbersome and prone to human error. Furthermore, the loss of the hardware token or forgotten passwords can render data integrity auditing ineffective, highlighting the need for alternative approaches that eliminate the reliance on private key storage [2]. One promising method in data authentication involves using biometric data, such as fingerprints and iris scans, as a private key, establishing a unique link with the user's identity. However, biometric data is inherently noisy and subject to variations due to various factors, making it unsuitable for direct use as a private key [3]. To ensure data security, the key manipulation process meticulously orchestrates cryptographic steps, beginning with the generation of a public-private key pair using a trusted cryptographic tool [4] [5]. In addition to this, password managers and strength checkers are recommended for secure password management, while regular password changes and avoiding password sharing are fundamental security practices. For enhanced security, two-factor authentication is advised, and key manipulators can be augmented with cloud details and exported with password protection [6].

2. RELATED WORK

H. Dewan and R. C. Hansdah provide an insightful exploration of cloud storage technologies [7]. This comprehensive survey delves into the landscape of cloud storage solutions, elucidating their features, capabilities, and the evolving paradigms of data management in the cloud. The authors systematically analyze various aspects, including data integrity, security, scalability, and accessibility, shedding light on the key challenges and trends shaping the field of cloud storage. S. G. Worku, C. Xu, J. Zhao, and X et al. in the realm of cloud storage, data privacy, and integrity are paramount concerns present an innovative scheme that addresses these critical issues [8]. Their scheme not only ensures the integrity of data stored in the cloud but also preserves user privacy during the auditing process. By introducing a secure and efficient approach, the authors offer a significant contribution to enhancing the trustworthiness and reliability of cloud storage systems.

B. Wang, B. Li, and H. Li offers a pioneering solution to the intricate challenges of privacy and security in cloud-based data-sharing scenarios involving large user groups [9]. Their proposed system, Knox, introduces innovative techniques for privacy-preserving data auditing, ensuring that shared data integrity remains uncompromised while safeguarding the confidentiality of user information. This article serves as a valuable resource for both practitioners and researchers engaged in the development and implementation of privacy-preserving solutions in cloud computing environments. By addressing the ever-growing concerns surrounding data privacy and security in cloud environments, J. Yu et al. introduce an auditing mechanism that incorporates key-exposure resistance, fortifying the confidentiality and integrity of stored data [10]. This research article not only provides a comprehensive understanding of the challenges in cloud storage auditing but also offers an innovative solution that reinforces cloud data security.

A. Sahai and B. Waters discuss a novel cryptographic paradigm that extends the realm of identity-based encryption [11]. Fuzzy identity-based encryption addresses the inherent limitations of conventional identity-based encryption systems by allowing for more flexible and nuanced access control. This innovative approach enables encryption and decryption based on approximate or fuzzy identities, enhancing the practicality and adaptability of cryptographic systems [12].

3. IMPLEMENTING LOCALIZED AND CUSTOMIZED KEY MANIPULATOR (LCKM) ALGORITHM

A critical consideration in this process is the selection of an appropriate key length, typically ranging from 2048 to 4096 bits, to bolster encryption security. The public key, a pivotal element, must be stored securely for later use in cloud storage service authentication, while the private key, equally essential, demands utmost protection to prevent unauthorized access. This procedure involves associating the public key with the user's identity within the cloud storage service, cementing a secure link. During user registration or setup, the public key is furnished to the cloud service to strengthen authentication. The public key is employed to encrypt data before transmission to the cloud, ensuring confidentiality. The cloud service, armed with the user's private key, can securely decrypt the data. Regular key updates are included to enhance overall system resilience. In the realm of password generation, complexity is paramount, considering factors like length and character diversity. Character sets, including uppercase letters, lowercase letters, numbers, and special symbols, are thoughtfully chosen to create resilient passwords. Randomness and unpredictability are emphasized, discouraging the use of easily guessable information.

