Mitigating Data Migration Risks in the Cloud via GA-Optimized Hybrid Cryptography Mechanisms

Anjali Dhaman, Ugrasen Suman
School of Computer Science and IT, DAVV, Indore, India

***Abstract*—Cloud computing has become one of the leading paradigms for bulk data storage and retrieval. However, ensuring data security remains a critical challenge, particularly during transmission when data is most vulnerable. Ensuring data security during transmission remains a critical challenge. The traditional algorithms focus mainly on reducing execution time and using static keys, which can create patterns in the ciphertext that an attacker can exploit. This study introduces an optimized ECC-AES-GA algorithm, which uses the Genetic Algorithm (GA) for generating an optimized parameter for ECC and bulk data is encrypted using the AES-256 algorithm. This algorithm provides security against man-in-the-middle, eavesdropping, replay, brute-force, impersonation, and forward secrecy attacks. The algorithm is also tested with the state-of-the-art algorithms and provides better results in terms of encryption/decryption and security parameters. Experimental analysis are performed based on avalanche tests, entropy level, throughput, execution time, and 10-times-run tests. Furthermore, the algorithm passed all necessary NIST STS tests, which confirms its cryptographic randomness and reliability. It offers security and efficiency, which improves computational overhead, thereby strengthening secure data migration in cloud environments.**

***Keywords*—ECC-AES-GA; cloud data migration; optimization; GA (Genetic Algorithm); ECC (Elliptic Curve Cryptography); AES-256; NIST statistical tests**

I. Introduction

Cloud computing performance, affordability, and accessibility have transformed the digital landscape [1][2]. It entails on-demand access to computer resources, specifically computational power and data storage through the Internet. Cloud data security has become increasingly important as the use of cloud services increases. Effective security measures include comprehensive policies, security solutions with encryption, and other existing methods that play a leading role in securing cloud data [3][4]. Cloud migration is a process that moves programs and data from a local architecture to a scalable environment on the cloud. It allows access to services over the internet. There are various cloud migration practices for corporate businesses, each with unique features and methodologies. The most popular cloud migration models are 'lift and shift' and 'cloud-to-cloud' [5].

Cloud data migration is a complex process in which security, performance, and business continuity frequently collide. It is performed when we want to migrate any application, data, or server to the cloud. It is very crucial to protect data during transit in the cloud. When data is transmitted to the cloud, attackers may compromise sensitive information by stealing cryptographic keys, intercepting session keys, or tampering with the ciphertext. Eavesdropping, man-in-the-middle attacks, and unauthorized access are some of the major problems involved in using the public cloud. These hazards are further exacerbated by the difficulties in keeping encryption keys and the use of ineffective authentication techniques. Despite their importance, such security measures often lack flexibility and scalability. It results in performance degradation, bandwidth issues, low speed, and high latency, which significantly affect the cloud computing environment [6].

The need to meet security criteria has increased in cloud migration [7][8]. Several methods exist to protect data communication in cloud environments. It includes encryption, which is a central aspect of data security, confidentiality, and integrity during data transfer. Encryption, data integrity, and authentication are all aspects of security information management that can be accomplished using cryptography or any other mathematical technique. Cryptography converts the sender's message into a secret message known as ciphertext, and only the intended recipient can decrypt this cipher to get the hidden message. In order to provide security services, it combines numerous mathematical methodologies [3][9]. Recently, encryption has been regarded as the most significant way to protect the privacy of users' data. In many cases, the users are required to share the encrypted secret codes with other parties for various operations on their sensitive data [2].

Several organizations migrating their data to cloud platforms. By 2022, the cloud had already 60% of business data. This value is increasing day by day, as the user saves their data in the cloud. Cloud computing market is expanding by 21.7% in 2023 and will be reaching $597.3 billion from now a day. According to researchers, user expenditures on public cloud services are increasing. It is necessary to protect the sensitive data on the cloud during migration. According to IBM, data breaches affected up to 550 organisations. In 2023, the average price for an information breach was $4.45 million. After a data breach occurs 51% of the organization spends its money on saving data from vulnerable states [10]. Another incident happens, where the global move toward cloud-based services during the Coronavirus epidemic has led to the rise of remote work. It is expected that by 2025, about 95% of digital data and workloads will be stored in the cloud. This rapid growth has also created many security problems, especially when data is migrated from one cloud to another. Most existing solutions depend on centralized third-party auditors (TPAs), which can be costly and may not always be trustworthy [5].

In order to protect data, it is necessary to find any possible solution that saves the data from being compromised. Cryptography is one of the prevention techniques to protect data moving from one cloud to another or vice versa. It is widely acknowledged as a challenge in the research community to provide a secure mechanism for storing or moving data in the

cloud. It is noticed that hybrid cryptography highlights the effective application of symmetric and asymmetric principles. And also, the implementation of lightweight encryption techniques provides low computational resources in cloud computing. The main contributions of this work are the following:

- A hybrid cryptographic architecture combining ECC and AES-256.
- The development of an adaptive parameter selection approach using a genetic algorithm to mitigate static key vulnerabilities.
- Performance validation using NIST SP 800-22 tests demonstrating 99% cryptographic randomness.

This study is organized as follows: Section II shows the existing literature performed on cloud computing security. Section III describes the proposed mechanisms for securing cloud data migration. Section IV provides experimental analysis and performance evaluation, which validates the proposed algorithm results against existing algorithms. Section V concludes the final results and provides future directions.

II. Related Work

In this section, we study several mechanisms for the security of cloud data migration. Various researchers have introduced extensive security frameworks using encryption, decryption, digital signatures, access control algorithms, and secure data storage. As per the cloud security, 90% of the organizations have high to moderate concern about the public security of data [11].

