International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Secure Cloud ECC: Advancing Multi-Tenant Security with an Enhanced Embellished Version of ECC Algorithm

Lalit Kumar[1], Nidhi Tyagi[2], Seema Gupta[3]

Submitted: 15/01/2024 **Revised**: 23/02/2024 **Accepted**: 01/03/2024

Abstract: This paper proposes Embellished Elliptic Curve Cryptography (EECC), a new approach to secure data in multi-tenant cloud storage. EECC leverages the strengths of both ECC and the Diffie-Hellman algorithm, offering robust data protection with smaller key sizes and minimizing key exchange risks. Importantly, EECC outperforms existing methods in various aspects: significantly faster encryption and decryption times, lower computation costs for data transfer, smaller ciphertexts leading to reduced storage needs, and faster key generation. These improvements translate to a more efficient and user-friendly cloud storage experience. While security remains paramount, EECC's efficiency and reduced storage footprint make it a compelling option for cloud storage solutions. Further research could explore additional EECC optimizations and real-world performance evaluation

Keywords*: Multi-tenant cloud storage, Security, Elliptic Curve Cryptography (ECC), Diffie-Hellman (DH) algorithm, Embellished Elliptic Curve Cryptography (EECC),*

1. Introduction

Securing multi-tenant environments, where multiple users or organizations share a common infrastructure, presents unique challenges. Traditional security solutions often struggle to balance the need for robust data protection with efficient resource utilization and isolation of tenant data. Elliptic Curve Cryptography (ECC) has emerged as a promising approach due to its efficient key sizes and fast operations, making it suitable for resource-constrained environments. However, leveraging ECC in multi-tenant settings requires careful consideration to address potential vulnerabilities and ensure the confidentiality, integrity, and isolation of data belonging to different tenants.

This paper proposes a novel embellished version of the ECC algorithm specifically designed to enhance the security of multi-tenant environments. Our proposed approach aims to address the inherent challenges of shared infrastructure while maintaining the efficiency benefits of ECC. We delve into the limitations of existing ECC implementations in multi-tenant settings and propose modifications to the algorithm that mitigate these vulnerabilities. We further discuss the security features incorporated into our modified ECC and its potential advantages over traditional approaches in securing data within a multi-tenant environment.

This introduction sets the stage for the paper by highlighting the challenges of multi-tenant security and the potential of ECC as a solution. It also introduces the concept of our proposed modified ECC algorithm and outlines its key features and benefits. The subsequent sections of the paper will delve deeper into the technical details of the proposed algorithm, its security analysis, and performance evaluation.

2. Literature Review

Cloud storage has become ubiquitous, offering convenient and scalable data storage solutions for individuals and organizations alike. However, the inherent reliance on third-party providers raises concerns about data security and privacy. This literature review examines ten research papers exploring various aspects of cloud security, key management, and broader cyber security challenges.

Cloud Storage Security and Key Management:

The authors delve into cloud storage security and key management strategies. Enthoti et al. propose a combined compression and security model for data access control in the cloud. This model aims to address both security concerns and storage efficiency by utilizing a novel access control mechanism alongside data compression techniques. Manthiramoorthy et al. provide a comparative analysis of several encrypted cloud storage platforms, evaluating their security features, functionalities, and potential vulnerabilities. Their work highlights the diverse

[1]*Research Scholar, Department of Computer Science & Engineering, Shobhit Institute of Engineering & Technology, (NAAC Accredited Grade "A", Deemed to- be- University), Meerut, India*
lalit.chahal@gmail.com
[2]*Professor, Department of Computer Science & Engineering, Shobhit Institute of Engineering & Technology, (NAAC Accredited Grade "A", Deemed to- be- University), Meerut, India*
nidhi.tyagi@shobhituniversity.ac.in
[3]*Associate Professor, Ideal Institute of Management & technology, GGSIP University, Delhi, India*
seemagupta.iimt@gmail.com

security approaches adopted by various cloud storage providers and underscores the need for informed selection based on individual security requirements. [1] [2]

