Journal of Discrete Mathematical Sciences & Cryptography
ISSN 0972-0529 (Print), ISSN 2169-0065 (Online)
Vol. 29 (2026), No. 6, pp. 2553–2587
DOI : 10.47974/JDMSC-2823

A deep learning and blockchain-based security framework for cloud data protection

N. Ajay [‡]
Department of Computer Science and Engineering
SJC Institute of Technology
Affiliated to Visvesvaraya Technological University Belagavi
Chickballapur 562101
Karnataka
India

H. S. Mohan [†]
Department of Computer Science and Engineering (Data Science)
RNS Institute of Technology
Affiliated to Visvesvaraya Technological University Belagavi
Bengaluru 560098
Karnataka
India

I. S. Rajesh [§]
Department of Artificial Intelligence and Machine Learning
BMS Institute of Technology and Management
Bengaluru 560119
Karnataka
India

[‡] *ORCID-ID:* 0000-0001-6373-419X
[†] *ORCID-ID:* 000-0001-6162-8481
[§] *ORCID-ID:* 0000-0002-7736-7156

[‡] *E-mail:* n.ajay2@gmail.com
[†] *E-mail:* mohan_kit@yahoo.co.in
[§] *E-mail:* rajeshaiml@bmsit.in

Sumit Gupta *
Department of Research and Development
DeepCognix AI Labs
Bangalore 560058
Karnataka
India

Abstract

Cloud computing provides scalability, cost efficiency and flexibility of numerous applications. Nevertheless, there remain security issues because of data integrity, confidentiality and cyber threats. Cryptographic measures such as encryption and hashing are secure, but do not have the dynamic threat assessment capabilities. This paper introduces a security platform, which combines deep learning with cryptographic hashing and blockchain technology to enrich key exchanges, data verification, and detection of tampering in cloud storage systems. The suggested key exchanged model builds network based key agreement by modifying Diffie-Hellman approach. It uses artificial noise models of neural networks to ensure high security in the key parameters against nefarious activities. The system adopts a CNN-based encryption scheme as well as a decryption system that had been specially developed to offer a flexible encryption security to cloud-based data. The blockchain technology is implemented as a part of the model because it introduces cryptographic hash verification of the stored data to the ledger system that cannot be changed further, as once it is verified it cannot be altered by an unauthorized party. The set of combined Merkle trees results in efficient efficiency improvements on verification processes by reducing the number of computations needed with preserving fast scalable integrity verification algorithms. The proposed model is also tested by performance standard and security analysis in order to determine its efficacy. Findings indicate that a 256-bit key provides the best PSNR of 30.9778 dB and SSIM of 0.8741 at 32x 32 resolution, and ensures the highest visual similarity and structural similarity after decryption.

Subject Classification: *94A60, 94A62, 68T07, 68M14, 68P25.*

Keywords: *Cloud computing, Cyber security, Integrity, Privacy, Deep learning, Blockchain.*

1. Introduction

During last few decades, the communication systems have grown drastically due to excessive surge in internet. The multimedia communication systems are widely adopted in various applications such as medical, satellite communication, video conferencing etc. [1] The amount of data generated by these devices is very huge therefore storing and processing this data becomes challenging issue. Computation technology advancements during recent years have made cloud computing highly popular because it provides Internet-based

* *ORCID-ID:* **0009-0007-4766-3468**

* *E-mail:* **sumit@deepcognix.com (Corresponding Author)**

access to applications along with services and storage capabilities and computing resources [2]. Cloud computing has widespread applications across medical science and agriculture together with business and information technology domains. The system provides resources that can be adopted flexibly while offering affordable independent services through its framework [3].

More organizations choose cloud computing due to benefits including scalability, reliability and flexibility together with its cost-efficient features. Cloud computing represents an evolving standard which permits shared network-access to adaptable resources through computer network systems. Users access virtual information technology (virtual IT) through the Amazon Elastic Compute Cloud (Amazon EC2) because it enables them to rent virtual machines for application execution. The computing capacity of Amazon EC2 exists under the Amazon Web Services (AWS) Cloud environment [4], [5]. The range of cloud computing services includes Google App Engine for application hosting alongside Google Apps and Microsoft Office Online as SaaS software solutions in addition to Apple iCloud as a network storage option and DigitalOcean as both an IaaS and PaaS provider [6].

The rapid expansion of cloud solutions alongside distributed information storage platforms necessitates immediate data protection against unauthorized modification as well as tampering and unauthorized access. Users at both individual and organizational levels depend extensively on cloud-based data storage systems to protect their protected information records and proprietary content. Traditional security methods including access control and encryption offer restricted protection against data transformation attacks and unauthorized changes and internal violations. Research and engineering fields have shown major interest in cloud storage systems because of their beneficial aspects which combine vast scalability with easy usability and flexible operations together with platform compatibility [7]. A fundamental element of cloud computing is cloud storage which provides superior features than local storage thus driving massive data movements to cloud-based systems by individuals alongside businesses. Cloud data owners hand their information to external cloud storage providers where information integrity stands as a primary security priority. Researchers focus on developing new algorithms because they aim to protect cloud-stored data integrity. The developed algorithms effectively uphold correct outsourced data characteristics and strengthen overall system protection. Cloud storage services exist for data owners but they show restraint toward CSPs because they doubt cloud reliability and worry about information sharing between users. Cloud computing systems that combine infrastructure features with storage capabilities mirror the operational model of Infrastructure as a Service (IaaS). The usage of IaaS platforms leaves users uncertain regarding both their data storage areas and the systems responsible for system management operations. Lack of transparency intensifies privacy concerns together with security issues especially when malicious users exist because it jeopardizes both data confidentiality and

integrity. Therefore, the acceptance and success of cloud computing models highly rely on achieving the desired security levels. Some of the major security challenges faced by cloud systems are illustrated in Figure 1.

Figure 1
Security challenges in cloud computing

A wide range of methods have been established for improving cloud computing system security since widespread adoption of cryptographic measures. The design and analysis of security protocols that defend private information from unauthorized viewers constitutes cryptography [8]. This transformation process known as cryptocurrency or cipher operation takes readable data and converts it into secure encryption formats through algorithmic methods. There are two main categories of algorithms with respect to key usage: symmetric key algorithms and asymmetric key algorithms [9]. Symmetric encryption represents an early encryption technique which requires one single key to perform encryption along with decryption operations. Symmetric encryption algorithms include DES, RC2, RC4, Blowfish, RC5, RC6 and AES among their examples [10]. The shared-key encryption system between sender and receiver exposes vulnerabilities because any interceptor of their key exchange could uncover the data through unauthorized access. The process requires a secure channel to transmit keys since it represents a major vulnerability point. In asymmetric encryption two separate keys work together where the public key handles encryption and the private key serve for decryption. The encryption methods DSA, RSA, and ECC apply this security scheme. A reachable public key allows anybody to encrypt messages but only the protected private key serves decryption purposes exclusively. Implementation of two keys strengthens security procedures through eliminating dependence on shared secrets thus minimizing transmission interception threats.

Several methods have been discussed to address the data integrity in cloud storage systems such as cryptographic hashing, third-party auditing, and blockchain-based methods. Hash functions such as SHA-256 and SHA-3 are widely used to generate unique fingerprints of stored data [11]. While

effective for detecting modifications, hashing alone does not prevent malicious alterations-an adversary can replace both the data and its corresponding hash. Hash verification requires frequent re-computation, leading to high computational costs in large-scale cloud systems. On the same note, Cloud service providers usually trust the services of independent auditors to verify the integrity of data stored. Although this approach removes verification on the end-user, it creates trust dependencies and the user must have full trust in the third-party verifiers. TPAs themselves are prone to failure as well as being hacked and can represent point of failure or cause security risk and privacy concerns. Existing studies have directed their attention to blockchain based technology that provides the storage of integrity proofs in a decentralized and immutable manner to minimize the chance of tampering. The Merkle Trees are some of the methods of enhancing the efficiency of verification by organizing hash values in a hierarchy. Nonetheless, blockchain has a heavy storage overhead and there is a scaling and latency problem in public blockchains because of the time spent in processing transactions in blockchain. These constraints emphasize the necessity of a smart, flexible, and scalable solution that can help improve data integrity checking and minimize the level of computation and storage.

The traditional standard methods such as AES, RSA follow and operate on mathematical transformation and provide efficient solution but these methods face challenges in dynamic adaptability, pattern recognition and adversarial learning based attacks. The proposed approach is customized for encryption and decryption of the image-based data where spatial features play a significant role. The CNNs extract these spatial attributes efficiently which helps the encryption approach to secure visual data formats. The model creates non-linear, key-dependent feature transformations that make brute-force or pattern-based attacks highly difficult.

Moreover, the proposed model uses neural noise model for both key generation and exchange whereas traditional systems typically depend on random number generators, which can be vulnerable. The neural noise introduces stochastic behavior, boosting entropy, and defending against deterministic attacks.

Therefore, this research mainly focus on development of advanced security mechanism for cloud systems therefore we have developed a novel approach to address these issues. This research presents an innovative hybrid approach that combines deep learning, blockchain technology, and cryptographic hashing to enhance data integrity, key exchange security, and tamper resistance in cloud computing environments. The key contributions of this work are as follows:

- We propose a new deep learning-based key exchange protocol based on the Diffie-Hellman protocol, which also employs the neural noise model.

The utilization of neural networks to learn together helps in the creation of a safe and effective key agreement between cloud participants, which reduces the chances of man-in-the-middle attacks.

- We introduced a user-specific CNN model to encrypt data and images in the cloud. The CNN-based method does not require conventional encryption methods, and unlike its competitors, the method can also learn complex reformations, adapt and empowers encryption, and is still computationally efficient in processing large-scale cloud applications.
- To prevent the tampering of data storage, we combined blockchain technology that provides commit-free logging of integrity demonstrations. All cloud storage operations are engraved into a blockchain, which offers transparency, decentralization, and resistance to unauthorized alterations.
- The proposed framework uses hash functions (SHA-256) to ensure the creation of distinctive cryptographic fingerprints on data stored. This efficacy of verification is improved by the usage of Merkle Trees, and the complexity of computing is lowered.