A. Hardware-Oriented and Traditional Approaches

Recent studies have investigated hardware-oriented cloud security solutions that maintain the confidentiality and integrity of data. ECC is implemented on FPGA (Field-Programmable Gate Arrays) and ASIC (Application-Specific Integrated Circuit) platforms with an additive homomorphic encryption technique. This approach uses booth-encoded modular multipliers for 192, 224, and 256-bit key sizes to optimize elliptic curve scalar multiplication. However, this strategy is dependent on static cryptographic parameters and hardware-focused optimization, which restricts its adaptability in the cloud [1]. Similarly, traditional stream ciphers RC4 algorithm, have been modified to a parallel RC4 technique. It uses four concurrent pipelines to improve data encryption and transmission efficiency in the cloud. It improves encryption speed by 3.7% yet remains vulnerable to modern cryptanalysis attacks [6].

B. Domain-Specific Security Frameworks

The rapid expansion of Cyber-Physical Systems (CPS) and the Internet of Medical Things (IoMT) has necessitated specialized security measures. The Crossbreed Blowfish and MD5 (CBM) technique has been developed to strengthen the security of health data within CPS. The technique optimizes data transmission channels by using a wireless sensor network and a fuzzified effective trust-based routing protocol (FET-RP) along with the Butter-Ant Optimization (BAO) algorithm [12]. Furthermore, the SymECCipher is designed as a hybrid encryption framework using ECC for key exchange and AES for data encryption. It outperforms the RSA-2048 algorithm and achieves lower encryption (5ms) and decryption times (4ms) [13]. Additionally, a lightweight Feistel-based substitution-permutation method been developed, which requires a 16-byte (128-bit) block for data cypher in cloud environments [14]. Another approach is introduced which secure the image data using a PRNG called ECSHA512. It combines the cryptographic hash function SHA-512 with an ordered EC, and its dynamically generated isomorphic ECs provide efficiency and security [15]. To solve the problem of data privacy in the cloud, AES-ECC with Merkle Hash Tree (MHT) is used for remote auditing. After that, the user file is encrypted and split into blocks; SHA256 generates tags for each block, resulting in their hash values. The Cloud Service Provider (CSP) then uses these tags to create an MHT and provides proof to the user from the generated hashtags [16].

C. Metaheuristic and Optimization-Based Cryptography

In order to address the limitations of static key management, researchers have chosen evolutionary and identity-based algorithms. Privacy Preservation in the Cloud Sector (PPCS) is a framework that uses Class Topper Optimization (ICTO) algorithm to address the data sanitization and key restoration problem [2]. Moreover, GA have been explored for its ability to create strong cryptographic techniques [3] and also used for the security of an image [17]. To further secure the communication channel, hash-based approaches are used for securely embedding messages into EC points along with the Diffie-Hellman protocol for the encryption process [18]. Beyond optimization, access control mechanisms such as Ciphertext Policy Attribute-Based Encryption (CP-ABE) [9] and Identity-based Elliptic Curve Access Control mechanism (Id-EAC) [10] are widely utilized cryptographic techniques for achieving privacy and precise access control in cloud computing.

D. Research Gap

A comprehensive analysis of existing literature has been performed by researchers on cloud data migration. A detailed comparative analysis is being carried out in Table I. Several encryption algorithms have been compared in terms of time complexity, encryption/decryption time, memory usage, and resistance to cryptanalysis attacks [19][20][21]. Unlike existing AI-based systems that focus on cyber threat detection, such as incident response frameworks [22] [23], rather than on securing data in a cloud environment. Moreover, there are several detection techniques that have been developed, but there is still a lack of prevention techniques. These findings emphasize the need to fill the gaps and improve cryptographic method to ensure robust data security in a dynamic threat environment. Despite this, the performance and security of data transmission remain inadequate due to factors such as high computational overhead, inefficient key management, and latency.

The comparative analysis shows that existing approaches address isolated security requirements and rely on fixed cryptographic configurations. It results in limited adaptability to dynamic cloud environments. ECC-AES based method provide strong confidentiality but they lacks intelligent parameter optimization.

TABLE I. COMPARATIVE ANALYSIS OF EXISTING CLOUD SECURITY APPROACHES

Ref.	Year	Approaches	Security Techniques	Security Objective	Optimization Technique	Assuring Integrity	Access Control	Outcomes	Limitation
[1]	2024	Hardware-Oriented ECC Security	ECC, Additive Homomorphic Encryption, FPGA/ASIC	Confidentiality and Data Protection	Booth Encoded Modular Multiplier	No	No	Faster scalar multiplication and hardware acceleration	Static cryptographic parameters and hardware dependency
[6]	2024	Parallel RC4 Encryption	RC4 Stream Cipher	Data Confidentiality	Parallel Processing (4 Pipelines)	No	No	3.7% improvement in encryption speed	Vulnerable to cryptanalytic attacks
[12]	2025	CBM Framework	Blowfish, MD5, FET-RP	Secure Healthcare Data Transmission	Butter-Ant Optimization (BAO)	Partial	No	Improved routing and transmission efficiency	Application-specific to CPS/IoMT environments
[13]	2024	SymECCipher	ECC + AES	Confidentiality and Secure Key Exchange	None	No	No	Encryption: 5 ms, Decryption: 4 ms	Static key management
[14]	2024	Lightweight Feistel Cipher	Feistel Network, SPN	Lightweight Data Encryption	None	No	No	Reduced computational complexity	Limited scalability and security adaptability
[15]	2024	ECSHA512	SHA-512, ECC, PRNG	Secure Image Encryption	Dynamic Isomorphic EC Generation	No	No	Improved randomness and security	Focused only on image data
[16]	2025	AES-ECC-MHT	AES, ECC, SHA-256, Merkle Hash Tree	Confidentiality and Integrity	None	Yes	No	Secure remote auditing and verification	Additional computational overhead
[2]	2025	PPCS Framework	Data Sanitization, Key Restoration	Privacy Preservation	Improved Class Topper Optimization (ICTO)	Partial	No	Better privacy protection	Limited focus on migration security
[3]	2024	GA-Based Cryptography	Evolutionary Cryptography	Stronger Key Generation	Genetic Algorithm	No	No	Enhanced cryptographic robustness	For securing storage at-rest
[17]	2024	GA-Based Image Security	Genetic Algorithm, Image Encryption	Image Confidentiality	Genetic Algorithm	No	No	Improved image encryption quality	Lacks generalizability to non-visual data
[18]	2024	ECC Message Embedding	ECC, Hash Functions, Diffie-Hellman	Secure Communication	None	Partial	No	Secure message transmission	Lacks migration security
[9]	2024	CP-ABE	Attribute-Based Encryption	Privacy and Fine-Grained Access Control	None	No	Yes	Strong policy-based access control	High computational cost
[10]	2024	Id-EAC	Identity-Based ECC	Authentication and Access Control	None	No	Yes	Lightweight access control	Lacks secure migration process