Singh et al. introduce a secure mechanism for anonymous authentication and key establishment using elliptic curves. This mechanism targets improved security for the Internet of Things (IoT) and cloud environments, where secure user authentication and key exchange are crucial for maintaining data confidentiality and integrity. [3] Irshad et al. propose a novel approach to enhance cloud-based inventory management. Their approach leverages a hybrid blockchain architecture, incorporating Generative Adversarial Networks (GANs) and Elliptic Curve Diffie-Hellman (ECDH) techniques. This combination aims to bolster data security and integrity within the inventory management system. [4]Finally, Kaleem et al. explore the Salp Swarm Algorithm for cryptographic key generation in cloud computing. Cryptographic keys play a vital role in securing data access and communication, and their generation process needs to be robust to prevent potential attacks. [5] This work demonstrates the potential of nature-inspired optimization algorithms for secure key generation in cloud environments.

These papers collectively showcase the ongoing research efforts towards robust security solutions for cloud storage and associated systems. They explore diverse methodologies, including encryption, access control models, blockchain, and novel optimization algorithms, demonstrating the evolving landscape of cloud security research.

The papers explore various aspects of cyber security beyond cloud storage, with a focus on cryptographic techniques and system security enhancements. Hankerson and Menezes provide a comprehensive reference on Elliptic Curve Cryptography (ECC), a widely used and efficient cryptographic technique employed in several of the reviewed papers. ECC offers several advantages over traditional public-key cryptography, including smaller key sizes and faster computation, making it a valuable tool for securing various applications. [6]

Shandilya et al. propose a Modified Firefly Optimization Algorithm for an Intrusion Detection System (IDS). Intrusion detection plays a crucial role in identifying and responding to cyberattacks. This work introduces a novel optimization algorithm inspired by the behavior of fireflies to enhance the efficiency and accuracy of intrusion detection systems, contributing to a more robust defense against cyber threats. [7] Pisla et al. explore the unique challenges of securing microgrids, distributed power systems that combine renewable energy sources with conventional generation. Their paper discusses various considerations for enhancing microgrid cybersecurity, including secure communication protocols, access control mechanisms, and cyber-physical security measures, aiming to protect these critical infrastructure systems from cyberattacks. [8]

Miyamoto et al. introduce a cybersecurity-enhanced encrypted control system using Keyed-Homomorphic Public Key Encryption (KH-PKE). Control systems are critical for managing various industrial processes and infrastructure, and securing them against cyberattacks is paramount. This work proposes a novel approach that leverages KH-PKE to enable secure communication and computation on encrypted data within control systems, offering a balance between security and functionality. [9]

Finally, Lee and Lee evaluate the vulnerability of YOLOv5, a popular object detection algorithm, to adversarial attacks. Machine learning (ML) models are increasingly being deployed in various applications, and their security becomes a pressing concern. This work highlights the susceptibility of YOLOv5 to carefully crafted adversarial attacks that can manipulate the model's output, emphasizing the importance of developing robust security measures for ML models to ensure their reliability and trustworthiness. [10]

These papers showcase diverse research areas within the broad domain of cybersecurity. They explore various methodologies, ranging from foundational cryptographic techniques like ECC to novel optimization algorithms for intrusion detection and secure control systems. Additionally, they highlight the growing need for securing ML models against adversarial attacks. This collective research contributes to a comprehensive understanding of the evolving landscape of cybersecurity threats and potential solutions.

In conclusion, this review explored research papers addressing various aspects of cloud storage security, key management, and broader cybersecurity challenges. The reviewed papers demonstrate the continuous development of novel security solutions and highlight the critical role of cryptography in securing data and systems.[11]

3. Methodology

In cloud computing, multi-tenancy refers to a paradigm where multiple users share access to resources, configurations, and infrastructure, each with designated access privileges. This approach offers advantages for both users and cloud service providers through optimized resource utilization and service delivery.

