

Multi-Tiered Security Framework for Container-Based Secure Data Migration in Cloud Environment Using Hardy Weinberg-APIary Organization-Based Optimization Algorithm

S. Durgaprasad^{1,*}, Arif Mohammad Abdul²

¹Research Scholar, GITAM Deemed to be University Hyderabad, Telangana 502329, India

²Computer Science and Engineering, GITAM Deemed to be University Hyderabad, Telangana 502329, India

Received 8 October 2025; Received in revised form 19 May 2026

Accepted 4 June 2026; Available online 30 June 2026

ABSTRACT

Cloud data migration transfers services, applications, and data from local infrastructure to the cloud to improve scalability, flexibility, and cost efficiency. However, migration faces challenges including security risks, data loss, downtime, performance degradation, and energy consumption. To address these issues, this study proposes a multi-tiered security framework consisting of five phases: encryption, load balancing, virtual machine (VM) selection, VM privacy protection, and data migration. User tasks are collected and allocated to containers for efficient resource utilization. A novel document-based chunking Blowfish algorithm (DBC-BA) enhances data confidentiality and processing speed during encryption. For load balancing, the proposed LPF-DBSCAN algorithm employs Local Poisson Forest density estimation to improve clustering accuracy under varying data densities. VM selection is optimized using the Hardy Weinberg-APIary Organization-Based Optimization Algorithm (HW-AOBOA), while T-Closeness protects sensitive VM information. Implemented in Python, the proposed method achieves secure cloud data migration with low energy consumption (38.56 J) and reduced delay (1857 ms).

Keywords: Cloud; Data migration; Encryption; Hardy weinberg-apiary organization-based optimization algorithm; Load balancing; Virtual machine

1. Introduction

Data migration refers to the process of transferring data, databases, and applications from on-premises hardware or legacy systems to cloud environments. Data migration is currently an obligatory process for firms wishing to enhance working flexibility, reduce costs, and enhance scalability due to the rapid uptake of the cloud [1]. Cloud databases offer several advantages, such as enhanced disaster recovery, remote management, enhanced performance, and reduced infrastructure maintenance. They are extensively used in a variety of sectors, including customer relationship management (CRM), e-commerce, banking, and big data analytics [2]. There are numerous benefits to that data movement, but large data transfers present a number of difficulties, such as security issues, performance lag, data loss, and outages. Due to the growing use of cloud computing platforms, data security is one of the most urgent problems in the cloud [3]. Customers and organizations can sometimes endure a significant loss when data is stolen or corrupted. At the same time, robust security and compliance protocols must be in place to safeguard sensitive data during the migration process. A cloud is a vast computer network, and while moving data, security must be improved. Sensitive data is protected during transmission and storage using an encryption technique that ensures only authorized people can access it. Another technique to improve data security is load-balancing [4]. This distributes the workload among computers connected to a network to prevent overload, speed up response times, and boost productivity. Distributing work evenly among all available servers is made possible by load-balancing techniques [5]. Because load-balancing techniques keep some computing nodes from becoming overloaded while

making sure that others are not left idle, they can be used to optimize each task's response time. To achieve this, the load balancer first groups incoming tasks into clusters and then sends the clustered data into virtual machines (VMs) for balanced processing [6]. Cloud computing uses a machine learning based load balancing approach to migrate the data and improve the performance. The machine learning based method has advantages, but it faces challenges and is costly. Since data centers consume a high amount of power, increasing operating expenses adversely affect the environment, and energy efficiency is also a pressing issue [7]. However, data transfer still poses challenges due to issues with load balancing, energy efficiency, data loss, performance issues, and security issues [8]. Along with these issues, data security remains a significant hurdle, as VMs remain vulnerable to numerous forms of attacks during the migration process [9]. To transfer a large amount of data to the cloud environment, one essentially needs optimized frameworks to provide efficiency and automated migration technology [10]. Existing cloud data migration frameworks mainly focus on individual objectives such as encryption, load balancing, VM selection, or data migration. To address the previously mentioned concerns, this study proposes a multi-tiered security framework for cloud data migration. The proposed framework introduces a unified multi-tiered architecture that jointly integrates five main phases including secure encryption, adaptive load balancing, intelligent VM selection, VM privacy preservation, and secure data migration management. This framework provides companies with the insights necessary to plan, execute, and streamline their cloud migration, ensuring a seamless transition and long-term stability. The main con-

tribution of this study is given below:

- Initially, the user tasks are collected and assigned to the container. The effective data collection and organization of user tasks are essential for optimizing the migration strategy and ensuring efficient resource allocation.
- To introduce an efficient encryption method to protect sensitive data during migration, a novel DBC-BA is proposed. In traditional BA, speed can be affected when changing keys. The proposed DBC-BA overcomes this limitation by incorporating DBC during the subkeys generation step, thereby enhancing processing speed.
- To reduce migration complexity, a novel LPF-DBSCAN is proposed for load balancing. Unlike traditional DBSCAN, which struggles with varying densities and assigns clusters based on density criteria, LPF-DBSCAN uses LPF-based density estimation to achieve more accurate clustering.
- A novel HW-AOBOA is proposed for optimal VM selection. In HWAOBOA, if the newly generated bee has fitness better than the queen, it will be selected to be the new queen using the HW principle.
- To ensure VM privacy preservation, T-Closeness anonymity is used to protect the VM's sensitive information (i.e., VM IP address, etc.).
- Finally, the proposed HWAOBOA is utilized to migrate data, enhancing migration speed while minimizing downtime and energy consumption.