In order to address these inadequacies, this study proposes an optimized ECC-AES-GA mechanism. Elliptic Curve Cryptography (ECC) is used to overcome limited computational power, memory, and energy resources [24]. ECC uses the algebraic characteristics of elliptic curves over finite fields as a public-key system. The Elliptic Curve Discrete Logarithm Problem's (ECDLP) intractability is the foundation of its security. It offers comparable security with much smaller keys than traditional systems, i.e., RSA [19] [25]. NIST P-256 (secp256r1) is known for providing security for TLS (Transport Layer Security), HTTPS, and digital signatures [26]. It is a public-key encryption that provides safe file sharing and improves security during transmission and storage [27]. In traditional ECC, selecting optimal curve parameters remains a computational challenge in cloud environments, since ECC relies on static parameters [28]. The GA is used to optimize ECC parameters, which eliminate pattern-based vulnerabilities dynamically. When combined with AES-256, it ensures high-speed data migration and high-throughput capabilities.

III. PROPOSED MECHANISM

The optimized ECC-AES-GA algorithm is a hybrid cryptographic framework designed for secure, efficient, and adaptive cloud data migration. It integrates ECC, Advanced Encryption Standard (AES), and a Genetic Algorithm (GA) to overcome the limitations of static security configurations in dynamic cloud environments. The proposed method focuses on optimizing cryptographic parameters while ensuring confidentiality, integrity, authentication, and performance efficiency during data migration.

A. Genetic Algorithm-Based Parameter Optimization

In the proposed optimized ECC-AES-GA algorithm, a GA is employed to optimize the ECC configuration used during cloud data migration.

1) Chromosome encoding: Each chromosome encodes a candidate cryptographic configuration comprising the selected elliptic curve, the key generation mode (ephemeral or static), and a repetition factor to ensure operational stability.

2) Fitness function: The fitness of each individual is evaluated, which balances computational latency, resource consumption, and cryptographic strength. To test real-world performance metrics, a trial ECDH exchange and HKDF derivation are performed on each chromosome.

3) Genetic operators: In order to maintain diversity and avoid local optima, tournament selection is used to identify the best candidates, followed by two-point crossover and index-based mutation.

4) Convergence: The process iterates until a termination criterion (e.g., fitness or maximum generations) is reached, which yields a lightweight ECC parameter.

A GA was implemented with a population of 50 individuals over 50 generations. A crossover probability of 0.8 was utilised for feature recombination, and a mutation rate of 0.2 was applied to maintain genetic diversity and prevent local optima. Parent selection is conducted with a tournament selection type, which maintains a high selection by choosing the fittest individuals from random subgroups. These parameters balance computational efficiency with the exploration-exploitation requirements of the cryptographic search space.

B. Encryption Phase

In the ECC-AES-GA algorithm, the encryption process begins after the optimal ECC configuration is selected by the GA. The sender generates an ephemeral ECC key pair using the GA-optimized elliptic curve to ensure forward secrecy. An Elliptic Curve Diffie–Hellman (ECDH) operation is then performed between the sender's ephemeral private key and

the receiver's public key to compute a shared secret. This shared secret is transformed into a fixed-length symmetric key using the HMAC-based Key Derivation Function (HKDF), ensuring cryptographic randomness and resistance to key-related attacks. The migration data is subsequently encrypted with AES-256-GCM. It provides authenticated encryption and data integrity. The encrypted payload consists of the ephemeral public key, nonce, and ciphertext. It is digitally signed by ECDSA (Elliptic Curve Digital Signature Algorithm), which guarantees sender authentication and protects against data tampering during transmission.

C. Decryption Phase

The decryption process at the receiver begins with verifying the digital signature attached to the encrypted payload. This step ensures the authenticity of the sender and the integrity of the received data before any decryption operation is performed. Upon successful verification, the receiver extracts the sender's ephemeral public key from the payload and recomputes the shared secret using its own ECC private key through ECDH. The same symmetric key is deterministically regenerated using HKDF, which maintains key consistency without explicit key exchange. The ciphertext is then decrypted and authenticated using AES-256-GCM to recover the original data. If either signature verification or authentication fails, the decryption process is immediately terminated. Thereby, enforcing strict security controls prevents unauthorised or corrupted data from being accepted during cloud data migration.