However, implementing multi-tenancy within cloud architecture introduces certain challenges that require effective solutions. Two primary concerns are task scheduling and security.

To address these challenges, a novel multi-tenant cloud framework is proposed. This framework incorporates

aembellished Elliptic Curve Cryptography (ECC) encryption/decryption technique to enhance security while ensuring secure key generation and transfer. The specific flow of the proposed Embellished ECC (EECC) algorithm is presented in a separate section.

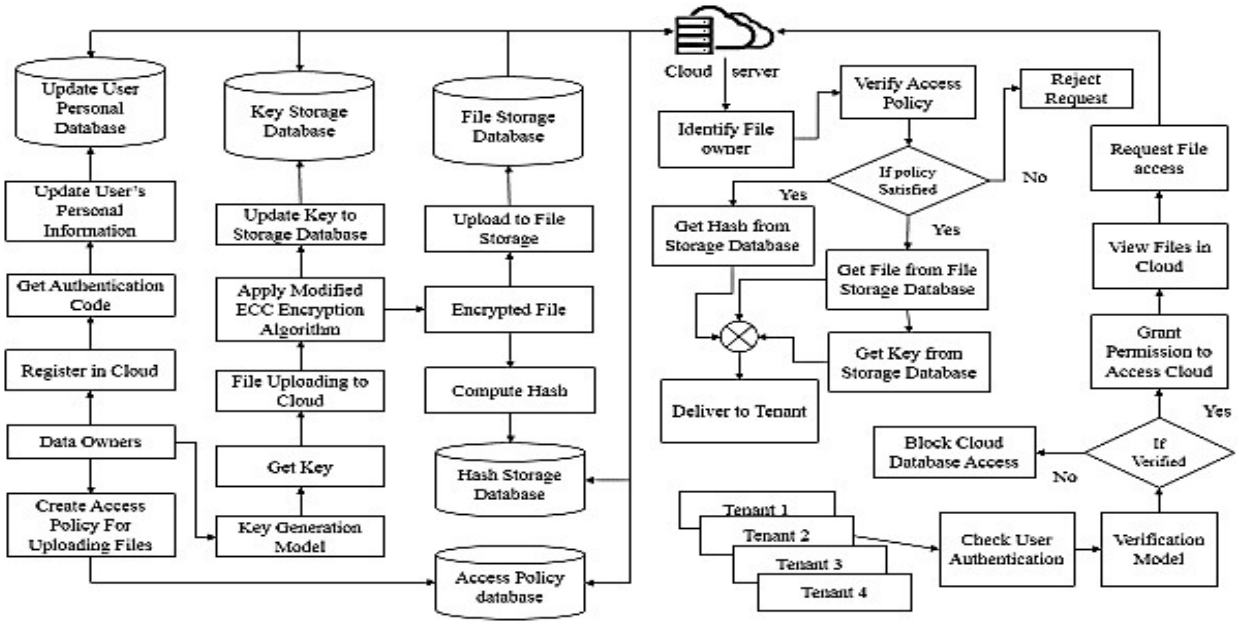

Fig 1: Proposed methodology for research

Within cloud computing, authorized users known as data owners can leverage the platform for secure file storage. To ensure data confidentiality, the proposed Embellished Elliptic Curve Cryptography (EECC) algorithm encrypts the owner's files. A secure key generation model creates unique encryption keys, which are then stored alongside the data owner's identity in a dedicated key storage database. Encrypted files are stored separately, and their corresponding hashes are saved in a hash storage database to verify data integrity. Additionally, data owners' personal information is anonymized for enhanced security before storage.

Data owners have the privilege to define access control policies, allowing them to control how other users, referred to as tenants, can access their uploaded data. This empowers them to manage access rights and grant specific permissions. However, data owners, particularly those with large volumes of private data (individuals or enterprises), might face limitations due to restricted storage capacities within the server configuration. To address this challenge, a communication module connects various cloud databases, forming a unified database management system that optimizes resource utilization.