The rest of the article is structured according to the following section: section II is the overview of the existing methods, section III is the proposed hybrid solution to the problem of cloud security, section IV is the result of the suggested approach, and the last section V is the concluding remarks of the research work.

2. Literature Review

In this section, the research briefly provides the literature review regarding current approaches to safe protocols in the sphere of cloud computing systems. The developments in distributed computing and virtualization have put cloud computing as one of the top choices in terms of data management and storage. Nevertheless, the data security and privacy rules, as well as authentication processes, are still the issues that prevent its extensive use. Santos, Ghita, and Masala [12] developed a data fragmentation system together with NoSQL database storage to achieve public cloud data security against unauthorized entry. The multimodal biometric authentication system was built to verify user identities and achieved lower latency metrics than encryption protocols while serving healthcare IT frameworks that demand low delay times. The authors of [13] introduced a two-stage optimization method which helps IaaS providers manage their cloud service costs. IaaS providers begin by selecting cheap security services under time and security requirements. Workflow scheduling during the second stage of the approach implements an enhanced firefly algorithm to achieve optimized cost reduction among cloud service resources without sacrificing performance or security levels. The solution minimizes costs better than previous

approaches and satisfies user-experienced demands. The proposed solution by Shakor et al. [14] introduces AES with volatile encryption keys as a dynamic approach along with key management elements. The secure management of keys makes use of blockchain technology to protect against risks connected to centralized storage systems. ECC enables secure cloud data transmission and storage through its deployment which provides both scalable and decentralized security solutions.

The research conducted by Liu et al. [15] examines zero-trust model based domain data sharing for cloud-edge-end systems. This method presents plaintext-checkable encryption systems for lightweight IoT devices while using a blockchain-based multi-domain sharded architecture together with fairness and scalability provisions in their cross-domain data-sharing framework. Security analysis with performance evaluations revealed that their method successfully supports both high performance data sharing and secure operations. The research by Li et al. [16] combats cloud environment data loss through their creation of multi-cloud storage systems with built-in public verification protocols. A PVPKEET-based method established by their team allows users to check the integrity of cloud server-stored encrypted data without having to decrypt it. The proposed security solution fulfils two essential requirements which protect against plaintext attacks but remain practically applicable because of its verified encryption feature. The cryptographic primitive Re-PAEKS serves as a solution to this challenge because it combines PRE with PAEKS according to Luo, Wang, and Yan [17]. This security method defends quantum computers from attacks alongside keyword guessing attacks (KGAs) with minimum end-to-end delays. The proposed scheme implements lattice-based identity-based encryption (IBE) to provide secure access delegation and reliable encryption.

Wang et al. [18] developed SESR which represents a secure and efficient similarity retrieval system for cloud computing through homomorphic encryption techniques. This approach implements Hamming distance measurement for similarity assessment together with a BK-KD tree data structure for building fast retrieval systems that manage access control. The scheme achieves privacy protection and data integrity as well as decreased storage requirements through Simhash-based feature extraction on a cooperative model between two cloud servers and message authentication methods. Shrivastava, Alam, and Alam [19] have implemented blockchain technology as a reinforcement measure to stop unauthorized cloud system access attempts. The cryptographic solution unites ECC technology with Elgamal encryption through FSO optimization to select ideal encryption keys. The blockchain controller protects data integrity through Proof-of-Authority (PoA) validations and SHA-256 hash generation processes.

Song et al. [20] resolve privacy issues in external data hosting by creating encryption methods based on decimals which enhance performance by removing limiting constraints from bit-length evaluations. A k-means clustering algorithm with privacy protection from the authors supports efficient query processing

through parallelization using threads. Security analysis demonstrates the scheme's accomplishment in maintaining data confidentiality and query protection. Zubair and Ahmed [21] established a hybrid cryptographic model which protects cloud system data storage and distribution while maintaining security measures. The scheme combines public along with private and secret keys to enable secure encryption and decryption as well as authentication systems that stop unauthorized access. Data transmission operations run efficiently using this protocol while the system maintains its security parameters. Akbar et al. [22] expand cloud security research through their work on developing the Cyber-Security Trust Model which unites Quantum Key Distribution (QKD) with Modified Advanced Encryption Standard (MAES). Blockchain technology creates tamper-resistant ledgers which enhance the data integrity protection system. The proposed encryption and decryption system enhanced with Merkle tree verification maintains 99.84% accuracy to protect against contemporary cyberattacks effectively. The research by Inam et al. [23] develops BCAES which stands for Blockchain-based Chaotic Arnold's Cat Map Encryption Scheme because it provides secure medical image protection during storage in cloud systems. Before transmission they encrypt the images while signed document hashes for authenticity and data integrity remain stored on the blockchain. This method undergoes multiple analytical tests to prove its security properties. The research by Rajan et al. [24] examines the essential demand for medical image retrieval without data loss in cloud storage systems. The proposed system utilizes chaotic map cryptography along with hash creation methods to implement ConvNeXt network deep learning hashing for achieving secure image search efficiency. Users retain privacy throughout protected retrievals by accessing medical images encrypted in index files through authenticating verification methods.

Hybrid metaheuristic algorithms integration has demonstrated its potential and achieved positive outcome in the optimization of complex scheduling challenges within a cloud setting. One such example is the SA-PSO-double Q-learning based method that has been offered to dynamically balance cloud load balancing, which aids to integrate simulated annealing, particle swarm optimization, and deep reinforcement learning techniques [25]. Additionally, the automatic identification of fovea in retinal fundus images has been advanced through various techniques, providing valuable insights for medical engineering applications [26]. Furthermore, in a recent research work by Anitha [27], an optimized VM allocation strategy is presented which uses threshold-based migration strategy to identify the overloaded VMs and do the migration. It is an extension of the work proposed by Siddesha, Jayaramaiah, and Singh [28], that introduced a deep reinforcement learning scheme for task scheduling in cloud computing, which optimizes task distribution and performs resource management. Recent works such as hybrid of biogeography-based optimization with genetic algorithm [29] and congestion-aware collaborative Dijkstra optimization approach guided by the Proximal Policy Optimization (PPO) method [30] has shown significant improvements in cloud workload scheduling and optimization field.

3. Proposed Model

This section presents the proposed hybrid approach to secure the data in cloud computing environments. The main aim of this approach to provide security to the data being stored or retrieved from the cloud system.

The proposed model introduces neural network based key exchange model, Diffie Hellman based neural noise model for efficient key exchange, a custom CNN model for data encryption and decryption. Furthermore, an integrated approach using blockchain and Hashing mechanism to enhance the security by ensuring the integrity check. Below subsection presents the detailed overview of proposed approach.

3.1 *Overview*

As discussed before, the secure key exchange protocol plays important role to protect the data transmitted between user and cloud servers. Literature review has discussed several cryptography schemes such as RSA, Diffie-Hellman but these approaches are vulnerable to quantum attacks and suffer from computational inefficiencies. In this work, a NN based key generation protocol is presented which is used to generate the cryptographic key dynamically. The main contribution of NN is that it adapts the traffic patterns and security threats to generate the secure and unpredictable keys. This method enhances resistance against attacks such as man-in-the-middle (MITM) and brute-force attacks while optimizing performance for cloud-based applications.

Next step performs the key exchange where generated keys are exchanged to perform the encryption and decryption tasks. In this work, we have used Diffie-Hellman key exchange protocol with Neural Noise. This approach integrates the DL based noise generator model to introduce the controlled randomness into the key generation process. This neural noise process helps in preventing the adversaries from accurately predicting exchanged keys, reducing the likelihood of MITM attacks in cloud environments.

Later, data encryption methods are applied to secure the data stored in the cloud servers. Several cryptography methods have been introduced but computational complexity is a major challenge in this domain. Therefore, the proposed approach introduced a custom CNN model to encrypt and decrypt the data to enhance the cloud security. The CNN model learns complex mappings between input and cipher, making the encryption process more resistant to traditional cryptanalysis. Moreover, it enables efficient and high-speed encryption, reducing computational overhead while ensuring robust security for data at rest and in transit. Unlike the traditional encryption methods, the CNN based model can adapt over time, dynamically adjusting encryption patterns based on evolving security threats in cloud environments.

Finally, the cloud storage systems require mechanisms to verify data integrity and prevent unauthorized modifications. The proposed model integrates the use of blockchain technology and cryptographic hashing to increase the level

of trust and protection of data in the cloud. All users of the cloud record their data transactions or modifications in a blockchain ledger, which is linked to an immutable yet transparent history of any change. Hashing algorithm (SHA-256) is used to generate fingerprints of stored data, and in case of unlicensed alterations, they can be instantly identified. The model provides tamper-proof integrity check by utilizing blockchain which makes data stored in clouds more resistant to cyber threats. Table 1 describes all the notations used in the modelling of proposed algorithms. The proposed architecture for secured access and data encryption mechanism is illustrated in Figure 2.