Algorithm 1 shows the GA-based optimization of ECC parameters, where a GA is employed to evolve elliptic curve parameters toward an optimal configuration by evaluating candidate solutions against security strength, computational efficiency, and key robustness, thereby establishing a strong and adaptive cryptographic foundation. Algorithm 2 shows that these optimized parameters by detailing the ECC-based key generation and shared secret establishment, in which communicating entities generate elliptic curve public–private key pairs and derive a common shared secret through an ECDH mechanism, ensuring confidentiality and forward secrecy without transmitting the secret itself. Algorithm 3 shows the ECC-AES-GA encryption and decryption process, where the securely established shared secret is converted into a symmetric session key and used by AES to encrypt and decrypt cloud data.

IV. Experimentation and Analysis

The experimental analysis of the ECC-AES-GA algorithm was conducted using Python on Amazon Web Services (AWS). The data migration is performed on the AWS Storage bucket. The local system perform optimization process, key generation, encryption, and decryption. And securely uploaded to AWS S3 buckets and also retreived from AWS S3 buckets. The entire practical was conducted on AWS for securely data transfer using application which is suitated at the local system.The proposed solution has been assessed and analyze across several file sizes (ex., 1024kb, 1GB etc.) and file types (ex., text file, pdf file, png etc.). The scalability was performed by encrypting and decrypting of different file size data. The security analysis was also performed to find its resilience against data analysis attacks and key guessing. These evaluations serve to underscore the importance and robustness of the proposed solution. The performance measurements were conducted using randomly generated data, created for the purpose of simulating various

Algorithm 1 GA-Based Optimization of ECC Configuration

Input: Population size N_p, Maximum generations G_{max}
Output: Optimized ECC configuration parameter set ψ_{opt}
Initialize population $P_0 = \{C_1, C_2, \ldots, C_N\}$ where each chromosome encodes ECC curve selection and repeat count
Set generation counter $g = 0$
while $g < G_{max}$ **do**
  Evaluate fitness for each chromosome using $f = \text{Latency} + 0.0005(\text{Memory Peak}) - 0.02(\text{Bit Variability})$
  Select individuals using Tournament Selection
  Compute adaptive crossover probability cx_{pb} and mutation probability mut_{pb}
  Apply Two-Point Crossover with probability cx_{pb}
  Apply Adaptive Mutation with probability mut_{pb}
  Preserve top 10% elite individuals
  Inject 5% random immigrants to maintain population diversity
  Update population P_{g+1}
  $g \leftarrow g + 1$
end
return $\psi_{opt} = \arg\min(f(C_i))$

Algorithm 2 ECC Key Generation and Shared Secret Establishment

Input: Optimized ECC parameters ψ_{opt}
Output: Symmetric session key K_{AES}
Sender: Select random ephemeral private key $d_s \in [1, n-1]$ **Sender:** Compute public key $Q_s = d_s \times G$ **Receiver:** Select private key $d_r \in [1, n-1]$ **Receiver:** Compute public key $Q_r = d_r \times G$ Exchange Q_s and Q_r over the cloud channel **Sender:** Compute shared secret $S_{shared} = d_s \times Q_r$ **Receiver:** Compute shared secret $S_{shared} = d_r \times Q_s$ Extract x-coordinate: $Z = S_{shared}(x)$ Derive $K_{AES} = \text{HKDF}(Z, \text{salt}, \text{info})$ **return** K_{AES}

Algorithm 3 ECC-AES-GA Encryption and Decryption

Input: Plaintext M, Symmetric key K_{AES}, Initialization Vector IV
Output: Decrypted Plaintext M'

`// Encryption Phase`
Apply padding to M if required Generate random IV (Nonce) Compute $\{C, Tag\} = \text{AES-256-GCM}_{K_{AES}}(M, IV)$ Generate digital signature $\sigma = \text{ECDSA}_{\text{PrivKey}}(C)$ Construct payload $\mathcal{L} = \{C, IV, Tag, \sigma, Q_s\}$

`// Decryption Phase`
Verify σ using PubKey_s. **if** *fail* **then**
  Abort
end
Re-derive K_{AES} using steps from Algorithm 2 Compute $M' = \text{AES-256-GCM}^{-1}_{K_{AES}}(C, IV, Tag)$ **if** *authentication tag fails* **then**
  Abort
end
return M'

TABLE II. 10 TIMES EXECUTION OF A FILE ON 2.28MB AND TIME MEASURED IN SECONDS (S)

Iteration	Encryption	Decryption	Key Generation
1	0.038	0.029	0.0002
2	0.038	0.029	0.0002
3	0.039	0.029	0.0002
4	0.020	0.015	0.0001
5	0.023	0.018	0.0001
6	0.020	0.014	0.0001
7	0.018	0.014	0.0002
8	0.020	0.014	0.0001
9	0.019	0.015	0.0001
10	0.019	0.014	0.0001
Average	**0.0254**	**0.0191**	**0.00014**

text inputs.

A. Performance Analysis

Various performance metrics are conducted, including avalanche tests, throughput, execution time, memory runtime, and run tests to evaluate the efficiency of the optimized ECC-AES-GA algorithm. The avalanche test demonstrates that even a little modification in the encryption key should produce a different ciphertext. It evaluates the sensitivity of the optimized ECC-AES-GA, which provides an effect of 50.07%. The entropy level is the value of randomness or unpredictability of a key, where a higher entropy level indicates greater randomness. It makes the key more resistant to attacks, where an attacker cannot easily predict the key pattern. The entropy value of an encrypted data is highly secure and it acheives 7.829 bits/byte out of a maximum possible score of 8. This result proves that ECC-AES-GA enhancing confidentiality against entropy-based cryptanalytic attack. Throughput is a measure used to find the data transmission rate in a cryptographic algorithm. The throughput of a 2.28 MB file size is 302.51 MB/s with encryption and decryption times of 0.060 s and 0.041 s. These results provide superior efficiency and optimized resource management. The performance evaluation of optimized ECC-AES-GA encryption and decryption time is consistent. In order to ensure accuracy, a file size of 2.28MB was tested 10 times and the average execution time was recorded. This demonstrates that multiple runs may not affect the time of a particular file and indicate consistent performance. The encryption and decryption time, along with key generation time for a 2.28MB file, is shown in Table II.