The key manipulation process involves a series of algorithmic steps, implemented with precision and care, to ensure the security and integrity of cryptographic keys used in various applications. To initiate this procedure, the first step is to generate a key pair using a reliable cryptographic library or tool. This pair consists of a public key and a private key, which are fundamental components of the encryption process. Security considerations play a pivotal role in this process, prompting the selection of an appropriate key length, typically ranging from 2048 to 4096 bits, to bolster the encryption's robustness. The public key, a crucial component, must be securely stored as it will later be shared with a cloud storage service for authentication purposes. Conversely, the private key, which is equally pivotal, must be safeguarded with utmost care, as its compromise could lead to unauthorized access. The next step involves linking the public key to the user's identity or account within the cloud storage service, establishing a secure association. During user registration or setup, the public key is furnished to the cloud service, further fortifying the authentication process.

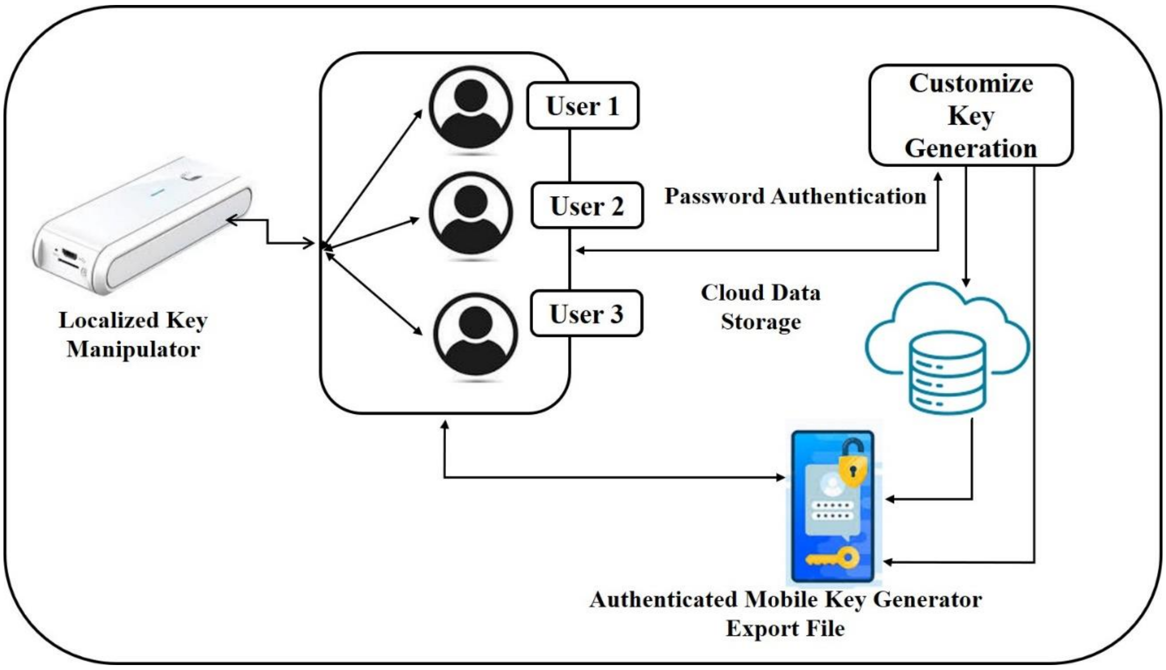

Fig 1: Architecture of Localized and Customized Key Manipulator Algorithm

Subsequently, in Fig 1. the generated public key is employed for data encryption before transmitting it to the cloud, ensuring that sensitive information remains confidential. The cloud service, equipped with the user's private key, can then decrypt the data securely. To maintain the highest levels of security, the procedure includes periodic key updates, enhancing overall system resilience against potential threats. Shifting our focus to password generation, it commences with determining the desired complexity level, encompassing factors such as password length and character types. By meticulously selecting character sets, including uppercase letters, lowercase letters, numbers, and special symbols, the resulting password becomes more resistant to brute-force attacks.