Users interact with the system by submitting queries, which are categorized as either read-only or read-write operations. The database management system governs the entire database and can modify queries to ensure efficient resource allocation. Users can download encrypted files (cipher texts) from the server.

The data access process commences with data owners uploading large datasets to the cloud storage. The ECC-based Diffie-Hellman (DH) algorithm then encrypts the data using a unique key. If a user requests access to specific data, the cloud system transmits the corresponding key via email upon receiving the request. The system initiates a search for the relevant data, starting with data owner. If the data is not found, the search continues with other data owners. Once the system locates the pertinent data, it delivers it to the user along with the decryption key, enabling them to access the information.

Proposed Embellished Elliptic Curve Cryptography (EECC)- workflow

Improved Cloud Security and Access Control in a Multi-Tenant Architecture

This research proposes a novel framework to enhance security and access control within a multi-tenant cloud environment. The core of this system lies in a hybrid cryptographic approach that leverages the strengths of both Elliptic Curve Cryptography (ECC) and the Diffie-Hellman (DH) algorithm. This combined approach addresses limitations like computational overhead and large key sizes commonly associated with traditional methods.

The proposed framework utilizes DH for secure key generation, minimizing key exchange operations while ensuring increased security. Tenant data uploaded to the cloud is encrypted using the Embellished Elliptic Curve Cryptography (EECC) algorithm, offering robust data confidentiality.

Secure Data Access for Tenants:

When a tenant requests access to specific data, the system first verifies their legitimacy through a dedicated verification model. Authorized tenants are then granted file visibility based on pre-defined access policies. Only tenants with explicit permission can directly access cloud files.

Upon receiving access permission, a secure key transfer process takes place. The encrypted data is transferred along with a hash key file, ensuring data integrity for the tenant. These innovative techniques significantly improve multi-tenancy performance within the cloud architecture.

EECC: A Secure and Efficient Algorithm:

The EECC algorithm offers high security for cloud data storage and utilizes a symmetric encryption technique for reliable data protection. This approach employs smaller key sizes, reducing computational overhead compared to conventional methods. The EECC framework follows a specific process outlined in the provided pseudocode. It starts by generating public and private key pairs for both the client and server. Secure key exchange between them is then facilitated using EECC.

Table 1: EECC Algo

Step 1: Initialization
D_o ← dataowner
C_{sp} ←CloudServiceProvider
Doac ← DataOwnerAuthenticationCode
Doac = create the D_o message confirmation code.
Sd ←SetData
P_uk ← publickey
P_r*k* ← privatekey
P_r*k* = *length*(P_r*k*)/2
Puk = P_r*k* ∗ k

Step 2: MECC Encryption
Ed ←E*ncryptedData*
while (Sd)
Ed = Sd.data +(k*Pu) *Fl
End while

Step 3: Access Control Policy
L_{od} ←Length *of data*
For x=0 to *L_{od} do*
Sd = SetData
End for
Tl←T*enantsList*
C_t ←Chosen*Tenants*
For x = 0 to *Tl do*
C_t = selectedtenants
End for

Step 4: Create access control policy
Doac ← DataOwnerAuthenticationCode
Dac = create the S_d message verification code.
Uac ← UserAuthenticationCode
Uac= provide the user with the message verification code.
L_{od} ←Length *of data*
For x=0 to *L_{od} do*
Sd = SetData
End for
Rh ← Obtain user's data hashing via C_{SP}.
Get key of Data
Ask for data to C_{sp}

Doacrd ← *DataAuthenticationCode*
Dacrd = *provide the message verification code against the data you've obtained.*
H_{res} ←*hashing results*
if Dacrd =Rh then
H_{res} =*hash is verified successfully*
Else
H_{res} = *hashing unverified*
End if

Step 5: EECC Decryption
*CT*1←*ciphertext*1
[isolated decryption intended at both Keys]
*CT*1←*Key*1 provided by *DH*
CT2←*Key*2 *provided by DH*
*CT =CT*1+*CT*2
DE_d ←*decrypted data*
*CT*1=$k * P_r k$
while DE_d.*data*
End while

The data encryption process involves two steps: first, the plain text is encrypted with the server's public key using ECC, resulting in cipher text. Subsequently, the private key is split into two parts based on size and encrypted using the DH algorithm. These encrypted key components are then stored in the cloud. Essentially, the ECC key is divided and secured through DH encryption.