B. Comparative Analysis with Existing Algorithms

The optimized ECC-AES-GA algorithm is compared with existing AES-128, Blowfish, Twofish, DES, GA, ElGamal, AES-RSA, and the RSA algorithms. All existing algorithms were implemented and configured under the same experimental environment. Table III shows the encryption and decryption times of existing algorithms, which were performed on 1024, 2048, 4096, and 8192 KB in size. The encryption and decryption time were measured on AWS using python time.perf_counter(). Fig. 1 and Fig. 2 show the encryption and decryption times of different algorithms in seconds, with file size in KB.

The performance of optimized ECC-AES-GA provides more computational efficiency as compared with existing algorithms. Standard algorithms, such as AES-128, Blowfish, and Twofish shows a linear correlation between file size and processing time, whereas DES provides much overhead. Asymmetric methods (RSA and Elgamal) provide higher delays, especially during encryption and decryption time. While the hybrid algorithm (AES-RSA) provides little improvement. But still, it remains slower than symmetric encryption. GA alone is not sufficient for encryption and decryption of data, as it guarantees less security.

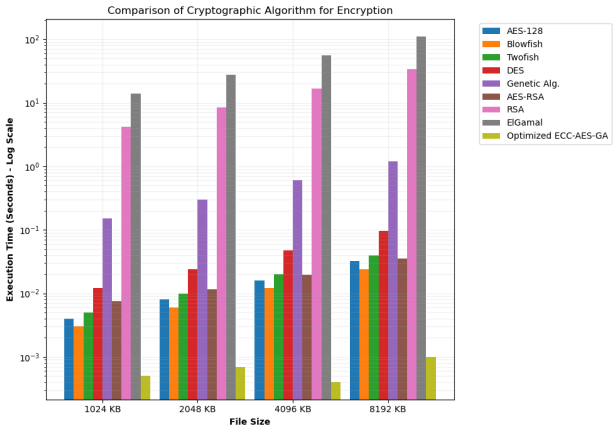

Fig. 1. Standard algorithm compared with optimized ECC-AES-GA for encryption process.

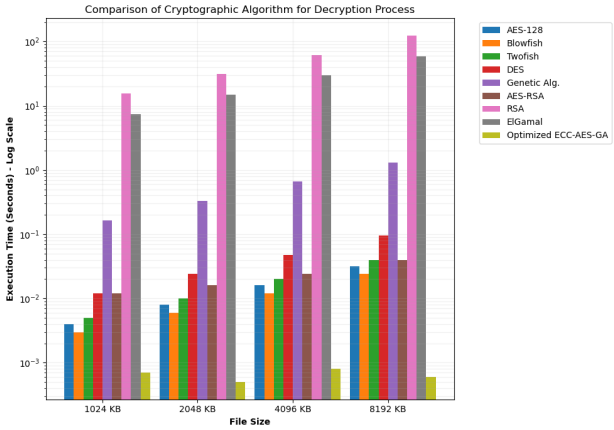

Fig. 2. Standard algorithm compared with optimized ECC-AES-GA for decryption process.

In constrast, the optimized ECC-AES-GA displays the lower encryption/decryption times, which remain nearly constant and independent of data size. The less execution time of the proposed ECC-AES-GA approach are attributed to the use of GA-optimized ECC parameters, which lows computational overhead during the encryption process compared to AES-128 and other existing algorithms. Thus, it achieves better scalability, efficiency, and adaptability for securing cloud data during migration. A prototype application was implemented to validate the practical feasibility of the proposed optimized ECC-AES-GA algorithm. The developed application integrates the complete cryptographic phases, including GA based ECC parameter optimization, secure key exchange, AES encryption,

TABLE III. COMPARISON OF ENCRYPTION AND DECRYPTION TIMES (SECONDS) ACROSS DIFFERENT FILE SIZES (KB)

Algorithm	Operation	1024 KB	2048 KB	4096 KB	8192 KB
AES-128	Enc / Dec	0.004	0.008	0.016	0.032
Blowfish	Enc / Dec	0.003	0.006	0.012	0.024
Twofish	Enc / Dec	0.005	0.010	0.020	0.040
DES	Enc / Dec	0.012	0.024	0.048	0.096
Genetic Alg.	Encryption	0.150	0.300	0.600	1.200
	Decryption	0.165	0.330	0.660	1.320
Elgamal	Encryption	13.80	27.60	55.20	110.40
	Decryption	7.400	14.80	29.60	59.20
AES-RSA	Encryption	0.0075	0.0115	0.0195	0.0355
	Decryption	0.0120	0.0160	0.0240	0.0400
RSA	Encryption	4.200	8.400	16.80	33.60
	Decryption	15.50	31.00	62.00	124.00
ECC-AES-GA	**Encryption**	**0.0005**	**0.0007**	**0.0004**	**0.0010**
	Decryption	**0.0007**	**0.0005**	**0.0008**	**0.0006**