In this context, the generation process prioritizes randomness and unpredictability, discouraging the use of easily guessable information such as birthdays or common words. Instead, a random sequence of characters is generated, adhering to minimum length requirements, often stipulating a minimum of 12 characters. Variety is a key generator in password complexity, necessitating the inclusion of a diverse mix of character types. Recognizing and avoiding common patterns, such as "12345" or "password," becomes paramount in the quest for a secure password. Personal information, such as names and addresses, should be strictly avoided to thwart potential breaches. Leveraging a password manager is a sound practice, facilitating the generation and secure storage of complex passwords. Password strength checkers serve as valuable tools to assess the password's security level. The following algorithm describes key generations step by step from key generation, cloud storage, authentication security level, etc.

```
# Generate a Key Pair
public_key, private_key = generate_key_pair (2048)  # Use a cryptographic library/tool
save_public_key(public_key) # Implement a secure storage mechanism
```

```
save_private_key(private_key) # Implement a secure storage mechanism

# Associate Public Key with User's Identity
associate_public_key_with_user(user_id, public_key)
cloud_service.register_user_with_public_key(user_id, public_key)

# Encrypt Data Before Sending to Cloud
encrypted_data = encrypt (data, public_key)

# Cloud Service Decrypts Data with Private Key
decrypted_data = cloud_service.decrypt(encrypted_data, user_id)

# Verify User's Identity with Private Key
authenticated_user = verify_identity(decrypted_data, private_key)

# Periodically Update Keys
if time_to_update_keys():
 new_public_key, new_private_key = generate_key_pair(4096)
save_private_key(new_private_key)
associate_public_key_with_user(user_id, new_public_key)

# Generate a Secure Password
password = generate_secure_password()
if not meets_minimum_requirements(password): raise PasswordRequirementsError

# Store or Manage Password Securely
store_password_securely(password)

# Enable Two-Factor Authentication (if possible)
enable_two_factor_authentication()

# Add Cloud Details in Each Key for Storage Identification
public_key_with_cloud_details = add_cloud_details(public_key)

# Export Key to Mobile File with Password Protection
export_key_to_mobile_file (private_key, password)
```

Security remains a dynamic process, and thus, regular password changes are recommended to mitigate potential vulnerabilities. It is imperative never to share passwords or jot them down in easily accessible locations. When the need arises for password storage, opt for a secure and encrypted method to safeguard this sensitive data. For an added layer of security, enabling two-factor authentication (2FA) wherever possible is advisable. Finally, in the context of cloud storage, key manipulators can be augmented with cloud details to facilitate storage identification. Moreover, these key manipulators can be exported as mobile files with password protection, reinforcing the overall security architecture.

4. PERFORMANCE ANALYSIS

This analysis emphasizes the critical importance of secure key manipulation processes and password generation techniques in the realm of data security. A comprehensive analysis of these procedures reveals several key takeaways. Firstly, the establishment of an appropriate key length, typically within the range of bits, is paramount in cryptographic key generation. In Fig. 2 the data integrity analysis in accordance with its privacy level checked for different users for state-of-art key generation using the Data Integrity method Vs proposed LCKM.