- The proposed method's performance is analyzed using various metrics, such as encryption and decryption time, security level, memory usage, throughput, computational complexity, energy consumption, clustering accuracy, cluster stability, execution time, network lifetime, delay, packet loss rate, response time, memory and CPU usage, resource utilization, and load, and comparing them with the existing techniques.

The structure of the paper is as follows. Relevant studies on data migration are reviewed in Section 2. Section 3 presents a detailed presentation of the proposed multi-tiered security framework for secure data migration. Section 4 discusses the proposed approach simulation findings. Section 5 concludes the paper.

2. Related Works

Sharma, N.K., et al. [11] developed an energy-conscious VM assignment strategy using a branch-and-price and multi-dimensional VM migrations (MDVMM) approach for a cloud data center. The MDVMM approach reduced energy consumption, service level agreement (SLA) violations, and VM migration overhead. However, maintaining a balance between energy savings and SLA compliance remained challenging.

Reddy, Y.H., et al. [12] presented the jaya dung beetle optimization (JDBO) strategy for secure VM migration and load balancing in cloud environments. The method improved scalability, migration efficiency, and resource utilization. However, the JDBO algorithm increased computational complexity.

Brahmam, M.G. [13] developed a load-balanced and energy-efficient VM mi-

gration strategy using ant colony optimization (ACO)-genetic algorithms (GA) along with bidirectional long short-term memory networks (Bi-LSTM) and a recurrent graph neural network (RGNN). The technique improved performance, security, and reduced energy consumption. However, integrating ACO with GA increased computing complexity. Ma, Z., et al. [14] presented an adaptive dynamic threshold method for controlling VM migrations and lowering energy consumption. The method optimized energy consumption while improving the quality of service. However, the approach mainly focused on VM consolidation, ignoring other cloud environment factors.

Thabit, F., et al. [15] presented a new lightweight cryptographic algorithm (NLCA) for secure cloud. The NLCA method offered both low computational cost and efficient encryption/decryption performance. However, it was more challenging to design and implement.

Ibrahim, M., et al. [16] developed an efficient adaptive migration algorithm (EAMA) for VM migration. The approach reduced migration overhead and improved resource utilization. However, this strategy resulted in more SLA violations and worse migration decisions.

Mangalagowri, R., and Venkataraman, R. [17] developed an SDN-based capability and access control (CAC) technique to enhance secure data transmission using the adaptive firefly algorithm (AFFA). The method improved throughput, security, and reliability while reducing energy consumption, cost, and time complexity. However, the SDN controller raised administrative expenses, latency, and system complexity.

Tyagi, S.S. [18] developed a secure image encryption model using a Convolu-

tional Autoencoder (CAE) with advanced encryption standard (AES) and Fernet encryption techniques. The method enhanced image confidentiality and data integrity in cloud environments. However, the CAE and dual encryption required higher memory and storage.

From the above studies, it is observed that most existing studies mainly focus on individual objectives such as encryption, load balancing, resource allocation, or data migration. Moreover, existing methods often suffer from higher computational complexity, increased energy consumption, higher delay and execution time, higher memory utilization, poor workload balancing, insufficient VM privacy protection, or inefficient migration management. To overcome these limitations, this study proposes a novel multi-tiered security framework that integrates modified encryption, adaptive clustering-based load balancing, intelligent VM selection, VM privacy preservation, and secure migration within a unified architecture.

3. Proposed Methodology

The process of transferring an organization's workloads, apps, and data from local systems to cloud environments is known as data migration. Large data transfers often come with difficulties, including security threats, performance issues, data loss, and downtime. To address these problems, this study suggests a secure cloud data migration framework that includes several essential phases, including encryption, load balancing, VM selection, VM information privacy protection, and data migration. The suggested multi-tiered security structure for secure data migration in a cloud environment is depicted in Fig. 1.

Initially, the process begins with the collection of user tasks, which are orga-

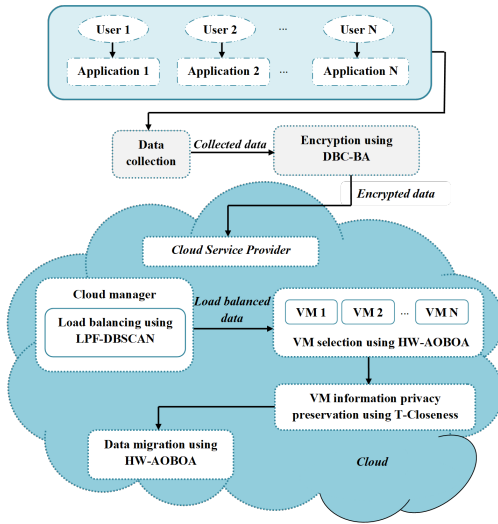


Fig. 1. Overview Diagram of the Proposed Secured Data Migration System.

nized and assigned to the container for efficient resource allocation and migration. To protect the sensitive data during migration, the proposed DBC-BA is used for encryption. The DBC-BA improves the traditional BA by introducing a DBC approach in subkeys generation, significantly enhancing processing speed. Once encrypted, the data is sent to the cloud service provider (CSP). A cloud manager then uses the proposed LPF-DBSCAN clustering algorithm for load balancing. Unlike traditional DBSCAN, which has trouble with varying densities, LPF-DBSCAN uses LPF-based density estimation to improve workload distribution and reduce migration complexity. Next, the optimal VM is chosen by the proposed HW-AOBOA. According to the HW principle, a newly produced bee that has a better fitness than the queen is selected as the new queen in HW-AOBOA. The sensitive data of the VM is then protected during the migration process by T-Closeness anonymity. Lastly, the data is migrated using the HW-AOBOA. The proposed system

provides a secure, effective, and seamless way to transfer data from on-premise systems to cloud environments. Details of the suggested framework are provided in the following sections.