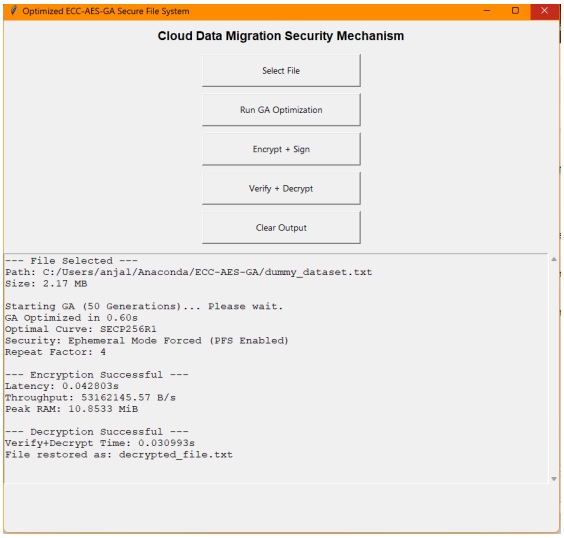

Fig. 3. Optimized ECC-AES-GA: Secure file encryption system.

and ECDSA integrity verification. The application enables users to select files and perform encryption/decryption with signature verification. Fig. 3 shows the application in which encryption and decryption are performed.

C. Complexity Analysis

The overall time complexity of the optimized ECC-AES-GA framework is primarily influenced by the GA during the initialization phase and AES during the data migration phase. The GA-based parameter selection process follows a complexity of $O(G \cdot N_p \cdot f)$, where G is the number of generations, N_p is the population size, and f is the cost of the fitness function evaluation. Notably, this phase occurs during setup and does not impact the real-time transmission speed. The ECDH key exchange process takes $O(k^3)$ or $O(k \cdot \log n)$ depending on the implementation of scalar multiplication, where k is the security parameter (bit-length). However, the overall data encryption using AES is $O(n)$, where n is the number of data blocks. The space complexity is $O(k + M)$, where k represents the key size and M is the size of the message.

D. Statistical Analysis

In order to validate the statistical security of the cryptographic keys generated by the proposed optimized ECC-AES-GA algorithm, the NIST Statistical Test Suite(STS), developed by the National Institute of Standards and Technology(NIST), was employed. This test suite provides a comprehensive framework for evaluating the randomness properties of binary sequences generated by cryptographic mechanisms [1][29][30][31][32]. There are 15 separate statistical tests in the suite. The evaluation incorporated a broad set of statistical tests, including the discrete fourier transform test, frequency test, longest run of ones in a block test, frequency test within a block, runs test, binary rank test, non-overlapping and overlapping template matching tests, Maurer's Universal statistical test, linear complexity test, serial test, cumulative sums test, approximate entropy test, random excursions, and random excursion variant tests. These tests are widely used in many cryptographic applications and are recognized as a suitable instrument for evaluating the unpredictability of binary sequences. For PRNGs to be secure in a cryptographic system, they must pass every statistical test in the NIST suite [15].

The primary objective of using NIST was to verify that the keys generated by the optimized ECC-AES-GA algorithm are sufficiently random to stand against statistical attacks. It ensures reliability and security for cryptographic applications. The binary sequences were generated by concatenating many 128-bit cryptographic keys resulting from ECC-256 operations within the optimized ECC-AES-GA algorithm, facilitating large-scale evaluation using NIST STS. In order to evaluate the randomness of data generated by the optimized ECC-AES-GA encryption algorithm, a file of size 2.16 mb was encrypted, and the resulting ciphertexts were analyzed as binary sequences.

This statistical tool is used to determine the effectiveness of the optimized ECC-AES-GA algorithm, specifically the NIST SP 800-22 test suite. Several independent bit sequences, obtained from one-time session keys and ciphertext, were examined to verify whether they exhibit proper random behaviour or not. The experimental results show that the optimised ECC-AES-GA algorithm successfully passes NIST tests, including Frequency, Block Frequency, Approximate Entropy, and Serial tests, with p-values consistently exceeding the required threshold $\alpha = 0.01$. The frequency (monobit) test yielded a p-value of 0.8414, indicating a balanced distribution of binary

TABLE IV. NIST STATISTICAL TEST RESULTS OF THE OPTIMIZED ECC-AES-GA ALGORITHM.

No.	Statistical Test	P-value	Proportion	Result
1	Frequency (Monobit)	0.841481	1.00	Pass
2	Block Frequency	0.758299	1.00	Pass
3	Runs Test	0.269576	1.00	Pass
4	Longest Run of Ones	0.645750	1.00	Pass
5	Approximate Entropy	0.804677	1.00	Pass
6	Cumulative Sums (Forward)	0.478352	1.00	Pass
7	Cumulative Sums (Reverse)	0.651308	1.00	Pass
8	Discrete Fourier Transform	0.581909	1.00	Pass
9	Binary Matrix Rank	0.168170	1.00	Pass
10	Linear Complexity	0.052453	1.00	Pass
11	Serial Test	$p_1 = 0.4261$ $p_2 = 0.6470$	1.00	Pass
12	Universal Statistical	0.180153	1.00	Pass
13	Overlapping Template	0.021427	1.00	Pass
14	Non-overlapping Template	Majority Passed	≥ 0.98	Pass
15	Random Excursions	Not Evaluated	–	$(J < 500)$
16	Random Excursions Variant	Not Evaluated	–	$(J < 500)$

values in the generated sequence. The block frequency test performed with a block size of 128 bits produced a p-value of 0.758299, confirming uniformity within fixed-length blocks. The entropy test was conducted with a block length of 10 bits, resulting in a p-value of 0.804677. The generated sequences passed all valid and applicable NIST SP 800-22 STS tests but random excursion and its variant tests were excluded due to insufficient number of cycles ($J < 500$) required by the NIST criteria. Table IV shows the result of all necessary NIST STS tests result provided by optimized ECC-AES-GA algorithm. These results indicate that the integration of ECC-based key generation, AES-256 encryption, and genetic optimization does not introduce statistical bias. It also satisfies NIST randomness requirements, which support its applicability to secure cloud data migration.