During decryption, the two key components are retrieved and decrypted using DH, then combined. Finally, the combined key is used with ECC decryption to recover the original plain text in a more secure manner compared to traditional methods.

This framework effectively addresses security concerns and access control challenges in multi-tenant cloud environments, making it a valuable contribution to secure cloud storage solutions.

Mathematical Model of the Proposed System

Elliptic Curve Diffie-Hellman (ECDH) for Secure Key Exchange:

The proposed system leverages Elliptic Curve Diffie-Hellman (ECDH) to establish a secure shared secret key between the cloud server and tenants without directly exchanging private keys. This enhances security by eliminating the risk of key interception during communication.

ECDH Key Exchange Process:

1. **System Setup:**
   - The system selects a suitable elliptic curve and a base point on that curve (publicly known).
2. **Key Generation:**
   - **Server:**
     - Generates a random private key (d_s)
     - Computes its public key (Q_s) by multiplying the base point (P) by the private key (d_s): Q_s = d_s * P
   - **Tenant:**
     - Generates a random private key (d_t)
     - Computes its public key (Q_t) using the same formula: Q_t = d_t * P
3. **Key Exchange:**
   - **Server:**
     - Sends its public key (Q_s) to the tenant.
   - **Tenant:**
     - Sends its public key (Q_t) to the server.
4. **Shared Secret Calculation:**
   - **Server:**
     - Calculates the shared secret key (K) by multiplying its private key (d_s) with the tenant's public key (Q_t): K = d_s * Q_t
   - **Tenant:**
     - Independently calculates the same shared secret key (K) using its private key (d_t) and the server's public key (Q_s): K = d_t * Q_s

Crucially, although both the server and tenant perform the same calculation steps with different private and public keys, they arrive at the same shared secret key (K) due to the mathematical properties of elliptic curves.

This shared secret key can then be used for secure communication and data encryption within the proposed system, enhancing overall security in the multi-tenant cloud environment.

4. Results and Discussion

The proposed system underwent performance analysis evaluated through various metrics, including encryption and decryption times, upload and download computation times.

Encryption

Data encryption safeguards information by transforming it into an unreadable format (cipher text) accessible only by authorized individuals using a specific key. The original, unencrypted data is referred to as plain text.

EECC Performance Comparison:

Table 3 showcases the proposed EECC method's encryption time, comparing it to existing methods like RSA, MRSA, and MRSAC across various key lengths (100, 128, 256, 512, 1024, 2048, and 4096 bits).

The results demonstrate that EECC exhibits significantly faster encryption times compared to other methods, especially for smaller key lengths. With a 100-bit key, MECC's encryption time is remarkably low (5ms), contrasting significantly with MRSA's time of 222ms, primarily due to its lengthy key generation process. Even for larger key lengths (4096 bits), MECC's encryption time remains efficient at 51ms, whereas existing methods reach 92ms.

This performance analysis emphasizes the efficiency of the EECC algorithm in terms of encryption speed, making it a valuable choice for secure data storage in cloud environments.

Decryption

Data decryption is the process of unlocking scrambled information, transforming it back into its original, understandable form. This transformation requires a specific key, similar to unlocking a door with the right key. Decrypted data becomes accessible and interpretable, allowing authorized users to retrieve the intended message or information.

Table 2: Decrypting time

Key length (in bit)	Existing	RSA	MRSA	MRSAC	Proposed
100	016	088	106	211	011
128	031	187	121	187	025
256	047	063	155	201	036
512	063	217	967	687	051
1024	078	1452	6937	7037	063
2048	108	15202	53608	83708	089
4096	186	18382	10956	10956	158

Decryption Performance of the EECC Algorithm

Table 4 compares the decryption time of the proposed EECC method with other established methods like RSA, MRSA, and MRSAC across various key lengths (100, 128, 256, 512, 1024, 2048, and 4096 bits).