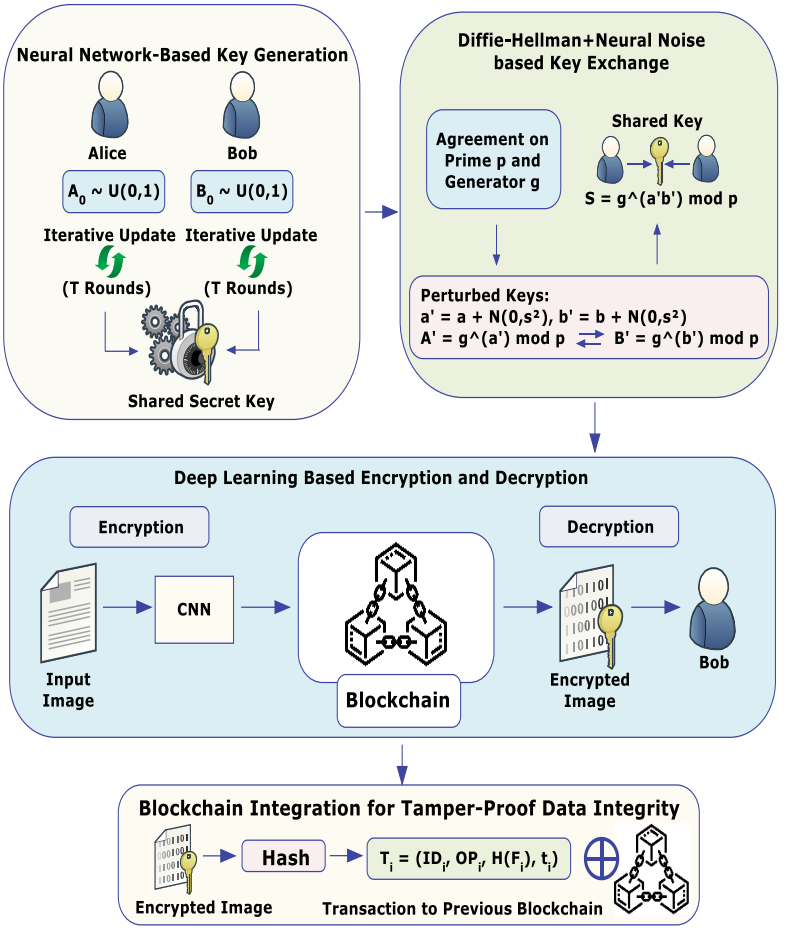


Figure 2
Proposed system architecture

Below Table 1 presents the notations used in this article for mathematical modelling and various computational analysis.

Table 1

Table of notations

Notation	Description
$\boldsymbol{A_0, B_0}$	Initial random weight vectors for Alice and Bob, respectively
$\boldsymbol{f, T}$	NN transformation function, No. of training rounds
$\boldsymbol{\mathcal{N}(\mu, \sigma^2)}$	Gaussian noise with mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma^2}$
$\boldsymbol{A_t, B_t}$	Weight vectors at iteration $\boldsymbol{t}$
$\boldsymbol{K}$	Final shared cryptographic key after training
$\boldsymbol{a', b'}$	Perturbed private keys using neural noise in Diffie-Hellman
$\boldsymbol{A', B'}$	Public keys computed from perturbed private keys
$\boldsymbol{S'_A, S'_B}$	Shared secret keys derived by Alice and Bob
$\boldsymbol{p, g}$	Agreed prime number, generator in DH exchange
$\boldsymbol{I}$	Input grayscale image $\boldsymbol{I} \in \mathbb{R}^{\boldsymbol{H \times W}}$
$\boldsymbol{K}\ (image)$	Secret key vector $\boldsymbol{K} \in \mathbb{R}^{\boldsymbol{N}}$
$\boldsymbol{X}$	Feature map after convolution: $\boldsymbol{X} \in \mathbb{R}^{\boldsymbol{H \times W \times C}}$
$\boldsymbol{F}$	Flattened feature map: $\boldsymbol{F} \in \mathbb{R}^{\boldsymbol{HWC}}$
$\boldsymbol{Z}$	Concatenated image features and key
$\boldsymbol{E}$	Encrypted vector output before reshaping
$\boldsymbol{I_{enc}}$, $\boldsymbol{I_{dec}}$	Encrypted image, Decrypted image
$\boldsymbol{X', F', Z', D}$	Intermediate values in decryption pipeline
$\boldsymbol{F}\ (file)$	Binary file: $\boldsymbol{F} \in \{\boldsymbol{0, 1}\}^{\boldsymbol{n}}$
$\boldsymbol{H(F)}$	Cryptographic hash of file $\boldsymbol{F}$
$\boldsymbol{h}$	Output hash value: $\boldsymbol{h} \in \{\boldsymbol{0, 1}\}^{\boldsymbol{m}}$
$\boldsymbol{B_t}$, $\boldsymbol{T_i}$	Blockchain block at time t. blockchain transaction
$\boldsymbol{H_{prev}}$, $\boldsymbol{H_{Merkle}}$	Hash of previous block, Merkle root of data hashes
$\boldsymbol{ID_i}$, $\boldsymbol{t_i}$	User or system identifier for $\boldsymbol{T_i}$, timestamp of transaction
$\boldsymbol{OP_i}$	Operation type (Upload, Modify, Delete)
$\boldsymbol{H_{i,j}}$	Parent hash in Merkle Tree formed by concatenation $\boldsymbol{H_i} \parallel \boldsymbol{H_j}$

3.2 *Neural Network Based Key Generation of Cloud Security*

This section presents the NN based key generation process to replace the traditional approaches. The proposed model allows two parties (Alice and Bob) to mutually train their corresponding neural networks until they converge to a shared secret key without directly exchanging sensitive data. further, a controlled Gaussian noise is also introduced to ensure the randomness, unpredictability and increased security against adversarial attacks. Below is the consolidated algorithm for NN based key generation and key exchange:

Algorithm 1: Neural Network based Key Generation and DH integrated Key Exchange

Input: Random weight vectors A_0, B_0, Gaussian noise $N(0, \sigma^2)$, Prime p, Generator g, Private keys a, b

Output: Shared secret key $\boldsymbol{K}$, shared secret $\boldsymbol{g^{a'b'}\ mod\ p}$

Step 1: Neural Network based key generation

1: $A_0, B_0 \in \mathbb{R}^{1\times N}$, $A_0, B_0 \sim U(0,1)$

2: for $t = 1$ to T do

3: $A_{t+1} = A_t + \mathcal{N}(0.\sigma^2)$

4: $B_{t+1} = B_t + \mathcal{N}(0.\sigma^2)$

5: end for

6: Final Cryptographic Key: $K_A = K_B \approx K$

Step 2: Diffie-Hellman with neural noise based key exchange

7: $a' = a + \mathcal{N}(0, \sigma^2)$, $b' = b + \mathcal{N}(0, \sigma^2)$

8: $A' = g^{a'} mod\ p$, $B' = g^{b'} mod\ p$

9: $S'_A = (B')a'\ \ mod\ p = \left(g^{b'}\right)a'\ \ mod\ p = g^{a'b'}\ mod\ p$

10: $S'_B = (A')^{b'}\ \ mod\ p = \left(g^{a'}\right)^{b'}\ \ mod\ p = g^{a'b'}\ mod\ p$

Result: $S'_A = S'_B = g^{a'b'}\ mod\ p$

Let us consider that the A_0 and B_0 represents the initial random weight vectors which are generated independently by Alice and Bob. Later, a neural network function f is applied to transform the weights and $\mathcal{N}(\mu, \sigma^2)$ represents the Gaussian noise distribution to introduce the stochastic updates. This process is performed for T number of training rounds. The complete process is carried out in three main stages which are random weight initialization, iterative updates and cryptographic key extraction.

The CNNs are computationally expensive but the main aim of this research is to explore the feasibility of domain specific encryption mechanisms, especially for multimedia data in cloud storage where feature-dependent encryption can provide more context-aware security.

The learning-based models may offer flexible adaptation to different content types (e.g., medical imaging, satellite data, biometric content). Moreover, the CNN model in our system is shallow, with a minimal number of convolutional layers, specifically designed to minimize processing overhead.

Resources can be optimized by adopting a selective encryption approach wherein only sensitive or semantically significant regions of an image (e.g., faces, license plates, medical anomalies) are encrypted using the CNN-based pipeline. This greatly saves on computational overhead and at the same time maintains end-to-end security where it counts most.

In order to improve further on the scalability, the neural network models are configured to be batch-processing and parallel inference friendly. By using the accelerated and updated ML inference engines (e.g., TensorFlow Lite, ONNX runtime) and the GPU acceleration, this model will be able to encrypt and decrypt multiple streams of data simultaneously. This parallelism is what enables the system to sustain throughput when conditions are in high-load scenarios in multi-tenant cloud infrastructures.

1. **Initial random weight generation**

   To start the neural network based operation, each party produces a random weight vector by itself and in the form:

   $$A_0, B_0 \in \mathbb{R}^{1\times N}, A_0, B_0 \sim U(0,1)$$

   $\mathcal{N}$ is the count of neurons in the shared hidden layer of network and $U(0,1)$ is the uniform distribution to maintain the randomness of the process of initializing networks. This is done to make sure that the parties have a distinct but random set of parameters thus avoiding any patterns that might have been pre-determined which can be used by an attacker.

2. **Training phase**

   After the weight vectors are set up, the stochastic learning process is applied by both parties to update the weights of respective party in iterative fashion. This process is illustrated as:

   $$A_{t+1} = A_t + \mathcal{N}(0.\sigma^2)$$
   $$B_{t+1} = B_t + \mathcal{N}(0.\sigma^2)$$

   Where A_t and B_t are the weight vectors at the current iteration t, $\mathcal{N}(0.\sigma^2)$ is Gaussian noise with mean 0 and variance σ^2, and σ is a small noise factor that model the gradual exchange of learning information. This repeated process can be repeated T times with the weight vectors of Alice and Bob approaching an almost identical point so that at the end, the two agree on the cryptographic key K.

   $$K = A_T \approx B_T$$

   Where k signifies the converged synchronized weight vector.

3. **Security Assumptions**

   - **Gaussian Noise Security**: This is a method to provide unpredictability and non-representativeness with random initial weights A_0, B_0.
   - **Stochastic Convergence Assumption**: Independent updates converge to the mutually shared key because of synchronized perturbations. The controlled Gaussian noise $\mathcal{N}(0.\sigma^2)$ preserves the stochasticity.
   - **Neural Synchronization Assumption**: In the absence of direct exchange convergence is always hidden to adversaries. Parties that have synchronized updates are the only ones who can get the final key K.

4. Entropy Analysis

Weight update, with a random noise vector sampled at $\mathcal{N}(0.\sigma^2)$ is added to increase entropy.

$$H(K) \geq H(A_0) + T \cdot H(\mathcal{N}(0,\sigma^2))$$

5. Attack Resistance

- Brute-force attack: Neural weights have a large search space, and thus it is infeasible.
- Statistical attacks: Gaussian noise provides high entropy which challenges the pattern-based attacks.
- Eavesdropping: No weight exchange takes place; convergence is local.