Based on the experimental evaluation, the proposed optimized ECC-AES-GA algorithm achieves strong cryptographic efficiency and statistical robustness. The measured results of key generation, encryption-decryption times, and achieving security with less computational overhead. Furthermore, the NIST SP 800-22 STS results validate the randomness quality of the keys ensured and successfully passed with p-values.

E. Security Analysis

The proposed optimized ECC-AES-GA algorithm ensures robust security against various cryptographic attacks. The use of ECDH for key exchange ensures the secret key is never transmitted over the communication channel. Authentication mechanisms prevent unauthorized entities from altering key exchange, which protects the data from a man-in-the-middle. Timestamps in each communication session prevent an attacker from reusing it. It provides each session a unique identity, which resists replay attacks. Data confidentiality is achieved through the AES-256-GCM algorithm. If an attacker is intercepting the communication, they are not able to understand the data over the channel due to the use of a secret key (by using the ECC method). It also provides integrity to the data and resists against known-plaintext and chosen-plaintext attacks. The use of optimized parameter of ECC using GA makes brute force attacks computationally infeasible. The digital signature (ECDSA) ensures mutual authentication between two entities. So an adversary cannot impersonate a legitimate user in the system, which resists an impersonation attack. The session keys are generated using HKDF. If one session key is disclosed, it will not affect other session keys. The use of ephemeral keys (using ECDH) ensures forward secrecy. Even if the long-term private keys are compromised, session keys are not revealed.

The suggested ECC-AES-GA mechanism provides robust security and efficient performance for cloud data migration. Existing ECC-AES approach is basically used fixed cryptographic parameters and key exchange process. The static configurations lacks optimal security performance. Unlike existing approaches focus on only secure storage, access control and lacks other key components for securing cloud data migration. This ECC-AES-GA provides end-to-end security of data migration along with key generation, key exchange, data encryption, transmission and integrity verification.

The current method has been assessed in a homogeneous cloud setup, but its performance in multi-cloud or heterogeneous cloud setups has yet to be explored. Such setups may provide issues in terms of interoperability, key management, and synchronization across distributed platforms. It assumes that endpoints are trusted and does not explicitly cover all dynamic or adaptive attack scenarios. Despite these limitations, the suggested technique is still well-suited for protecting large-scale data transmission where security and data integrity take place over challenging real-time requirements.

V. CONCLUSION

The development of the optimized ECC-AES-GA algorithm represents a significant advancement in securing cloud data migration without compromising security issues. The approach enhances key generation, secure exchange, and encryption robustness while addressing limitations of traditional key management systems. The performance evaluation demonstrates an effective balance between security and efficiency in terms of latency, throughput, and memory usage. And also the security validation through the NIST statistical tool, which confirms high randomness and strong resistance to cryptographic attacks. Future work will focus on providing security against emerging quantum computing threats.

ACKNOWLEDGEMENT

This research was supported by the University Grants Commission (UGC)-NET India through the Senior Research Fellowship (SRF).

REFERENCES

[1] R. Parthasarathy and P. Saravanan, "Side-channel attack resilient implementation of homomorphic encryption using elliptic curve cryptography for secure cloud computing," *Integration*, vol. 104, p. 102439, 2025.

[2] S. Mittal, "Fully homomorphic encryption-based optimal key encryption for privacy preservation in the cloud sector," *Journal of Information Security and Applications*, vol. 91, p. 104048, 2025.

[3] O. Can, F. Thabit, A. O. Aljahdali, S. Al-Homdy, and H. A. Alkhzaimi, "A comprehensive literature of genetics cryptographic algorithms for data security in cloud computing," *Cybernetics and Systems*, vol. 56, no. 5, pp. 413–447, 2025.

[4] M. A. Qasem, F. Thabit, O. Can, E. Naji, H. A. Alkhzaimi, P. R. Patil, and S. Thorat, "Cryptography algorithms for improving the security of cloud-based internet of things," *Security and privacy*, vol. 7, no. 4, p. e378, 2024.

[5] M. Aruna, M. K. Hasan, S. Islam, K. Mohan, P. Sharan, and R. Hassan, "Cloud to cloud data migration using self sovereign identity for 5g and beyond," *Cluster computing*, vol. 25, no. 4, pp. 2317–2331, 2022.

[6] A. Ullah, T. Alam, F. S. Khan, and H. Aznaoui, "Parallel implementation of rc4 data encryption method for cloud computing," *Computing Open*, vol. 1, p. 2350002, 2023.

[7] Y. K. Saheed, O. H. Abdulganiyu, and T. Ait Tchakoucht, "Modified genetic algorithm and fine-tuned long short-term memory network for intrusion detection in the internet of things networks with edge capabilities," *Applied Soft Computing*, vol. 155, p. 111434, 2024.

[8] C. Uikey and D. Bhilare, "Trustrbac: Trust role based access control model in multi-domain cloud environments," in *2017 International Conference on Information, Communication, Instrumentation and Control (ICICIC)*. IEEE, 2017, pp. 1–7.

[9] G. Sravya, P. S. Kumar, and R. Padmavathy, "Survey of post-quantum lattice-based ciphertext-policy attribute-based encryption schemes for cloud storage: Taxonomy, open issues, and future directions," *IEEE Transactions on Services Computing*, vol. 17, no. 6, pp. 4540–4557, 2024.