The analysis reveals that EECC consistently outperforms existing methods in terms of decryption speed. This is particularly evident for smaller key lengths. At a key length of 100 bits, EECC's decryption time is only 11ms, significantly faster than MRSAC's encryption time of 212ms. Even for larger key lengths (4096 bits), EECC maintains a faster decryption time of 159ms, compared to the considerably slower decryption times of MRSA and MRSAC (10,957ms).

These results highlight the efficiency of the EECC algorithm in both encryption and decryption, making it a strong candidate for secure and efficient data storage in cloud environments.

Computation Time for Upload

It refers to the time it takes for a specific piece of data to be uploaded to the cloud. This time can be influenced by several factors, including:

- **File size:** Larger files naturally take longer to upload compared to smaller ones.

- **Network speed:** The upload speed of your internet connection significantly impacts upload times. Faster internet connections lead to quicker uploads.
- **Server load:** If the cloud server is experiencing high traffic or workload, it might take longer to process and store your data, affecting upload times.
- **Encryption:** If the data is encrypted before upload for security purposes, the additional processing involved can contribute to slightly longer upload times.

Therefore, the "computation time for uploading" reflects the combined time taken for transferring the data and any additional processing steps involved during the upload process.

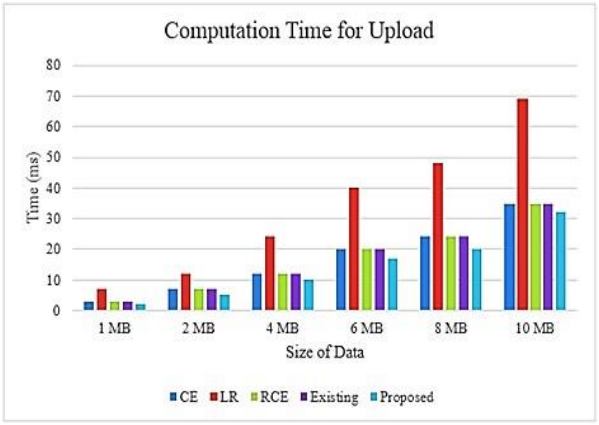

Fig 3: upload computing time

Figure 3 compares the upload computation times of the proposed EECC algorithm with existing algorithms like CE, LR, RCE, and AES for various data sizes. The results demonstrate that EECC exhibits **lower computation times for uploading data to the cloud** compared to the other methods.

Computation Time for Download: The time it takes to retrieve data from the cloud is referred to as the **downloadcomputation time**. Similar to upload times, download times are influenced by factors like:

- **File size:** Larger files naturally take longer to download.
- **Network speed:** Your internet connection's download speed plays a crucial role. Faster download speeds lead to quicker retrieval times.
- **Server load:** High server load can lead to delays in processing download requests, impacting download times.
- **Decryption:** If the data was encrypted before storage, the decryption process adds some time to the download process.

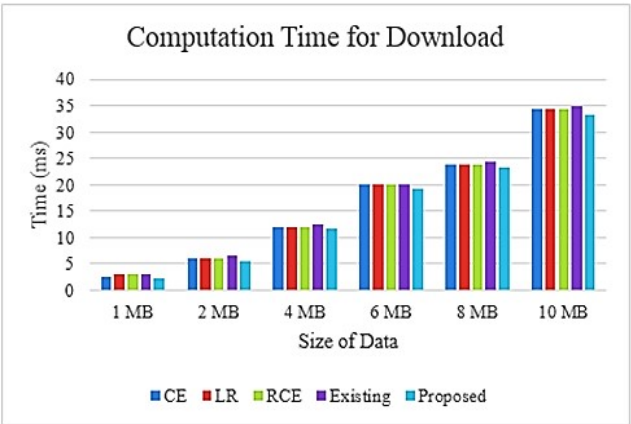

Fig 4: Download computing time comparison

Download Performance:

Figure 4 complements the analysis by comparing the **download computation times** of the proposed EECC algorithm with existing methods like CE, LR, RCE, and AES across various data sizes. The results demonstrate that EECC offers **faster download times** compared to these methods, further highlighting its efficiency.