3.1 Encryption using document-based chunking-blowfish algorithm

The BA is a 64-bit symmetric block cipher that encrypts plaintext input one block at a time. Both encryption and decryption use the same key length, which can range from 32 to 448 bits. This BA process consists of three main steps: key expansion, encryption, and decryption. In the key expansion stage, a P-array with 18-bit sub-keys and four 32-bit S-boxes is utilized. However, the speed of the typical BA may be affected by frequent key changes. To solve this issue, the proposed DBC-BA introduces a DBC in the subkey generation step, which greatly improves processing speed.

3.1.1 Key generation using document-based chunking

In the DBC-BA, key generation is improved by using DBC in the subkey generation step. The total length of the documents and the number of chunks or iterations are used to create a hash value. This hash is then used to create a DBC key, which is used to initialize the BA subkeys. By modifying the standard key expansion step with this approach, the DBC-BA improves the speed and efficiency of re-keying.

Step 1: Document Representation:

Let, $D = \{d_1, \dots, d_n\}$ be the input document represented as a byte stream. Where, each $d_m \in B$ (one byte), d_m the m-th byte of the document, and B binary set each byte $\in B$.

The total length of the document in bytes is calculated using the following Eq.

(3.1):

$$L_d = \sum_{m=1}^n \text{length}(d_m). \quad (3.1)$$

Step 2: Chunking: To enable efficient key management, the document is divided into fixed-size or variable-size chunks. Assuming a fixed chunk size c_s , the total number of iterations or chunks c_n is provided by:

$$c_n = \frac{L_D}{c_s}. \quad (3.2)$$

Each chunk is denoted as c_1 where i denotes iteration or chunk index $i \in \{1, 2, \dots, c_n\}$.

Step 3: Hash Computation: For each chunk index i , a unique 256-bit hash is derived using SHA-256. The hash input string st_i is constructed as:

$$st_i = \text{str}(L_d) \parallel \text{str}(i) \parallel k_x, \quad (3.3)$$

where st_i hash input string for the i -th chunk, k_x is a user-defined base encryption key, and $\parallel$ is string concatenation.

Step 4: DBC key generation: The SHA-256 hash of produces a 256-bit digest:

$$h_s^i = \text{SHA256}(st_i). \quad (3.4)$$

This hash output is converted to a 32-byte key stream:

$$k_s^i = \text{bytes}(h_s^i) = \{b_1^{(i)}, \dots, b_{32}^{(i)}\}, \quad (3.5)$$

where, k_s^i is a byte stream derived from the h_s^i . This byte stream is referred to as the DBC key and is used to initialize the BA subkeys.

Step 5: Sub-keys generation using DBC key: In DBCBA, S-boxes and P-array are dynamically updated by the DBC key k_s^i . The sub-keys generation process based on DBC is given below:

(i) Start.

(ii) Set the S-boxes and P-array with standard fixed values.

(iii) Perform an XOR operation between each P-array entry and the corresponding bits of the created DBC key (P1 with DBC key bits, P2 with DBC key bits, and so on) until all P-array entries have been XORed.

$$p[j] = p[j] \oplus k_s^i[j \bmod 32], \\ j = 0, 1, \dots, 17. \quad (3.6)$$

(iv) Utilizing the subkeys generated in the preceding stage, use the BA to encrypt all-zero strings.

(v) Use the outcome produced in step iv to swap $P1$ and $P2$.

(vi) Encrypt the step-iv outcome using the updated subkeys that were generated in step-iii.

(vii) Use the outcome produced in step-vi to swap $P3$ and $P4$.

(viii) Repeat this process until every entry in the P array has been updated.

(ix) Update the S-box entries using the same encryption procedure.

$$S_q[n] = S_q[n] \oplus k_s^i[(q \times 256 + n) \bmod 32], \\ q = 0, 1, \dots, 3, n = 0, 1, \dots, 255. \quad (3.7)$$

(x) Stop.

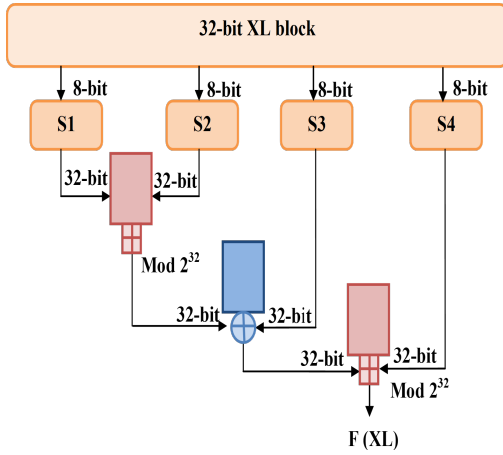


Fig. 2. F-Function diagram of the DBC-BA.