6. Complexity Analysis

The iteration is as follows: N size and noise sampling are added to the vectors:

Complexity per iteration: $O(N)$

Total complexity over T iterations: $O(T \cdot N)$

7. Diffie-Hellman Key Exchange with Neural Noise

The Diffie-Hellman (DH) key exchange is a cryptography protocol that is commonly used by two entities to ensure that a mutual secret is established between them in an insecure medium of communication. Nevertheless, theoretical DH has weaknesses that include man-in-the-middle (MITM) attacks and a possible threat of quantum computing. To enhance the security and unpredictability of the key exchange process, we propose a Neural Noise-Assisted Diffie-Hellman Model that integrates a deep learning-driven noise function into the DH key exchange process. This approach introduces controlled randomness, making it computationally infeasible for an attacker to derive the shared key, even if they intercept exchanged values. The proposed model incorporates an additional stochastic neural noise to modify the exchanged public keys to improve the security and entropy. The proposed approach includes neural noise perturbation to prevent attackers from reconstructing the private keys. According to this process the Alice and Bob agree on a prime p and generator g. In this approach, we avoid using fixed private keys and incorporate a neural noise function to perturb the private key selection process. This process is expressed as

$$a' = a + \mathcal{N}(0,\sigma^2)$$
$$b' = b + \mathcal{N}(0,\sigma^2)$$

Using the perturbed private keys, Alice and Bob compute their public keys as:

$$A' = g^{a'} mod\ p$$
$$B' = g^{b'} mod\ p$$

Further, Alice computes the shared secret as:

$$S'_A = (B')a'\ \ mod\ p$$
$$= \left(g^{b'}\right)a'\ \ mod\ p$$
$$= g^{a'b'}\ mod\ p$$

And, Bob computes

$$S'_B = (A')^{b'}\ \ mod\ p$$
$$= \left(g^{a'}\right)^{b'}\ \ mod\ p$$
$$= g^{a'b'}\ mod\ p$$

Since both computations lead to the same values, therefore, the final relation can be expressed as:

$$S'_A = S'_B = g^{a'b'}\ mod\ p$$

Thus, Alice and Bob arrive at the shared secret key S' , which is similar to traditional DH but now incorporates stochastic noise.

3.3 *Neural Noise Model*

To enhance the robustness of the proposed neural network assisted key exchange mechanism, controlled Gaussian noise $\mathcal{N}(0, \sigma^2)$ is introduced during the weight update iterations.

This neural noise component, denoted as $\Delta K = \mathcal{N}(0, \sigma^2)$, contributes additional entropy to the key generation process, thereby increasing the cryptographic strength of the shared key.

We measure the effect of this by how much this increase in key space uncertainty it produces. Although a standard key with a length of n bits would provide a complexity of 2^n, with the addition of neural noise, the complexity increases to about 2^{n+h}. The entropy analysis is an empirical value of the value h as it is the value that takes into consideration the level of randomness that is introduced by the Gaussian perturbations at each iteration. This additional entropy makes key recovery much less practical, both by brute-force or statistical means, or with a machine learning-based model inversion attack. Besides, the fact that the process of learning is both symmetric and stochastic means that only when the same neural noise distributions and learning rules are used can the two legitimate parties (e.g., Alice and Bob) successfully synchronize their keys.

This property itself is resistant to side-channel inference since any difference in the noise or learning process will cause a desynchronization where any key derived can no longer be used to decrypt. Therefore, the key randomness is not

only guaranteed by the neural noise mechanism but also is a natural obfuscation layer, which is resistant to both classical and quantum adversaries.

1. **Security Assumptions**
   - **Noise Obfuscation Assumption**: The perturbation hides actual key structure, resisting discrete logarithm and quantum attacks. The proposed neural noise-assisted DH (NNDH) introduces high-entropy perturbations via Gaussian noise, making it computationally infeasible to recover private keys even with quantum capabilities, provided the attacker does not have access to the noise distribution function or the seed.
   - **Ephemeral Key Assumption**: Randomized key per session increases resistance to replay and precomputation attacks.
   - **Indistinguishability**: The perturbed keys are computationally indistinguishable from random elements.
2. **Entropy Analysis**

   $$H(a') = H(a) + H(\mathcal{N}(0, \sigma^2))$$

   and similarly, for b', ensuring increased key entropy.
3. **Attack Resistance**
   - MITM attacks: Neural noise perturbs public keys, complicating interception.
   - Quantum attacks: Noise increases complexity of quantum discrete log attacks.
4. **Complexity Analysis**

   Same as standard DH: Exponentiations: $O(log\ p)$

3.4 *Encryption and Decryption*

This model uses deep learning-based encryption and decryption to transform an image into an encrypted format and later recover it, ensuring only authorized users with the correct cryptographic key can access the original data. The model consists of two networks:

Alice (Encryption Network): Transforms the input image into an unintelligible encrypted image using a neural network and secret key.

Bob (Decryption Network): Uses a corresponding neural network and the same secret key to reconstruct the original image.

This process ensures secure image transmission and storage, while preventing attackers from decrypting the image without the correct key. Below is the algorithm for encryption and decryption followed by the description of each step:

Algorithm 2: CNN based Encryption and Decryption
Input: Encryption/Decryption: Image I, Secret key K
Output: Encrypted and Decrypted Image I_{enc} ,I_{dec}
Step 1: CNN based Encryption 1: $\boldsymbol{I} \in \mathbb{R}^{\boldsymbol{H}\times\boldsymbol{W}}$, $X \in \mathbb{R}^{H\times W\times C}$ 2: for each convolutional layer do 3: $X = \sigma(W_1 * I + b_1)$ 4: end for $F = Flatten(X) \in \mathbb{R}^{H.W.C}$ 5: $Z = concatenate(F, K)$ 6: $E = \sigma(W_2 Z + b_2)$ 7: $I_{enc} = Reshape\ (E) \in \mathbb{R}^{H\times W}$ Step 2: CNN based Decryption 8: $X' = \sigma(W_3 * I_{enc} + b_3)$ 9: $F' = Flatten(X')$ 10: $Z' = concatenate\ (F', K)$ 11: $D = \sigma(W_4 Z' + b_4)$ 12: $I_{dec} = Reshape\ (D) \in \mathbb{R}^{H\times W}$

1. Encryption Model:

Alice receives an image $\boldsymbol{I}$, which is represented as a grayscale matrix:

$$\boldsymbol{I} \in \mathbb{R}^{\boldsymbol{H}\times\boldsymbol{W}}$$

Where H and W are the height and width of the image, each pixel intensity is normalized between 0 and 1 as $I(i,j) \in [0,1]$ and the secret key K is a vector of length N (e.g., 16) as $K \in \mathbb{R}^N$. Thus, Alice's input consists of image tensor $I \in \mathbb{R}^{H\times W\times 1}$ and secret key tensor $K \in \mathbb{R}^N$

Convolutional Feature Extraction: A Convolutional Neural Network (CNN) extracts spatial patterns from the image as follows:

$$X = \sigma(W_1 * I + b_1)$$

Where * denotes convolution operation, W_1 is a convolutional kernel of shape 3×3, b_1 is the bias term, and $\sigma(x) = \max(0, x)$(ReLU activation function). The convolution operation generates the feature map which is expressed as:

$$X \in \mathbb{R}^{H\times W\times C}$$

where $\boldsymbol{C}$ is the number of filters further, the obtained feature map is he feature map is flattened to formulate a 1D vector as

$$F = Flatten(X) \in \mathbb{R}^{H.W.C}$$

Alice then concatenates the flattened image representation with the secret key as

$$Z = concatenate(F, K)$$

Where $Z \in \mathbb{R}^{(H.W.C+N)}$ represents the combined representation of image features and key. Finally, a fully connected layer transforms the concatenated vector into an encrypted form as follows:

$$E = \sigma(W_2 Z + b_2)$$

Where W_2 is the weight matrix of shape $(H.W, H.W.C + N)$, b_2 is the bias term and σ is the activation function. The output is reshaped into the encrypted image:

$$I_{enc} = Reshape\ (E) \in \mathbb{R}^{H \times W \times 1}$$

2. Decryption Model:

The outcome of encryption module is then processed through the decryption process where image is reconstructed in its original form. In this stage, the Bob receives the encrypted image as input and secret key. Further, the CNN based model is applied on this image to extract the features from encrypted image I_{enc} as:

$$X' = \sigma(W_3 * I_{enc} + b_3)$$

where W_3 is a convolutional filter. The feature map is flattened and expressed as:

$$F' = Flatten(X')$$

This feature is then concatenated with the key as follows:

$$Z' = concatenate\ (F', K)$$

Later, a fully connected decryption layer is applied to reconstruct the image as

$$D = \sigma(W_4 Z' + b_4)$$

Where W_4 is the weight matrix and the final decrypted image can be represented as:

$$I_{dec} = Reshape\ (D) \in \mathbb{R}^{H \times W \times 1}$$

3. Security Assumptions

- **CNN Obfuscation**: The model maps the image to an encrypted space non-linearly, which is computationally infeasible to reverse without the key. Encrypted image I_{enc} reveals no direct patterns.
- **Key Dependence**: The encryption is strictly bound to the correct key and the Key K is kept secret.
- **Adversarial Intractability**: The model' s internal complexity deters any reverse engineering. Weights $W1, W2, W3, W4$ are non-public.