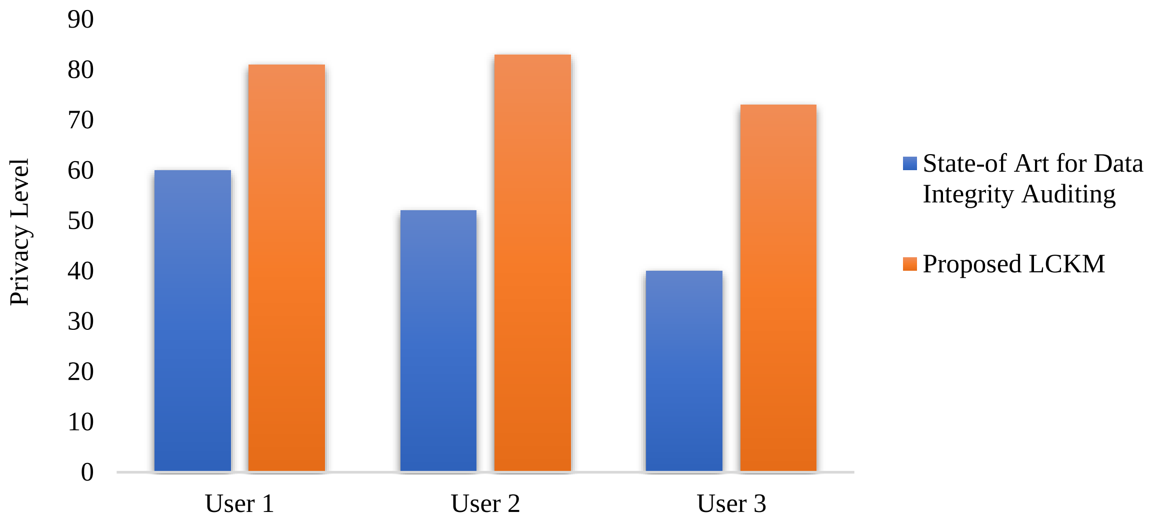

Fig 2: Data Integrity Analysis of Data Integrity Auditing vs LCKM Algorithm

This choice directly impacts the encryption's robustness and resistance to attacks. Secure storage of the public key, alongside the stringent protection of the private key, is imperative to prevent unauthorized access and maintain the integrity of cryptographic systems. The association of the public key with a user's identity in cloud storage authentication fosters a secure link between the user and their data. Regular key updates fortify the system's resilience against evolving security threats as in Fig 3.

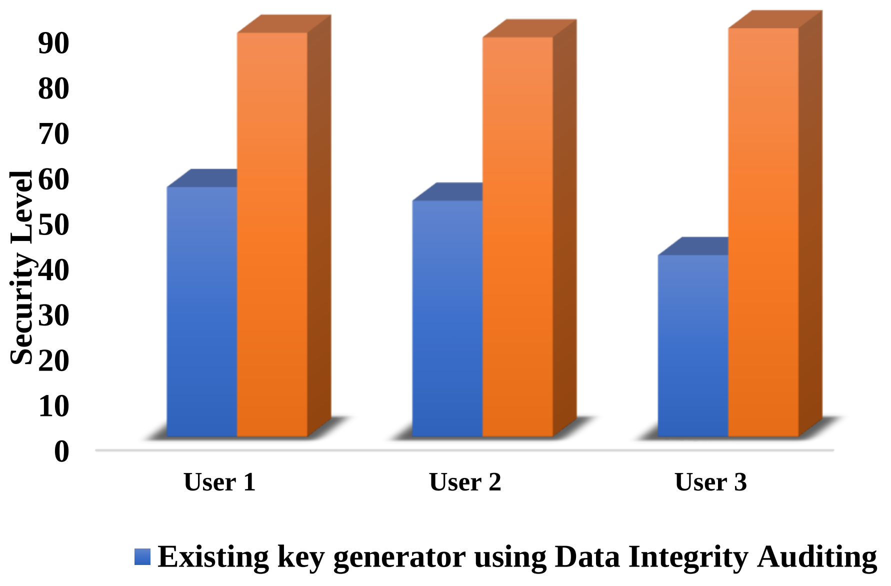

Fig 3: Security Level Analysis comparisons between State-of-art vs. proposed LCKM