3.1.2 Encryption and decryption

Encrypting a message to prevent it from being deciphered by an outsider is the primary objective of the DBC-BA. The goal of decryption is to return an encrypted message to its original format. The encryption and decryption processes are identical, except that the P-arrays (P0, P1, ..., P17) are used in reverse during decryption. For the decryption process, the encryption key remains in use.

The data is encrypted using a 16-round Feistel network, which splits the input data block in half before encrypting it across multiple rounds. The DBC-BA uses an F-function to split a 32-bit plaintext into four equal parts for encryption and decryption. Each 8-bit subdivision is then converted into a 32-bit data stream using the S-box. The 32-bit data resulting from each 8-bit subdivision is XORed to get the final 32-bit output. The round sub-keys are generated using the F-function. The F-function uses four s-boxes, one XOR operation, and two OR operations in Eq. (3.6). Fig. 2 illustrates the F-Function.

$$F(xL) = ((s_1 + s_2, \mod 2^{32}) \cdot \text{xors}_3) + s_4, \mod 2^{32}. \quad (3.8)$$

The DBC-BA encryption pseudocode is as follows:

- (i) Create two 32-bit halves, such as xL and xR , from the 64-bit input data.
- (ii) While $i = 1$ to 16, repeat steps (a)-(c):
 - (a) Use xL and $P[i]$ to perform an XOR operation.
 - (b) Determine xL F-function such that:

$$F(xL) = ((s_1 + s_2, \mod 2^{32}) \cdot \text{xors}_3) + s_4, \mod 2^{32}.$$
 - (c) XOR the result of $F(xL)$ with xR .
- (iii) Switch xL and xR .
- (iv) Perform an XOR function among xR and $P[16]$.
- (v) Execute the XOR function among xL and $P[17]$.
- (vi) Integrate xL and xR to get the ciphertext.

The following is the pseudocode for the decryption procedure:

- (i) Create two 32-bit halves, such as xL and xR , from the 64-bit input data.
- (ii) While $i = 16$ to 1, repeat steps (a)-(c):
 - (a) Use xL and $P[i]$ to perform an XOR operation.
 - (b) Determine xL F-function such that:

$$F(xL) = ((s_1 + s_2, \mod 2^{32}) \cdot \text{xors}_3) + s_4, \mod 2^{32}.$$
 - (c) XOR the result of $F(xL)$ with xR .

- (iii) Switch xL and xR .
- (iv) Perform an XOR function among xR and $P[1]$.
- (v) Execute the XOR function among xL and $P[0]$.
- (vi) Combine xL and xR to get the plain-text.

3.2 Load balancing using local Poisson forest-based DBSCAN

In this phase, the encrypted data undergoes load balancing to ensure efficient VM selection. Load balancing distributes the workload across multiple VMs to optimize resource utilization and prevent overload. This approach provides flexible and balanced data distribution, prevents certain VMs from being overloaded while others remain idle, reduces downtime, and enhances scalability, adaptability, and efficiency. For effective load balancing, the proposed LPF-DBSCAN clustering algorithm is employed, which reduces complexity during migration by effectively grouping data. Unlike traditional DBSCAN, LPF-DBSCAN uses LPF density estimation to handle varying densities, enabling more accurate cluster formation and improved performance.

DBSCAN is a stochastic clustering approach that facilitates the formation of clusters based on the density of elements and their distance from one another, considering item closeness. Two parameters are used in the algorithm operation: epsilon (ε) (a distance threshold) and MinPts (a minimum number of points). Epsilon is the greatest distance between two points for them to be regarded as neighbors. Two points are deemed adjacent if their distances are less than or equal to epsilon.

In traditional DBSCAN, a fixed global ε value is used for all data points, which may lead to inaccurate clustering when the data contains regions of varying density. Dense regions may become over-clustered, while sparse regions may be incorrectly identified as noise. To address this, we replace the global radius with a point-specific local radius $\varepsilon(p)$ derived from an LPF intensity estimate. The LPF mapping uses the Poisson nearest-neighbor distribution to determine adaptively $\varepsilon(p)$ for each data point, improving clustering accuracy across regions of differing densities. This adaptive mechanism improves clustering robustness, reduces incorrect noise identification, and enhances load balancing performance during VM migration.

A Poisson point process is a basic stochastic model where the number of points in each region follows a Poisson distribution and points appear independently. In clustering, such models underpin approaches for estimating local point densities, as distance-based sampling methods convert nearest neighbor distances into local density estimates. The nearest-neighbor distance distribution r for a homogeneous Poisson process with constant intensity λ is provided by:

$$G(r) = 1 - \exp(-\lambda\pi r^2), \quad (3.9)$$

where λ is the local intensity (mean number of points per unit area) and $G(r)$ is the probability that a randomly chosen point. By selecting a target inclusion probability $P \in (0, 1)$, which specifies the desired probability that a point has at least one neighbor within distance r , the LPF-adaptive radius can be computed by solving $G(r) = P$, yielding:

$$\varepsilon(p) = r = \sqrt{\frac{-\ln(1 - P)}{\lambda\pi}}. \quad (3.10)$$

The resulting r becomes $\varepsilon(p)$ in LPF-DBSCAN. This adaptive $\varepsilon(p)$ ensures LPF-DBSCAN adapts to both dense and sparse regions, improving clustering accuracy.