[10] B. Prabhu Kavin, S. Ganapathy, U. Kanimozhi, and A. Kannan, "An enhanced security framework for secured data storage and communications in cloud using ecc, access control and ldsa," *Wireless Personal Communications*, vol. 115, no. 2, pp. 1107–1135, 2020.

[11] R. Doshi and V. Kute, "A review paper on security concerns in cloud computing and proposed security models," in *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*. IEEE, 2020, pp. 1–4.

[12] R. Priyadarshini, A. Quadir Md, N. Rajendran, V. Neelanarayanan, and H. Sabireen, "An enhanced encryption-based security framework in the cps cloud," *Journal of Cloud Computing*, vol. 11, no. 1, p. 64, 2022.

[13] P. Selvi and S. Sakthivel, "A hybrid ecc-aes encryption framework for secure and efficient cloud-based data protection," *Scientific Reports*, vol. 15, no. 1, p. 30867, 2025.

[14] F. Thabit, S. Alhomdy, A. H. Al-Ahdal, and S. Jagtap, "A new lightweight cryptographic algorithm for enhancing data security in cloud computing," *Global Transitions Proceedings*, vol. 2, no. 1, pp. 91–99, 2021.

[15] F. Tellez and J. Ortiz, "Comparing ai algorithms for optimizing elliptic curve cryptography parameters in e-commerce integrations: A pre-quantum analysis," *arXiv preprint arXiv:2310.06752*, 2023.

[16] S. Gangadharaiah and P. Shrinivasacharya, "Secure and efficient public auditing system of user data using hybrid aes-ecc crypto system with merkle hash tree in blockchain," *Multimedia Tools and Applications*, vol. 83, no. 29, pp. 72 301–72 320, 2024.

[17] K.-W. Wong, W.-S. Yap, D. C.-K. Wong, R. C.-W. Phan, and B.-M. Goi, "Cryptanalysis of genetic algorithm-based encryption scheme," *Multimedia Tools and Applications*, vol. 79, no. 35, pp. 25 259–25 276, 2020.

[18] Y. Lahraoui, S. Lazaar, Y. Amal, and A. Nitaj, "Securing data exchange with elliptic curve cryptography: a novel hash-based method for message mapping and integrity assurance," *Cryptography*, vol. 8, no. 2, p. 23, 2024.

[19] D. Ramakrishna and M. A. Shaik, "A comprehensive analysis of cryptographic algorithms: Evaluating security, efficiency, and future challenges," *Ieee Access*, 2024.

[20] A. Dhaman and U. Suman, "A survey on security techniques for cloud data migration," in *2024 International Conference on Advances in Computing Research on Science Engineering and Technology (ACROSET)*. IEEE, 2024, pp. 1–5.

[21] U. Suman and A. Dhaman, "Towards a comparative analysis of security attacks in cloud data migration," in *2025 World Skills Conference on Universal Data Analytics and Sciences (WorldSUAS)*. IEEE, 2025, pp. 1–7.

[22] S. Mouhim and F. Lachhab, "Towards a context awareness system using iot, ai, and big data technologies," *IEEE Access*, 2025.

[23] M. A. M. Farzaan, M. C. Ghanem, A. El-Hajjar, and D. N. Ratnayake, "Ai-enabled system for efficient and effective cyber incident detection and response in cloud environments," *arXiv preprint arXiv:2404.05602*, 2024.

[24] S. Ullah, J. Zheng, N. Din, M. T. Hussain, F. Ullah, and M. Yousaf, "Elliptic curve cryptography; applications, challenges, recent advances, and future trends: A comprehensive survey," *Computer Science Review*, vol. 47, p. 100530, 2023.

[25] P. Velmurugadass, S. Dhanasekaran, S. S. Anand, and V. Vasudevan, "Enhancing blockchain security in cloud computing with iot environment using ecies and cryptography hash algorithm," *Materials Today: Proceedings*, vol. 37, pp. 2653–2659, 2021.

[26] V. Tanksale, "Efficient elliptic curve diffie–hellman key exchange for resource-constrained iot devices," *Electronics*, vol. 13, no. 18, p. 3631, 2024.

[27] M. Y. Shakor, M. I. Khaleel, M. Safran, S. Alfarhood, and M. Zhu, "Dynamic aes encryption and blockchain key management: a novel solution for cloud data security," *IEEE Access*, vol. 12, pp. 26 334–26 343, 2024.

[28] J. Guruprakash and S. Koppu, "Ec-elgamal and genetic algorithm-based enhancement for lightweight scalable blockchain in iot domain," *IEEE Access*, vol. 8, pp. 141 269–141 281, 2020.

[29] K. Sasikumar and S. Nagarajan, "Comprehensive review and analysis of cryptography techniques in cloud computing," *IEEE Access*, vol. 12, pp. 52 325–52 351, 2024.

[30] N. B. Bakhtiari, M. Rafighi, and R. Ahsan, "A trust management system for fog computing using improved genetic algorithm," *The Journal of Supercomputing*, vol. 80, no. 14, pp. 20 923–20 955, 2024.

[31] S. Potluri, "Multi-layered security policy enforcement for confidential data in serverless cloud functions," *International Journal of Emerging Trends in Computer Science and Information Technology*, vol. 6, no. 1, pp. 124–134, 2025.

[32] S. K. Kotha, M. S. Rani, B. Subedi, A. Chunduru, A. Karrothu, B. Neupane, and V. Sathishkumar, "A comprehensive review on secure data sharing in cloud environment," *Wireless Personal Communications*, vol. 127, no. 3, pp. 2161–2188, 2022.