Key Generation Time:

Figure 5 presents the **key generation time** comparison. EECC is compared to existing methods like MRSAC, MRSA, RSA, and others [40] across various key lengths (100, 128, 256, 512, and 1024 bits).

The analysis reveals that EECC significantly reduces key generation time. For a key length of 1024 bits, EECC takes only 35ms, considerably faster than the 1625ms required by the MRSAC method. Similarly, at a key length of 100 bits, EECC exhibits a faster key generation time of 65ms compared to MRSAC's 153ms.

These findings solidify EECC's efficiency throughout the critical stages of data management in the cloud environment, including uploads, downloads, and key generation. EECC's fast performance across these aspects makes it a compelling choice for secure and efficient cloud storage solutions.

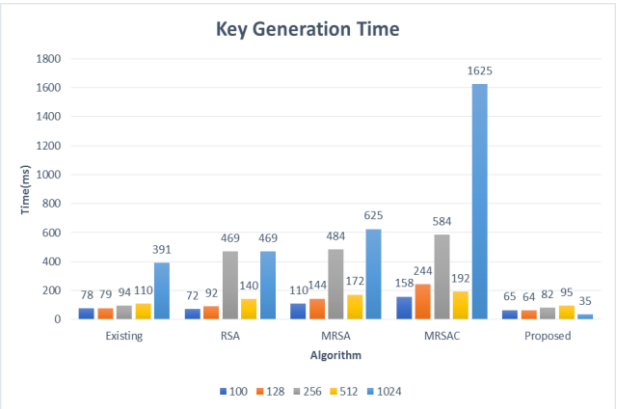

Fig 5: Key generation time a comparison

Cipher Text

A comparative analysis was conducted for encryption performed by the proposed EECC algorithm and various existing algorithms, including AES, Blowfish, and Twofish. The comparison focuses on cipher text size for a standard text file size of 240KB.

Key Observation:

The proposed EECC algorithm generates a **smaller cipher text size (825KB)** compared to the other algorithms:

- AES: 846KB
- Blowfish: 954KB
- Twofish: 954KB

This translates to **reduced storage requirements** for encrypted data using EECC, potentially leading to cost savings and improved storage efficiency in cloud environments.

Importance of Reduced Ciphertext Size:

Smaller ciphertext sizes offer several advantages in cloud storage:

- **Reduced storage costs:** Cloud storage providers typically charge based on storage space used. Smaller ciphertext size implies less storage space required, translating to potential cost savings for users.
- **Faster transmission times:** Smaller files transfer quicker over networks, leading to faster upload and download speeds for encrypted data.
- **Improved server performance:** Reduced data size on the server can contribute to improved server performance and resource utilization.

5. Conclusion

Summary of the Paper on EECC for Secure Cloud Storage

This research paper proposes a novel cryptographic approach, Embellished Elliptic Curve Cryptography (EECC), for enhancing security within a multi-tenant cloud architecture. EECC aims to outperform existing algorithms by addressing critical user needs:

- **Enhanced Security:**EECC leverages the strengths of both Elliptic Curve Cryptography (ECC) and the Diffie-Hellman (DH) algorithm, offering robust data protection while minimizing key exchange risks. Additionally, the system employs alternate key generation, further bolstering its security posture.
- **Improved Performance:** The paper presents a comprehensive performance analysis of EECC across various metrics:
    - **Encryption/Decryption Time:**EECC demonstrates significantly faster encryption and decryption times compared to existing methods, particularly for smaller key lengths.
    - **Upload/Download Efficiency:**EECC reduces computation times for both uploading and downloading data to the cloud, leading to faster data transfer.
    - **Reduced Ciphertext Size:**EECC generates smaller ciphertexts compared to AES, Blowfish, and Twofish, resulting in lower storage requirements and potentially lower cloud storage costs.
- **Key Generation Efficiency:**EECC offers a rapid key generation process, leading to faster system setup and improved overall operational performance.
- **Resource Savings:** Smaller key sizes and reduced data size on the server contribute to efficient resource utilization within the cloud environment.