The smallest number of neighbors (data points) inside an $\varepsilon(p)$ radius is called *MinPts*. The following is the general formula for determining the minimal *MinPts*.

$$MinPts \geq D + 1, \quad (3.11)$$

where, D denotes the number of dimensions.

Using two parameters, the method identifies the core point, boundary point, and noise point, as shown in Fig. 3. Starting from the initial point, DBSCAN creates a cluster by calculating the neighborhood radius $\varepsilon(p)$ to find the core point. If the total number of points in this neighborhood is greater than or equal to a preset *MinPts*, then that point is considered a core point. DBSCAN iteratively adds reachable points to the cluster from a core point inside the $\varepsilon(p)$ neighborhood. If these points are close to a core point, they join the same cluster. A border point is a point that is near a core point but has fewer points inside $\varepsilon(p)$ than *MinPts*. An outlier or noise is a point that is not a core or a boundary point. A point p $\varepsilon(p)$ neighborhood in a dataset is described below:

$$N(p) = \{q \in D | dist(p, q) \leq \varepsilon(p)\}. \quad (3.12)$$

Here, $N(p)$ is the neighborhood of the point p and $dist(p, q)$ denotes the distance between two neighbors

The definition for a core point condition is: if

$$|N(p)| \geq MinPts. \quad (3.13)$$

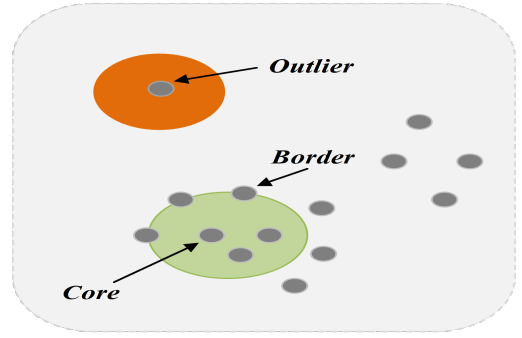


Fig. 3. LPF-DBSCAN clustering.

The definition for a border point condition is: if

$$|N(p)| = MinPts. \quad (3.14)$$

The definition for a noise point condition is: if

$$|N(p)| < MinPts. \quad (3.15)$$

The LPF-DBSCAN algorithm proceeds as follows: The algorithm begins by selecting a data point that has not yet been visited. For each data point, the algorithm must determine the adaptive $\varepsilon(p)$ value, which defines the distance used to mark the neighborhood of the core point. Then, using this marked region, it determines the points that fall within that region. The values of the points that are located in this area are similar. Core points are selected iteratively, and the neighborhoods are expanded until the edge of the point is reached. Noisy data points that fall outside the range will be eliminated. As it proceeds, the algorithm combines core points and their surrounding border points to form clusters. Once a cluster is fully expanded, the algorithm moves on to the next unexplored point. The procedure concludes when every point has been analyzed and grouped into a cluster or marked as noise. Thus, the encrypted

data is effectively clustered using the LPF-DBSCAN, ensuring optimized distribution of workloads and improved migration efficiency.

3.3 VM Selection using HW-AOBOA

In this phase, the load-balanced data is then prepared for efficient VM selection. The framework uses the proposed HW-AOBOA to identify the optimal VM. AOBOA is a novel metaheuristic optimization technique based on the biological processes and organizational behavior of the honeybees within the apiary. AOBOA replicates the behaviors of hives' workers, drones, and queens throughout their life cycles. In traditional AOBOA, the fitness of the freshly created worker bee and the existing queen is compared using a predetermined threshold, which may limit the accuracy of optimal selection. To overcome this limitation, HW-AOBOA integrates the HW principle, ensuring both optimality and genetic diversity during queen selection. A newly created bee will be chosen as the new queen in the HW-AOBOA if it has better fitness than the existing queen, ensuring that the most effective VM is chosen for data migration. The proposed HW-AOBOA steps are given below:

Step 1: Initial population: The population of individuals in each apiary is randomly produced and can be represented as (N). Every individual in the HW-AOBOA population represents a potential solution to the VM selection. In the population, there are (h) hives, and each hive has (b) bees. The population size is computed according to Eq. (3.16).

$$N = h \times b. \quad (3.16)$$

The population consists of a queen, which is determined to be the optimal solution,

several drones (10%–20% of the bees), and workers (80%–90% of the bees).

Optimization Problem Formulation: The VM selection process in the proposed framework is formulated as an optimization problem. The objective of the proposed HW-AOBOA is to identify the optimal VM for secure data migration by minimizing the total energy consumption during VM selection and migration.

Let $VM = \{VM_1, \dots, VM_n\}$ represent the set of available VM in the cloud environment. Each solution in the HW-AOBOA population represents a candidate VM selection solution. The VM selection decision variable is represented as:

$$x_i = \begin{cases} 1, & \text{if VM is selected for migration} \\ 0, & \text{Otherwise} \end{cases} \quad (3.17)$$

Energy consumption is considered the primary optimization objective because excessive energy utilization directly increases operational cost and resource overhead in cloud environments. Therefore, the proposed HW-AOBOA selects the VM with minimum energy consumption to achieve energy-efficient secure data migration. The fitness function based on energy consumption is expressed in Eq. (3.18).

$$fitness = minimum(\alpha EC), \quad (3.18)$$

where α denotes the weights ranging from 0 to 1, F denotes the fitness value, and EC represents the total energy consumption.