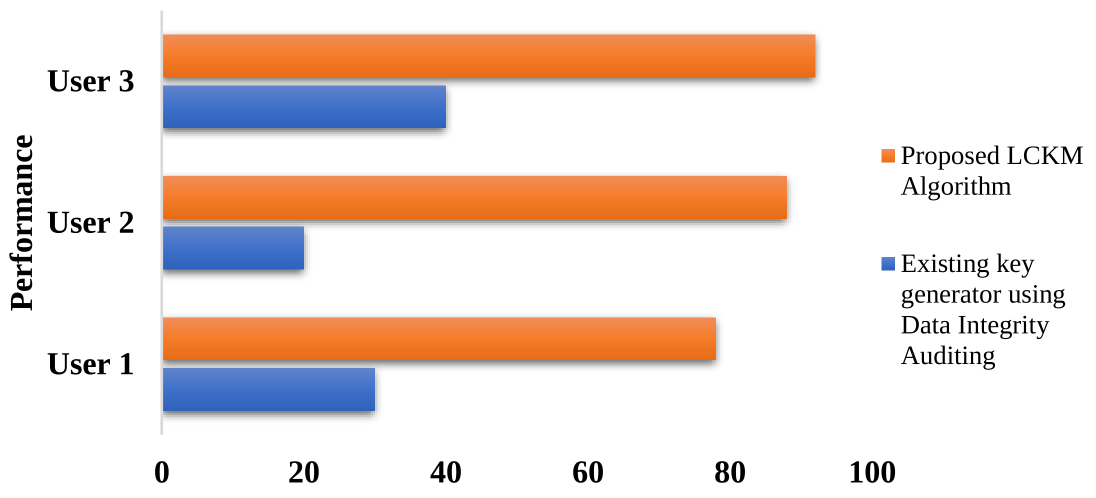

Fig 4: Overall Performance Analysis of Existing Vs Proposed Method

Fig 4 it depicts the performance analysis for generating, storing, and assessing the security of their passwords for existing methods and proposed work. Additionally, the practice of regular password changes and the avoidance of sharing or recording passwords in easily accessible locations are fundamental for maintaining robust security.

5. CONCLUSION

In conclusion, the meticulous key manipulation process, encompassing both cryptographic key management and password generation, serves as a robust framework for bolstering data security in this proposed technique. The selection of appropriate key lengths, ranging from 2048 to 4096 bits, is pivotal for encryption security. Secure storage of the public key, coupled with the stringent protection of the private key, forms the cornerstone of cloud storage service authentication. Regular key updates enhance the overall resilience of the system against emerging threats. In parallel, password generation prioritizes complexity, randomness, and the avoidance of easily guessable information. The use of diverse character sets and the elimination of common patterns contribute to the creation of resilient passwords. Going forward, future enhancements in this field may involve the integration of advanced cryptographic techniques and the development of even more secure password management solutions to address the evolving landscape of cybersecurity challenges. Continuous vigilance and adaptation to emerging threats will remain central to maintaining the integrity and security of cryptographic keys and passwords in the digital age.

REFERENCES

[1] C. Ellison and B. Schneier, “Ten risks of PKI: What you’re not being told about public key infrastructure,” vol. 16, no. 1, 12 2000.

[2] K. Ren, C. Wang, and Q. Wang, “Security challenges for the public cloud,” IEEE Internet Computing, vol. 16, no. 1, pp. 69–73, Jan 2012.

[3] A. F. Barsoum and M. A. Hasan, “Provable multicopy dynamic data possession in cloud computing systems,” IEEE Transactions on Information Forensics and Security, vol. 10, no. 3, pp. 485–497, March 2015.

[4] N. Garg and S. Bawa, “Rits-mht: Relative indexed and time stamped Merkle hash tree based data auditing protocol for cloud computing,” Journal of Network & Computer Applications, vol. 84, pp. 1–13, 2017.

[5] H. Jin, H. Jiang, and K. Zhou, "Dynamic and public auditing with fair arbitration for cloud data," IEEE Transactions on Cloud Computing, vol. 13, no. 9, pp. 1–14, 2014.

[6] H. Wang, Q. Wu, B. Qin, and J. Domingo-Ferrer, "Identity-based remote data possession checking in public clouds," IET Information Security, vol. 8, no. 2, pp. 114–121, March 2014.

[7] H. Dewan and R. C. Hansdah, "A survey of cloud storage facilities," in 2011 IEEE World Congress on Services, July 2011, pp. 224–231.

[8] S. G. Worku, C. Xu, J. Zhao, and X. He, "Secure and efficient privacy-preserving public auditing scheme for cloud storage," Comput. Electr. Eng., vol. 40, no. 5, pp. 1703–1713, Jul. 2014.

[9] B. Wang, B. Li, and H. Li, "Knox: privacy-preserving auditing for shared data with large groups in the cloud," in International Conference on Applied Cryptography and Network Security, 2012, pp. 507–525.

[10] 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, 2015.

[11] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in Advances in cryptology—EUROCRYPT 2005, ser. Lecture Notes in Comput. Sci. Springer, Berlin, 2005, vol. 3494, pp. 457–473.

[12] J. Yu and H. Wang, "Strong key-exposure resilient auditing for secure cloud storage," IEEE Transactions on Information Forensics and Security, vol. 12, no. 8, pp. 1931–1940, Aug 2017.