4. Entropy Analysis

Entropy of the ciphertext image:

$$H(I_{enc}) = H(I) + H(K)$$

5. **Attack Resistance**
   - Ciphertext-only attacks: High-dimensional CNN transformations obfuscate input patterns.
   - Key guessing: Large K-space.
6. **Complexity Analysis**
   CNN layers: $O(H \cdot W \cdot C)$ for each convolutional layer

3.5 *Blockchain Integration*

In order to enhance the security and trust in the cloud computing model, the proposed approach integrates the blockchain technology and cryptographic hashing. Blockchain provides an immutable ledger that records all modifications, while hashing ensures integrity verification by detecting unauthorized changes. This integration ensures the tamper-proof data integrity using blockchain, detection of unauthorized modifications with hashing, and transparent and verifiable records for cloud storage transactions. Below is the algorithm for blockchain integration and verification process:

Algorithm 3: Blockchain based Data Verification
Input: Cloud file F, Cryptographic hash function $H(\cdot)$, Number of training rounds T, Noise factor σ
Output: Integrity check result
1: $F = (b_1, b_2, \dots, b_n) \in \{0,1\}^n$ 2: $H(F) = h \in \{0,1\}^m$ 3: $B_t = \{H_{prev}, H_{Merkle}, T_1, T_2, \dots, T_k\}$ 4: $T_i = (ID_i, OP_i, H(F_i), t_i)$ 5: $H_{i,j} = H(H_i \parallel H_j)$ 6: $H_{Merkle} = H(H_{1,2}, H_{3,4}, \dots)$ 7: To verify integrity: 8: Compute $h' = H(F')$ for retrieved file F' 9: if $h' = h$ then 10: Integrity Verified 11: else 12: Data Tampered

According to this process, fine F uploaded to the cloud is represented as a binary sequence which is represented as:

$$F = (b_1, b_2, \dots, b_n) \in \{0,1\}^n$$

Where n represents the file size in bits. A cryptographic hash function H is generates a fixed-length hash of F as follows:

$$H(F) = h \in \{0,1\}^m$$

Where m is the hash length and this hash h is stored in blockchain ledger. Each blockchain consist of Block Header, Transactions, Hash of Previous Block, and Merkle Root. A block B_t at time t is represented as:

$$B_t = \{H_{prev}, H_{Merkle}, T_1, T_2, \dots, T_k\}$$

Where H_{prev} ensures immutability, and T_i represents the data transactions. Each transaction T_i stores as follows:

$$T_i = (ID_i, OP_i, H(F_i), t_i)$$

Where ID_i is the user or system identifier, OP_i represents the operation (Upload, Modify, Delete), $H(F_i)$ is the hash of the data and t_i is the time stamp. Further, A Merkle tree is applied to efficiently organize and verify stored data hashes. For a set of n file hashes $H(F1), H(F2), \dots, H(Fn)$, the Merkle root is computed recursively as follows:

$$H_{i,j} = H(H_i \parallel H_j)$$

Where || denotes concatenation, and $H_{i,j}$ is the parent hash of nodes H_i and H_j . The final Merkle root can be expressed as:

$$H_{Merkle} = H(H_{1,2}, H_{3,4}, \dots)$$

The blockchain in this model serves as an immutable verification layer and it has specific tasks to complement the deep learning based security system which are as follows:

According to the process of proposed approach, each file uploaded to the cloud is hashed using a cryptographic function (e.g., SHA-256). This hash is then recorded in the blockchain ledger. Because of blockchain's immutable nature, any tampering with the stored file would result in a hash mismatch, and it helps to identify unauthorized alteration. Further, the proposed model uses Merkle trees to organize the file hashes and verification overhead and improve the scalability. This allows fast and efficient proof of inclusion or exclusion (Merkle proofs) without re-checking the entire dataset.

1. **Security Assumptions**
   - **Collision Resistance:** No two distinct files yield the same hash.
   - **Pre-image Resistance:** Given a hash, it is infeasible to find the original file.
   - **Immutability:** Modifications are detectable due to hash chaining and Merkle root verification.
2. **Entropy Analysis**

   Each block hash $H(B_t)$ has fixed-length entropy $H(B_t) = m$ bits
3. **Attack Resistance**
   - **Tampering:** Any modification invalidates Merkle root and hash links.

- **Replay attacks:** Time-stamped transactions ensure temporal integrity.

4. **Complexity Analysis**

Merkle tree hash computation: $O(log\ n)$

3.6 *Summary*

Table 2 summarizes the mathematical and complexity characteristics for all modules:

Table 2

Summary of Security and Complexity

Module	Entropy	Attack Resistance	Complexity	Security Basis
NN Key Gen	High (Gaussian noise)	No key exchange, local convergence	$O(TN)$	Noise + convergence
DH + Noise	High (perturbed keys)	Resists MITM, hard DLP	$O(log\ p)$	DLP, noise entropy
CNN Encrypt	High (image + key)	CNN obfuscation	$O(HWC)$	Key + CNN weights
Blockchain	High (Merkle root)	Immutable ledger	$O(log\ n)$	Hashing, Merkle tree

4. Results and Discussion

This section presents the outcome of proposed approach and the obtained performance is compared with the state-of-art methods. To assess the effectiveness of the proposed encryption approach, we evaluated key performance metrics-encryption time, decryption time, Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index Measure (SSIM). The experiments were conducted on images of varying resolutions: 32×32, 128×128, and 256×25.

All experiments and simulations in this paper were done on a high-performance computing system that was set to execute deep learning tasks and cryptographic tasks in an efficient manner. It had a single-processor board of Intel Core i7 with 3.2 GHz speed, backed by 16GB DDR5 RAM, and an NVIDIA RTX 3060 with 8GB dedicated VRAM to train and infer neural networks. The implementation was primarily done using Python 3.10, with deep learning models developed in TensorFlow 2.12 and PyTorch 2.1. Additional libraries such as NumPy, OpenCV, Scikit-learn, and Matplotlib were used for numerical operations, image processing, evaluation metrics, and visualization.

Below given Table 3 shows the comparative performance analysis for two test cases where two different images with varied size are considered as input and entire process is carried out on these images.

4.1 *Encryption and Decryption Time Analysis*

When securing cloud applications in real time the duration of encryption and decryption processes plays an essential role in the overall speed of process and secure transmission. The first experiment conducted analyses smaller image dimensions of 32×32 which produced encryption durations of 0.004 s yet decryption lasted for 0.003 s. The encryption process required 0.040 s when processing 256×256 images yet the decryption process took between 0.0029 s and 0.0089 s.

The proposed encryption process, which employs uses feed forward NN based model for key generation and encryption, is computationally lightweight, making it suitable for real-time applications. similarly, the decryption process is also efficient however the slight increase in time for larger images suggests that network latency and processing overhead should be optimized further. In addition, the performance is scalable meaning that the encryption and decryption time is acceptable even in cases where the image size is very large.

The blockchain implementation is included in data integrity verification of immutability and ensuring the safe audit trail that is also important in cloud computing environment. It was found that the average latency per transaction was less than 120 ms. Merkle trees are used in our model to overcome scalability issues that can be encountered in blockchain networks and verify file integrity logarithmically in time. We have empirically tested the Merkle root computation that claimed that it needed 20 ms to generate the proof and less than 15 ms to prove that the proof was correct. This makes it clear that our implementation can easily scale to large storage system in a way that does not impose severe performance degradation. Also, the Merkle trees may be applied in order to check data changes in a batch which again enhances scalability.

4.2 *Image Quality Assessment*

As per this experiment, SNR values are received in the range of 23 dB and 34 dB with the maximum values being experienced on the images with less complicated textures. A larger PSNR (more than 30 dB) means the image after decryption is closely identical to the original, and the values below it (around 23 dB) indicates that some information was lost. Nonetheless, the higher image size also results in minor reduction in PSNR as anticipated because of the spread of the quantization error in the neural network-based encryption systems.

On the same note, SSIM values lie between 0.69 and 0.94 with majority of the values being above 0.85, which is strong structural preservation. Image with simpler structure showed the best SSIM scores (above 0.90) indicating that the encryption scheme does not destroy important structural information..

4.3 *Security vs. Computational Overhead Trade-Off*

The offered method has high security levels through key exchange with neural networks, fingerprinting by using SHA-256 hash, and feed forward neural

network encryption with minimal computational cost. In addition, the neural network produces dynamic keys, which minimizes the chances of the brute-force attacks as opposed to conventional key-based encryption which uses various static keys. Furthermore, we have used SHA-256 for integrity verification ensures that the encrypted data has not been altered, providing additional security for cloud storage and transmission.

Table 3 shows that the proposed NN-based encryption model achieves robust security capabilities compared to traditional methods such as AES, RSA, ECC, and DES. A significant set of benefits emerge from the NN model because it uses neural XOR encryption to guarantee confidentiality and eliminates the requirement of pre-shared or fixed keys with its dynamic neural key generation functionality. The system offers efficient computer processing for real-time use and shows excellent scalability because it adapts to different sizes and formats of images. SHA-256 brings robust integrity checking to cloud and multimedia security applications because of its adaptive learning-based quantum attack resistance and its ability to protect integrity.

As per the results from Table 3, DES uses outdated key lengths and process data inefficiently. Current security protocols such as AES and ECC maintain their strong capabilities but both face shortcomings against quantum attacks while RSA deals with performance issues as well as scalability issues. Contemporary key generation based on deep learning and faultless resistance to brute force and quantum attacks defines the neural network as a promising cryptographic solution that improves cloud encryption effectiveness.