The energy is mostly determined by the power used by each VM during optimal VM selection and data migration. Consequently, the total energy consumption of the cloud setup is directly dependent on the power consumed by the underlying re-

sources of each VM. Below is the VM energy consumption model.

$$EC = \frac{1}{T} \sum_{t=1}^T G, \quad (3.19)$$

where T is the entire amount of time and is the amount of power used in the cloud VM. The expression for the power usage is given below:

$$G = p \times G_{\max} + (1 - p) \times G_{\max} \times RU^c. \quad (3.20)$$

The maximum power consumption is represented by the term $G_{\max}$, while the cloud's resource utilization is represented by RU^c , and is denoted by:

$$RU^c = \frac{1}{N} \sum_{r=1}^N RU. \quad (3.21)$$

Therefore, the proposed HW-AOBOA searches for the optimal VM with minimum energy consumption for secure cloud data migration.

Step 2: Fitness calculation: In HW-AOBOA, each member's quality (fitness) is determined using Eq. (3.18). The individual with the lowest energy consumption, representing the best fitness, is chosen as the optimal VM for data migration.

Step 3: Update phase: In this stage, the population is updated through the six biologically inspired processes of HW-AOBOA, as given below.

Phase 1: Drone Exchange: Drone exchange is carried out by randomly selecting two hives $((h_1, h_2))$ and drones $((d_1, d_2))$, followed by a swap operation.

Phase 2: Queen Fertilization and Bee Breeding: Every hive has a queen. By choosing a queen (the most fit individual) and a group of the best drones, queen fertilization creates new individuals. To create K new bees, each drone fertilizes the

queen under the control of the fertilization ratio (ftr_{ratio}). It is estimated that 20% to 40% of drones are fertilized. The fertilization process is depicted in Eq. (3.21).

$$new - b_{hi} = \begin{cases} Q_{hi} + rand(d_{hn} - Q_{hi}), \\ \text{if } \left(\frac{d_{hi} - Q_{hi}}{d_{hi}} < 0.5 \right) \\ \frac{1}{2}(Q_{hi}) + rand\frac{1}{2}(d_{hi}), \\ \text{Otherwise} \end{cases}, \quad (3.22)$$

where Q_{hi} and d_{hi} stand for the i -th hive's queen and drone, respectively, and is a random number between 0 and 1.

Phase 3: Worker Lifecycle: The age of every worker affects the possibility of replacing a current bee with a fresh one. The lifespan of a worker could range from 1 to 60 days. A worker's life cycle has been described by the number of days that make up their age. Depending on the age of the worker, the number of days needed to complete simple or 1-opt (also called one-optima), 2-opt, and 3-opt tasks determines the worker's life cycle. This definition is as follows: when the worker's age is between 1-10 days, a simple translocation is performed; between 11-25 days, a 2-opt operation is applied; between 26-50 days, a 3-opt operation is applied; and between 51-60 days, the 2-opt operation is applied. If the worker's age exceeds 60 days, it is discarded. According to Eq. (3.22), a specific number of changes are made to the selected worker each time to produce a new bee:

$$NW_{hi} = W_{hi} + r \left(\frac{W_{hi}^{\max-g} - W_{hi}^g}{W_{hi}^{\max-g}} \right) \times W_{hi}, \quad (3.23)$$

where r denotes a random number between 0-1, $W_{hi}^{\max-g}$ denotes the worker's maximum age, and W_{hi} denotes a worker of the i -th hive.

Phase 4: Queen Investiture: This stage mimics the queen's replacement behavior. The newly created bee will be chosen to be the new queen if its fitness surpasses that of the queen, while the old one will perish. In traditional AOBOA, the fitness of the freshly created worker bee and the existing queen is compared using a threshold θ . If the new bee's fitness surpasses this threshold, it replaces the queen; otherwise, the queen is retained. However, this deterministic rule may limit the diversity of candidate solutions. To overcome this limitation, we enhance this step by applying the HW principle during queen selection, which introduces a probabilistic mechanism that maintains optimality and genetic diversity. The integration of the HW principle into AOBOA is motivated by its ability to preserve population diversity during iterative optimization. The HW principle models probabilistic genotype distribution using allele frequencies, enabling balanced exploration and exploitation during the search process. This mechanism improves the global search capability of the algorithm, enhances convergence stability, and reduces the possibility of trapping in suboptimal solutions during VM selection.

For the current hive h_i , the normalized allele frequencies are computed as: Let

$$p = \frac{f(Q_i)}{f(Q_i) + f(new - b)}, \text{ and } q = 1 - p, \quad (3.24)$$

where p denotes the normalized contribution of the queen, q the contribution of the new bee, $f(Q_i)$ = fitness of current queen, and $f(new - b)$ = fitness of the best new bee from breeding.