Key Takeaways:

- EECC presents a compelling solution for secure and efficient data storage in multi-tenant cloud architectures.
- Its faster performance across encryption, decryption, data transfer, and key generation translates to a more responsive and user-friendly cloud experience.
- The reduced storage footprint of EECC's cipher texts can lead to cost savings in cloud storage solutions.

Additional Considerations:

While EECC exhibits promising results, it's essential to acknowledge that the selection of an appropriate encryption algorithm involves a trade-off between various factors, including security requirements, performance considerations, and storage efficiency. Future research might explore ways to further optimize EECC and evaluate its performance in real-world cloud storage scenarios.

References

[1] B. Enthoti, G. Chakrapani and H. P. Sydulu, "DATA ACCESS CONTROL USING COMBINED COMPRESSION AND SECURITY MODEL FOR CLOUD STORAGE," vol. 04, no. 05, May 2021.

[2] C. Manthiramoorthy, K. M. S. Khan and others, "Comparing several encrypted cloud storage platforms," International Journal of Mathematics,

Statistics, and Computer Science, vol. 2, p. 44–62, 2024.

[3] K. Singh, A. Nayyar and A. Garg, "A secure elliptic curve based anonymous authentication and key establishment mechanism for IoT and cloud," Multimedia Tools and Applications, vol. 82, p. 22525–22576, 2023.

[4] R. R. Irshad, Z. Hussain, I. Hussain, S. Hussain, E. Asghar, I. M. Alwayle, K. M. Alalayah, A. Yousif and A. Ali, "Enhancing Cloud-Based Inventory Management: A Hybrid Blockchain Approach With Generative Adversarial Network and Elliptic Curve Diffie Helman Techniques," IEEE Access, vol. 12, p. 25917–25932, 2024.

[5] W. Kaleem, M. Sajid and R. Rajak, "Salp Swarm Algorithm to solve Cryptographic Key Generation problem for Cloud computing," International Journal of Experimental Research and Review, vol. 31, p. 85–97, 2023.

[6] D. Hankerson and A. Menezes, "Elliptic curve cryptography," in Encyclopedia of Cryptography, Security and Privacy, Springer, 2021, p. 1–2.

[7] S. K. Shandilya, B. J. Choi, A. Kumar and S. Upadhyay, "Modified Firefly Optimization Algorithm-Based IDS for Nature-Inspired Cybersecurity," Processes, vol. 11, p. 715, 2023.

[8] M. A. Pisla, M. Stanculescu, R. F. Porumb, B.-A. Enache and G.-C. Seritan, "Microgrid Cyber Security Enhancement Considerations," in 2023 13th International Symposium on Advanced Topics in Electrical Engineering (ATEE), 2023.

[9] M. Miyamoto, K. Teranishi, K. Emura and K. Kogiso, "Cybersecurity-Enhanced Encrypted Control System Using Keyed-Homomorphic Public Key Encryption," IEEE Access, 2023.

[10] C. Lee and S. Lee, "Evaluating the Vulnerability of YOLOv5 to Adversarial Attacks for Enhanced Cybersecurity in MASS," Journal of Marine Science and Engineering, vol. 11, p. 947, 2023.

[11] Jagatheesan, S. ., Parveen, N. ., Mahajan, D. ., Nerkar, S., Singh, G. ., Khan, H. ., & Kumar, M. . (2023). Long Range (LoRa) Communication Protocol with a Novel Scheduling Mechanism to Minimize the Energy in IoT. *International Journal of Intelligent Systems and Applications in Engineering*, *12*(2), 184–193.