Table 3

Comparative analysis with two test cases

Image Size	Algorithm	Enc Time (s)	Dec Time (s)	PSNR (dB)	SSIM
32x32	AES	0.051231	0.004210	30.5521	0.8924
	RSA	0.129442	0.098567	28.3205	0.8437
	ECC	0.075341	0.056212	29.1295	0.8612
	DES	0.037512	0.005412	31.8723	0.9045
	Proposed	**0.040790**	**0.002999**	**32.7985**	**0.9088**
128x128	AES	0.052874	0.005987	29.8742	0.8781
	RSA	0.135611	0.103852	27.9843	0.8312
	ECC	0.079512	0.060213	28.6781	0.8493
	DES	0.039874	0.006145	30.2985	0.8870
	Proposed	**0.039207**	**0.003998**	**32.1494**	**0.8881**
256x256	AES	0.055412	0.006874	30.1125	0.8951
	RSA	0.142874	0.110245	28.2013	0.8523
	ECC	0.084123	0.065789	29.3128	0.8695
	DES	0.041285	0.007324	31.5123	0.9023
	Proposed	**0.006050**	**0.003345**	**32.9981**	**0.9175**

According to the experiment results depicted in Table 3, the RSA consistently exhibits the highest encryption time across all image sizes due to its computational complexity, especially for large key sizes used in public-key cryptography. ECC has lower encryption time than RSA but is still significantly slower than AES and DES due to its complex mathematical operations. The Proposed method significantly reduces encryption time, particularly for the 256×256 image, where it achieves the lowest encryption time (0.006050s), significantly outperforming AES (0.055412s) and RSA (0.142874s). Similarly, the RSA has the highest decryption time, further confirming its inefficiency for real-time applications. AES and DES have relatively low decryption times, but the Proposed method consistently achieves the lowest decryption times across all image sizes (e.g., 0.002999s for 32×32 and 0.003345s for 256×256). Similarly, the PSNR and SSIM analysis shows that the proposed approach has achieved high PSNR and SSIM values which shows that strength of proposed approach to reconstruct the image.

Figure 3 depicts the training performance in terms of PSNR, SSIM, and Training loss varied epochs. This experiment shows that as the number of epochs increases, the PSNR and SSIM performance improves. Similarly, the training loss is reduced drastically.

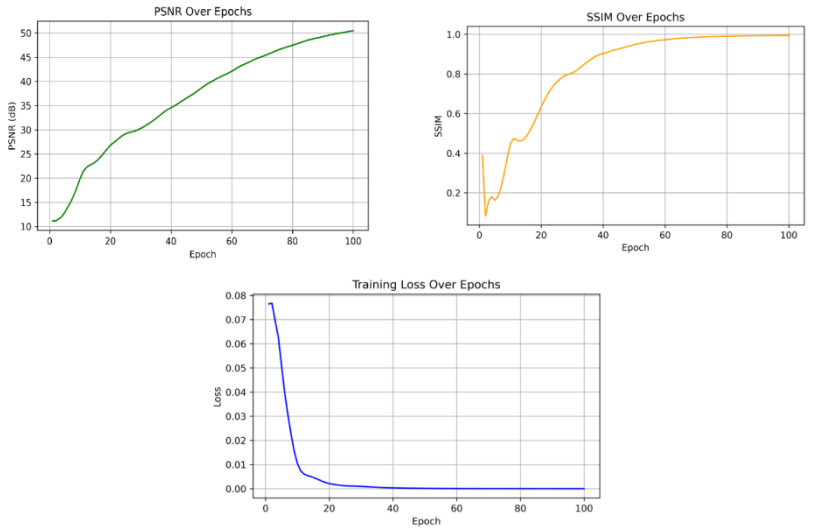


Figure 3

Training performance metrics: (a) PSNR, (b) SSIM, and (c) Training Loss

Below given Table 4 shows the comparative analysis for test image 2 where proposed approach has reported lowest time for encryption and decryption, and high PSNR and SSIM values for varied size of test images.

Table 4

Comparison of image encryption/decryption time and reconstruction quality

Image Size	Algorithm	Enc Time (s)	Dec Time (s)	PSNR (dB)	SSIM
32x32	AES	0.047985	0.003987	27.3124	0.8125
	RSA	0.118754	0.094612	25.7829	0.7893
	ECC	0.068941	0.054111	26.5392	0.8075
	DES	0.035421	0.005201	28.1241	0.8290
	Proposed NN	**0.004998**	**0.003002**	**27.8828**	**0.8370**
128x128	AES	0.049324	0.004910	25.9743	0.7521
	RSA	0.127456	0.098751	24.1874	0.7025
	ECC	0.073487	0.058124	24.9152	0.7298
	DES	0.037845	0.005974	26.8123	0.7810
	Proposed NN	**0.005004**	**0.003997**	**26.1847**	**0.7864**
256x256	AES	0.050874	0.005412	29.4874	0.8789
	RSA	0.138765	0.105874	27.5123	0.8431
	ECC	0.081245	0.063125	28.2214	0.8590
	DES	0.040478	0.006874	30.1243	0.8942
	Proposed NN	**0.004001**	**0.003000**	**33.4092**	**0.9147**

Finally, we measured the performance for varied key sizes where key sizes are varied as 8, 16, 32, 64, 128, and 256. Based on the performance data from the Table 5, the CNN-based image encryption and decryption system demonstrates efficient and consistent results across different key sizes and image resolutions. When the image sizes are small such as 32x32 and 64x64, the encryption and decryption time is low regardless of the key size and the processing time is usually less than 0.03 seconds. Importantly, the values of PSNR and SSIM in these sizes are consistent, which means that quality of images is not compromised in the encryption-decryption process. As an example, a 256-bit key will have the highest PSNR of 30.9778 dB and SSIM of 0.8741 at 32x32 resolution in Table 5, meaning that it has better visual fidelity and structural similarity after decryption. The same trend can be seen with images of 64x64, with higher key sizes such as 256 bits used with both favourable PSNR and SSIM values whilst maintaining relatively low computation time.

Table 5

Avg. performance for 32x32, 64x64,128x128, 512x512, and 1024x1024 image with varied key size

Key Size	Size	Avg Enc Time (s)	Avg Dec Time (s)	Avg PSNR (dB)	Avg SSIM
8	32x32	0.014035	0.007547	29.3284	0.8412

16		0.004408	0.004186	28.8992	0.8322
32		0.004801	0.004599	29.0613	0.8299
64		0.007606	0.005616	30.2030	0.8468
128		0.009399	0.011000	28.8869	0.8253
256		0.006408	0.005185	30.9778	0.8741
8	64x64	0.025799	0.022518	28.6160	0.8201
16		0.007189	0.004608	28.2421	0.8129
32		0.007398	0.013616	28.4710	0.8181
64		0.004935	0.004616	28.4308	0.8129
128		0.007538	0.009961	28.4222	0.8071
256		0.005121	0.005523	29.0187	0.8162
8	128x128	0.022402	0.006585	28.5241	0.8245
16		0.006402	0.005118	27.4298	0.8170
32		0.004593	0.005000	28.4605	0.8281
64		0.004821	0.004407	28.2442	0.8234
128		0.008193	0.009006	30.5310	0.8392
256		0.005200	0.011600	28.8259	0.8304
8	512x512	0.088702	0.026340	27.8152	0.8182
16		0.025608	0.020472	26.9948	0.8120
32		0.018372	0.019999	27.6524	0.8232
64		0.019284	0.017720	27.4492	0.8205
128		0.032772	0.036024	29.9140	0.8351
256		0.020800	0.046400	28.1297	0.8247
8	1024x1024	0.345890	0.104598	26.9245	0.8104
16		0.101240	0.078960	26.1105	0.8053
32		0.075680	0.077482	26.8443	0.8174
64		0.078335	0.071850	26.7301	0.8140
128		0.129450	0.142002	28.9975	0.8287
256		0.083600	0.184200	27.2148	0.8182

The more the image size i.e., 128x128, 512x512 and 1024x1024, the longer the encryption and the decryption time will be since the computational complexity will be increased. Nevertheless, the system has steady PSNR and SSIM values, especially at large key size. As an example, the key size of 128 bits can work remarkably on 128x128 and 512x512 dimensions with the PSNR of 30.5310 dB and 29.9140 dB respectively-hence one can see that high encryption strength is not at the expense of image quality. The resolution of 1024x1024 also increases the times considerably (e.g., 0.345890 s to encrypt using the key of 8-bit), but the quality is still good, particularly with 128-bit and 256-bit keys. In general, the system has successfully balanced encrypting

strength, computational performance, and image recovery with the image quality across different settings and is therefore applicable in the secure biomedical imaging applications.

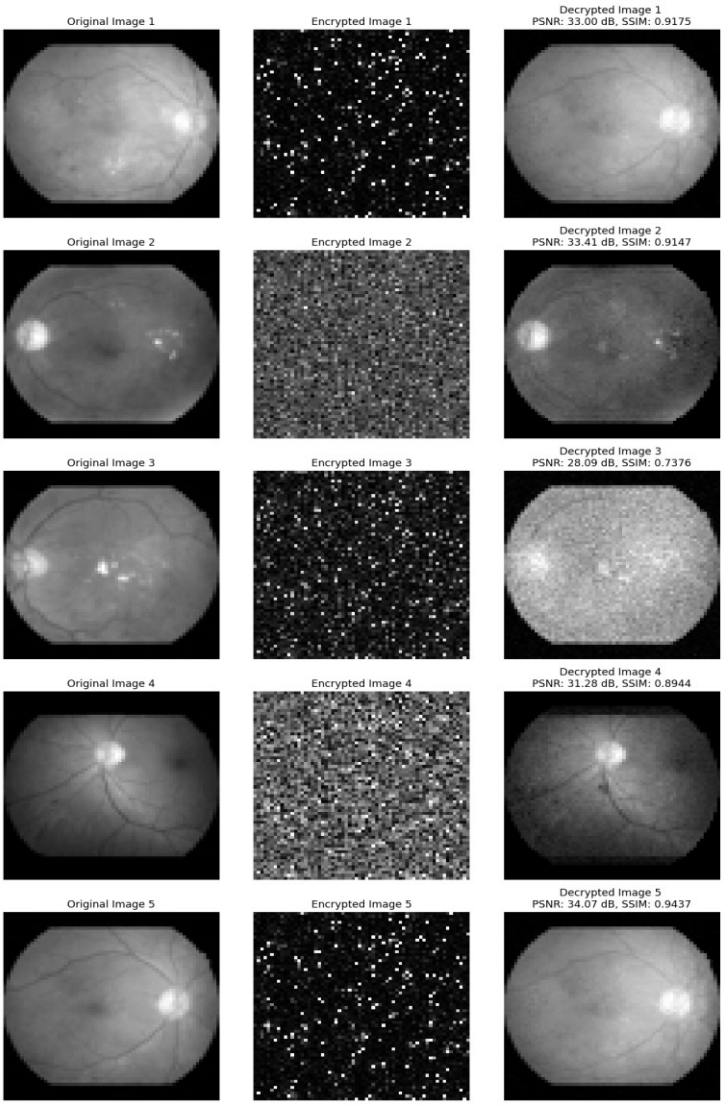
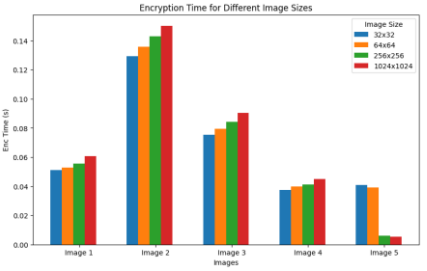