By the HW principle, the expected distribution of genotypes is given by:

$$p^2 + 2pq + q^2 = 1, \quad (3.25)$$

where p^2 represents the probability of keeping the queen, $2pq$ represents the probability of forming a hybrid queen, and q^2 the represents replacing the queen with a new bee. Based on these probabilities, a random number $r \in [0, 1]$ is generated to determine the queen update:

If $r < p^2$: retain the current queen ($Q_{next} = Q_i$).

If $p^2 \leq r < p^2 + 2pq$: replace queen with a hybrid ($Q_{next} = \text{hybrid}(Q_i, new - b)$).

If $r \geq p^2 + 2pq$: replace the queen fully with the new bee ($Q_{next} = new - b$)

Final check: Replace only if

$$f(Q_{next}) - f(Q_i) \geq \theta. \quad (3.26)$$

By probabilistically selecting between the current queen, hybrid queen, and newly generated bee, the proposed HW-AOBOA maintains diversity among candidate solutions while still favoring fitter individuals. This mechanism improves convergence behavior and optimization performance for energy-efficient VM selection in secure cloud data migration.

Phase 5: Fading out: Mathematically, to regulate the quantity of dying bees, a control parameter known as fd_{ratio} is applied. It is estimated that the fd_{ratio} , which is a ratio inspired by nature, is between 10% and 30% of bees.

Phase 6: Swarming: This stage mimics another of the queen's behaviors, known as the swarming phenomenon. This stage resembles the swarming behavior of honeybees, where a crowded hive is divided into two hives. Mathematically, to regulate the hive separation process a control parameter known as s_{ratio} is added. s_{ratio} is thought to be a user-defined value that is adjusted according to the specific issue.

Step 4: Termination: The algorithm iteratively continues the HW-AOBOA up-

date process until the optimal *VM* for data migration is discovered.

By using HW-AOBOA, the optimal *VM* is selected for data migration. Before initiating migration, the framework applies T-Closeness anonymity to safeguard sensitive *VM* details, ensuring secure processing of selected *VM* during migration.

3.4 *VM* privacy preservation using T-closeness

To ensure privacy preservation in cloud environments, T-Closeness anonymity is used to protect the sensitive information of *VMs* (i.e., IP addresses, configuration details, etc.), thereby preventing unauthorized disclosure. T-Closeness aims to limit the amount of knowledge an attacker can obtain regarding a specific *VM*-sensitive attribute. It ensures every equivalency class's distribution of confidential attributes is similar to that of the entire dataset. As a result, attackers find it more difficult to execute attribute linking as there is less correlation between the sensitive properties and the quasi-identifier attributes.

The equivalence class satisfies T-Closeness if the difference between the distribution of the sensitive attribute in this class and the distribution of the attribute throughout the entire dataset is less than a threshold. If every equivalency class satisfies T-Closeness, the dataset is considered to meet this condition. Assessing T-Closeness primarily depends on the specific distance metric used between distributions. The Earth Mover's Distance (EMD), which calculates the least amount of cost needed to shift the probability mass of one distribution U into another V , is used to assess the separation between distributions. The cost depends on both the amount of mass moved and the distance it is moved. The number of

bins between two bins determines the distance between them for numerical properties like IP addresses. The ordered distance is $(x_i, x_j) = (i - j)/(m - 1)$ if the values of the numerical property are $\{x_1, \dots, x_m\}$, where $x_i < x_j$ if $i < j$. Now, the EMD of the ordered distance can be computed as follows if U and V are distributions over which probability u_i and v_i .

$$Emd(U, V) = \frac{1}{m-1} \sum_{i=1}^m \left| \sum_{j=1}^j u_j - v_j \right|. \quad (3.27)$$

By applying T-Closeness, the distribution of sensitive *VM* attributes in each anonymized group remains close to the overall dataset distribution. This prevents attackers from inferring specific *VM* details and protects against homogeneity attacks and background knowledge attacks.

3.5 Data migration using HW-AOBOA

After applying T-Closeness to protect the sensitive information of the selected *VM*, the framework proceeds to the data migration phase. In this phase, the proposed HW-AOBOA is used to execute the data migration process, improving migration speed while ensuring data security. The pseudocode for HW-AOBOA is given in Algorithm 1.

Algorithm 1: Pseudocode of HW-AOBOA.

Input: Load Balanced User Data, Privacy Preserved Selected *VM*.

Output: Secure Data Migration to the Selected *VM*.

1. Use Eq. (3.16) to create solutions.
 2. Determine the solution fitness using Eq. (3.17).
 3. Select Q_i
 4. The remaining solutions should be divided into two (W) and (D).
 5. Set W^g to 1
 6. While (not termination condition) do
-

Phase 1 (Drone Exchange)

7. For $i = 1$ to N do
8. For $j = 1$ to $exch_{ratio}$ do
9. Choose two hives h_1 and h_2 at random.
10. Choose two (D) at random, d_1 and d_2 from h_1 and h_2 .
11. $h_1 \leftarrow d_2$
12. $h_2 \leftarrow d_1$
13. End for

Phase 2(Fertilization and Bee Breeding)

14. For $m = 1$ to $fert_{ratio}$ do
15. Select d_{best} and w_{best} in h_i .
16. Fertilize Q_i with d_{best} using Eq. (3.21) to generate several $new - b(s)$.