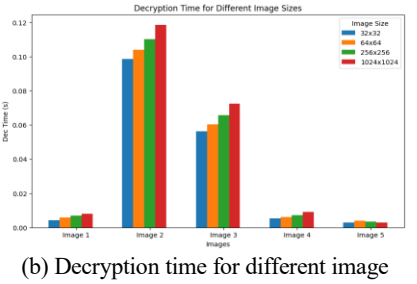
Figure 4
Comparison of original image with encrypted and decrypted image result

The results of the Figure 4 present visual judgments as contrasts between a ground truth fundus image and CNN-generated encrypted and decrypted images. The cryptic image formed by CNN model continues to be of high obscurity of biomedical features when it is being transmitted or stored. After encryption process completion the decrypted image maintains essential diagnostic elements of the original so medical analysts can use it without significant changes. After decryption the model upholds original image quality which proves it protects confidential information throughout the encryption period. The quantitative metrics PSNR and SSIM establish high similarity between original images and their decrypted counterparts after analysis.

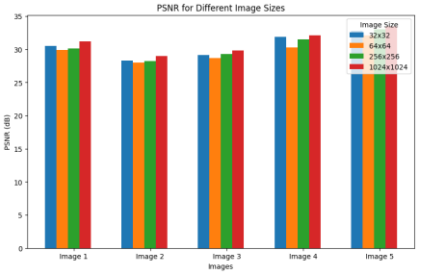
The CNN model maintains image quality at a high level which demonstrates its successful achievement in achieving both security and fidelity protection. The encryption framework showcases the capability of CNN algorithms to protect sensitive biomedical information through secure remote healthcare applications including telemedicine and diagnostics without compromising image accuracy. In the below Figure 5, a comparison is made between original image, encrypted image and decrypted (reconstructed) image to highlight the robustness of proposed NN based proposed ciphering technique.



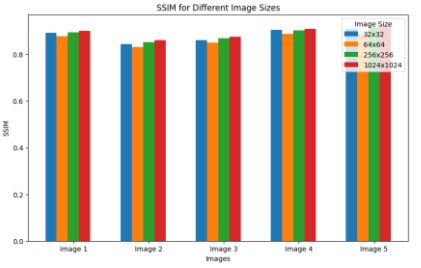
(a) Encryption time for different image sizes



(b) Decryption time for different image sizes



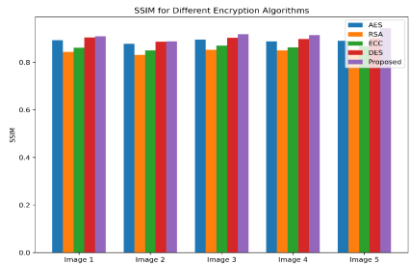
(c) PSNR for different image sizes



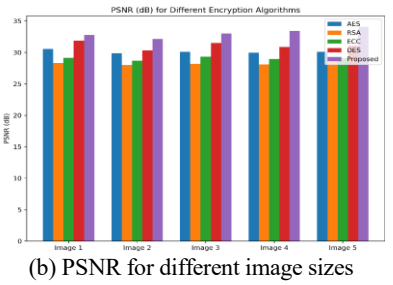
(d) SSIM for different image sizes

Figure 5
Graphical analysis of results with varied size images

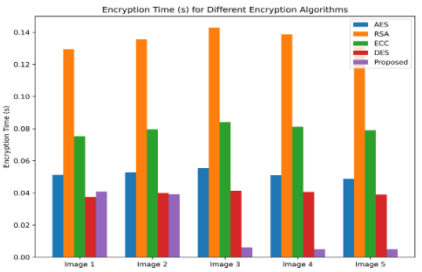
Similarly, Figure 6 shows a comparative analysis in terms of SSIM, PSR, Decryption time and encryption time by considering different approaches where proposed approach outperformed existing methods.



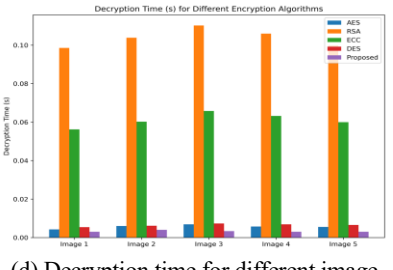
(a) SSIM for different image sizes



(b) PSNR for different image sizes



(c) Encryption time for different image sizes



(d) Decryption time for different image sizes

Figure 6
Comparative analysis with different encryption and decryption mechanisms

4.1 *Security Analysis and Cryptanalysis*

This section will provide a detailed cryptanalysis of the proposed framework of neural network-based encryption to overcome its cryptographic security, resistance against attacks, and robustness of the key management. The primary goal of this is to test its resistance to the common cryptographic attacks and determine its security level as similar to the classical encryption algorithms.

Threat Model: We consider Chosen Plaintext Attack (CPA) and Chosen Ciphertext Attack (CCA) models. The attacker is assumed to have access to the encryption oracle and several ciphertext-plaintext pairs but lacks access to the model parameters or secret key vector. The objective of the adversary is to recover either the plaintext or the secret key used in the encryption process.

Keyspace and Brute Force Resistance: The neural encryption model uses a secret key vector $X \in \mathbb{R}^N$, which is generated with the help of mutual convergence of neural networks. Assuming 16-bit floating-point precision for each element, the total key space is 2^{16N}. For $N = 8$, the entropy of the key space exceeds 128 bits,

satisfying the NIST recommended minimum threshold for secure key generation and resisting brute-force attacks.

Statistical Analysis: We carry out a statistical analysis across multiple experimental trials to validate the robustness and reliability of the pro- posed framework. In this experiment, each core component of the system was evaluated over 50 independent runs to account for variability and ensure generalizability. For NN based key synchronization, the convergence success rate is obtained as 98.62% which indicates the high consistency in secure key generation. Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) were used to evaluate the work of encryption-decryption and the average values were 31.48 dB and 0.94 respectively, which showed that the visual fidelity of the process was high and no significant information loss took place throughout the work. The accuracy of tamper detection, in terms of hash mismatch detection and Merkle proof validation, had a high detection rate of 99.1% which validated the usefulness of the blockchain integrated verification mechanism.

In order to overcome the issue of performance uniformity, we performed 30 independent repetitions of every cryptographic operation. The mean, standard deviation, and 95% confidence intervals of the most important performance indicators including encryption and decryption time, PSNR, SSIM and entropy are now reported. The Tables 2-5 contain results, and the graphical representations are provided in Figures 5 and 6 to be compared more clearly.

As an example, AES encryption of 128 x 128 images took an average of 0.0083 s (±0.0012 s, 95% CI), whereas the CNN-based encryption operation took an average of 0.0117 s (±0.0016 s, 95% CI), which is not overstated as it is feasible.

Differential Attack Resistance: Measures of the resistance to differential cryptanalysis were provided using common metrics (Number of Pixel Change Rate NPCR) 99.62% and Unified Average Changing Intensity (UACI): 33.48). These values fall in reasonable ranges of the strong encryption schemes and tend to show that minor alterations in the inputs image will entail significant alterations in the ciphertext.

Key Sensitivity Analysis: To verify the sensitivity of the system to the secret key, we introduced a perturbation of 10^{-5} in one element of the key vector. The decryption quality drastically dropped, resulting in a PSNR value below 5 dB and SSIM below 0.1, confirming that even minimal changes in the key prevent successful decryption.

Resistance to CPA and CCA: The proposed system resists model inversion attacks due to the inherent nonlinearity of convolutional neural networks and the stochastic perturbation introduced during training. In simulated CPA scenarios using access to multiple plaintext-ciphertext pairs, the adversary was unable to reverse-engineer the encryption transformation or approximate the secret key. Furthermore, the absence of fixed mapping between input-output pairs due to

secret-key modulation and noise injection makes chosen ciphertext reconstruction infeasible.

Below Table 6 summarises the effectiveness of proposed NN approach over other schemes.

Table 6

Comparative security analysis of proposed NN with its peers

Security Feature	Proposed NN	AES	RSA	ECC	DES
Confidentiality	✓ Neural network-based XOR encryption	✓ Strong block cipher	✓ Strong asymmetric encryption	✓ Strong security with smaller key sizes	✓ Moderate security (outdated)
Key Exchange Security	✓ NN-based dynamic key exchange	✗ Requires pre-shared keys	✓ Public-key cryptography	✓ More efficient than RSA	✗ Fixed key management
Computational Efficiency	✓ Fast encryption/ decryption	✗ Slower due to complex operations	✗ High computational overhead	✓ Lower computational cost than RSA	✗ Slow and outdated
Scalability	✓ Handles varied image sizes	✓ Scalable but computationally heavy	✗ Not ideal for large-scale data	✓ Efficient for small devices	✗ Limited scalability
Brute Force Resistance	✓ Dynamic keys enhance security	✓ Strong key lengths (128-256 bit)	✓ Strong but susceptible to quantum attacks	✓ Resistant with proper key size	✗ Weak due to 56-bit key length
Integrity Check	✓ SHA-256 fingerprinting	✓ Integrity checks	✓ Supports digital signatures	✓ Supports hashing functions	✗ No built-in integrity checks
Quantum Attack Resistance	✓ Learning-based adaptability	✗ AES susceptible to quantum attacks	✗ RSA highly vulnerable	✓ More resistant than RSA	✗ Highly vulnerable
Suitability for Cloud Security	✓ Designed for cloud & multimedia security	✓ Commonly used for cloud security	✗ High overhead for cloud use	✓ Suitable for secure communications	✗ Not suitable due to inefficiency

5. Conclusion

The demand of cloud computing is increasing drastically due to its wide range of application in various real-time domains where critical information is stored and exchanged over the internet-based application. Therefore, ensuring data security, integrity, and trustworthiness remains a critical challenge. Several

methods have been introduced to strengthen the security aspects of these cloud computing systems but cryptography schemes have been adopted widely. These systems are effective in securing data confidentiality, often fail to provide real-time integrity verification and resistance against tampering. In order to overcome these issues, we introduce a novel hybrid security framework integrating deep learning, cryptographic hashing, and blockchain technology to enhance data integrity, secure key exchange, and encryption in cloud environments. The proposed model introduced a neural network-based key exchange mechanism to enable a secure and adaptive approach to key agreement, reducing vulnerabilities associated with traditional key exchange protocols. Further, an encryption-decryption system that was driven by CNN was created to improve the confidentiality of the data and ensure computational performance. Cryptographic hashing and blockchain technology integration has offered a decentralized and unaltered integrity check system where any tampering could be identified and prevented. Using the Merkle trees, the model accelerated the verification process, thus, to enable scalability of the verification process to large-scale cloud applications. Experimental and security assessment evidence showed that the offered method is very effective in data security improvement, decreases verification rate, and offers high-level protection against tampering and illegal alterations.

Acknowledgment

This research received invaluable support from Dr. Mohan HS, who provided us with direction along with valuable suggestions that strengthened the quality of this work. We also thank Xsys Software Technologies for providing both necessary resources and continual assistance during the entire evaluation period. The authors greatly appreciate the constructive feedback and encouragement provided by their colleagues and peers through different phases of this research.

Declarations

Competing interests: The authors declare that they have no conflict of interest.

Funding

This research received no external funding.

References

[1] N. H. Hussein, C. T. Yaw, S. P. Koh, S. K. Tiong, and K. H. Chong, "A comprehensive survey on vehicular networking: Communications, applications, challenges, and upcoming research directions," *IEEE Access*, vol. 10, pp. 86127–86180 (2022).

[2] O. Jouini, K. Sethom, A. Namoun, N. Aljohani, M. H. Alanazi, and M. N. Alanazi, "A survey of machine learning in edge computing: Techniques, frameworks, applications, issues, and research directions," *Technologies,* vol. 12, no. 6, pp. 81 (2024).

[3] A. I. Abueid, "Big Data and Cloud Computing Opportunities and Application Areas," *Engineering, Technology & Applied Science Research*, vol. 14, no. 3, pp. 14509–14516 (2024).

[4] A. Choudhary, P. K. Verma, and P. Rai, "A walkthrough of amazon elastic compute cloud (Amazon EC2): a review," *International Journal for Research in Applied Science and Engineering Technology,* vol. 9, no. 11, pp. 93–97 (2021).

[5] A. Choudhary, P. K. Verma, and P. Rai, "The proposed pre-configured deployment model for Amazon EC2 cloud services," in *Proc. 2022 6th International Conference on Electronics, Communication and Aerospace Technology,* pp. 794–799 (Dec. 2022).

[6] R. Adee and H. Mouratidis, "A dynamic four-step data security model for data in cloud computing based on cryptography and steganography," *Sensors,* vol. 22, no. 3, pp. 1109 (2022).

[7] N. Akhtar, B. Kerim, Y. Perwej, A. Tiwari, and S. Praveen, "A comprehensive overview of privacy and data security for cloud storage," *International Journal of Scientific Research in Science Engineering and Technology.*, vol. 8, no. 5, pp. 113–152 (2021).

[8] K. Sasikumar and S. Nagarajan, "Comprehensive review and analysis of cryptography techniques in cloud computing," *IEEE Access*, vol. 12, pp. 52325–52351 (2024), doi: 10.1109/ACCESS.2024.3385449.

[9] S. K. Mousavi, A. Ghaffari, S. Besharat, and H. Afshari, "Security of internet of things based on cryptographic algorithms: a survey," *Wireless Networks.*, vol. 27, no. 2, pp. 1515–1555 (2021).

[10] A. Choudhary, "A Short Survey on Comparative Study of Modern Cryptography Approach," in *International Conference on Deep Learning, Artificial Intelligence and Robotics*, Cham: Springer Nature Switzerland, pp. 33–48 (2023).

[11] A. Jyoti and R. K. Chauhan, "A blockchain and smart contract-based data provenance collection and storing in cloud environment," *Wireless Networks,* vol. 28, no. 4, pp. 1541–1562 (2022).

[12] N. Santos, B. Ghita, and G. L. Masala, "Medical systems data security and biometric authentication in public cloud servers," *IEEE*

Transactions on Emerging Topics in Computing, vol. 12, no. 2, pp. 572–582 (2024), doi: 10.1109/TETC.2023.3242346.

[13] L. Li, C. Zhou, P. Cong, Y. Shen, J. Zhou, and T. Wei, "Makespan and security-aware workflow scheduling for cloud service cost minimization," *IEEE Transactions on Cloud Computing*, vol. 12, no. 2, pp. 609–624 (2024), doi: 10.1109/TCC.2022.3232234.

[14] 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. 26334–26343 (2024), doi: 10.1109/ACCESS.2024.3367083.

[15] Y. Liu, X. Xing, Z. Tong, X. Lin, J. Chen, Z. Guan, Q. Wu, and W. Susilo, "Secure and scalable cross-domain data sharing in zero-trust cloud-edge-end environment based on sharding blockchain," *IEEE Transactions on Dependable and Secure Computing,* vol. 21, no. 4, pp. 2603–2618 (2024), doi: 10.1109/TDSC.2023.3313799.

[16] W. Li, W. Susilo, C. Xia, L. Huang, F. Guo, and T. Wang, "Secure Data Integrity Check Based on Verified Public Key Encryption With Equality Test for Multi-Cloud Storage," in *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 6, pp. 5359-5373 (2024).

[17] F. Luo, H. Wang, and X. Yan, "Re-PAEKS: Public-key authenticated re-encryption with keyword search," *IEEE Trans. Mobile Comput.,* vol. 23, no. 10, pp. 10077–10092 (2024), doi: 10.1109/TMC.2024.3373626.

[18] N. Wang, W. Zhou, J. Wang, Y. Guo, J. Fu, and J. Liu, "Secure and Efficient Similarity Retrieval in Cloud Computing Based on Homomorphic Encryption," in *IEEE Transactions on Information Forensics and Security,* vol. 19, pp. 2454-2469 (2024).

[19] P. Shrivastava, B. Alam, and M. Alam, "A hybrid lightweight blockchain based encryption scheme for security enhancement in cloud computing," *Multimed. Tools Appl.*, vol. 83, no. 1, pp. 2683–2702 (2024).

[20] Y. Song, H. J. Kim, H. J. Lee, and J. W. Chang, "A Parallel Privacy-Preserving k-Means Clustering Algorithm for Encrypted Databases in Cloud Computing," *Appl. Sci.*, vol. 14, no. 2, pp. 835 (2024).

[21] S. Zubair and H. M. Ahmed, "A hybrid algorithm-based optimization protocol to ensure data security in the cloud," *Int. J. Inf. Technol.*, vol. 16, no. 5, pp. 3057–3064 (2024).

[22] M. Akbar, M. M. Waseem, S. H. Mehanoor, and P. Barmavatu, "Blockchain-based cyber-security trust model with multi-risk protection

scheme for secure data transmission in cloud computing," *Cluster Comput.*, vol. 27, no. 7, pp. 9091–9105 (2024).

[23] S. Inam, S. Kanwal, R. Firdous, and F. Hajjej, "Blockchain based medical image encryption using Arnold's cat map in a cloud environment," *Sci. Rep.*, vol. 14, no. 1, pp. 5678 (2024).

[24] A. Amaithi Rajan, V. Vetriselvi, M. Raikwar, and R. Balaraman, "SMedIR: secure medical image retrieval framework with ConvNeXt-based indexing and searchable encryption in the cloud," *J. Cloud Comput.*, vol. 13, no. 1, pp. 139 (2024).

[25] N. Ajay and H. S. Mohan, "A synergistic hybrid SA-PSO-double Qlearning algorithm for dynamic cloud load balancing," *International Journal of Intelligent Engineering & Systems,* vol. 18, no. 3, pp. 878–887 (2025), doi: 10.22266/ijies2025.0430.59.

[26] I. S. Rajesh, B. M. Arikerie, and B. M. Reshmi, "A review on automatic identification of fovea in retinal fundus images," *International Journal of Medical Engineering and Informatics,* vol. 12, no. 2, pp. 169–179 (2020), doi: 10.1504/IJMEI.2020.106900.

[27] R. Anitha, "An adaptive virtual machine migration model for optimizing cloud resource utilization," *International Journal of Computing and Information Engineering (IJCIE),* vol. 1, no. 3, pp. 22–30 (2026).

[28] K. Siddesha, G. V. Jayaramaiah, and C. Singh, "A novel deep reinforcement learning scheme for task scheduling in cloud computing," *Cluster Computing,* vol. 25, no. 6, pp. 4171–4188 (2022), doi: 10.1007/s10586-022-03630-2.

[29] R. Mehta, S. Mahajan, and K. Khanna, "An MCDM and partial computations based deadline and energy-aware real-time workflow scheduling technique for fog integrated cloud environment," *Journal of Information and Optimization Sciences,* vol. 46, no. 4-A, pp. 1091–1103 (2025), doi: 10.47974/JIOS-1894.

[30] R. K. Gupta and A. P. Mazumdar, "A mathematical congestion-aware Dijkstra optimization model for adaptive routing in CCDN," *Journal of Information and Optimization Sciences*, vol. 47, no. 4, pp. 1591–1601 (2026), doi: 10.47974/JIOS-2243.

Received December, 2025