Phase 3 (Worker Lifecycle)

17. Use Eq. (3.22) to fertilize Q_i with w_{best} to produce many $new - b(s)$.
18. Case age of W^g do
- 19.1–10: Use basic fertilization on W^g
20. 11 to 25: use 2-optimal fertilization on W^g
21. 26 to 50: use 3-optima fertilization on W^g
- 22.51 to 60: use 2-optimal fertilization on W^g
23. Otherwise: drop W^g
24. End case
25. End for

Phase 4 (Queen Investiture Modified using HW principle)

26. Compute allele frequencies:

$$p = \frac{f(Q_i)}{f(Q_i) + f(new - b)},$$

and $q = 1 - p.$

27. Compute genotype probabilities: p^2 -retain queen; $2pq$ -hybrid; q^2 -replace the new bee.
28. Generate random $r \in [0, 1)$
29. if $r < p^2$: $Q_{next} = Q_i$. Final check: Replace only if

$$f(Q_{next}) - f(Q_i) \geq \theta \text{ Eq. (3.23)}$$

30. else if $p^2 \leq r < p^2 + 2pq$:
 $Q_{next} = \text{hybrid}(Q_i, new - b)$
31. else if $r \geq p^2 + 2pq$: $Q_{next} = new - b$
32. end if
33. if $f(Q_{next}) - f(Q_i) \geq \theta$
then $Q_{next} - Q_i$
34. end if

Phase 5 (Fading out)

35. Drop d_{worst} and w_{worst} from h_i .

Phase 6 (Swarming)

36. $h = 0$
37. If the Size of $h_1 \geq s_{ratio}$ then:
38. Divide h_i into the two hives shown below:

39. $\frac{1}{2}Wh_i$ stay in h_i , and the second $\frac{1}{2}Wh_i$ will be moved to the new hive $h(n + 1)$.
40. $\frac{1}{2}dh_i$ stay in h_i , the second $\frac{1}{2}dh_i$ will be moved to the new hive
41. Qh_i remains, proceed to step 14 to produce the new hive's $h(n + 1)$ queen.
42. End if
43. $h = h + 1$;
44. $N = N + h$;
45. End for
46. End While
47. End.

Hence, the proposed HW-AOBOA securely and efficiently migrates data, achieving faster data transfer, reduced energy consumption, minimized delay, and enhanced privacy preservation.

4. Results and Discussion

This section presents the results of the proposed multi-tiered secure cloud data migration framework. The proposed framework was implemented using Python on a Windows 10 operating system with Intel Core i5 processor and 6 GB of RAM. The experimental evaluation was conducted using the CloudSim simulation platform to analyze the effectiveness of the proposed framework in a cloud environment. The experiments were performed using the GWA-Bitbrains Dataset, collected from <https://www.kaggle.com/datasets/gauravdhamane/gwa-bitbrains>. The dataset contains performance metrics of 1,750 VM from a real-world cloud data center, which are used to simulate realistic cloud workload conditions and evaluate the effectiveness of secure data migration. The cloud simulation environment consists of heterogeneous VMs with varying resource capacities and workload conditions to emulate realistic cloud computing environments. The experiments were conducted using 1000 user workload tasks distributed

across 50 VMs to evaluate migration efficiency, load balancing performance, energy consumption, and privacy preservation under different workload conditions. Table 1 shows the simulation parameters of the proposed model.

4.1 Experimental results

The results of the suggested framework are presented in this section. The framework comprises five main phases: encryption, load balancing, VM selection, VM privacy preservation, and data migration. Each phase's performance is assessed using different metrics.

4.1.1 Performance comparison of encryption

The suggested DBC-BA-based encryption algorithm effectiveness is evaluated against other methods, such as Twofish (TF), ChaCha20, and traditional BA, in terms of computational complexity, memory utilization, throughput, encryption and decryption time, and security level.

Fig. 4 illustrates the encryption and decryption time comparison between the proposed DBC-BA and existing techniques. The proposed DBC-BA obtains a minimum of 2256 ms and 2189 ms for encryption and decryption, respectively. Classical BA gets 5471 ms and 5689 ms, ChaCha20 records 6282 ms and 6491 ms, and TF gets 7481 ms and 7623 ms for encryption and decryption. Based on the outcomes, the suggested DBC-BA required less time for encryption and decryption. This is due to the use of DBC-based subkeys generation that boosts the processing speed. Fig. 5 shows the comparison of memory utilization during encryption and decryption for the suggested DBC-BA with existing algorithms. The suggested DBC-BA has a minimum memory usage of 3875 kb and 3824 kb during encryption and decryption. As a comparison,

Table 1. Simulation Parameters.

Models	Parameters	Values
DBC-BA	Block size	64 bits
	Key size	256 bits
	Number of rounds	16
	Hash function	SHA-256
LPF-DBSCAN	Chunk size	4 KB
	MinPts	6
	Initial ϵ range	0.5
	Neighborhood radius	LPF-based adaptive radius
	LPF intensity parameter (λ)	0.5
	Target inclusion probability (P)	0.95
	Maximum cluster iterations	100
	Noise threshold	5%
	Distance measure	Euclidean distance
	Population size	300
HW-AOBOA	Maximum iterations	100
	Worker bee ratio	80%
	Drone bee ratio	20%
	Fitness objective	Minimum energy consumption
	Number of hives	10
	Bees per hive	30
	Fertilization ratio	30%
	Fading factor	20%
	Swarming threshold	60%
	Threshold value	0.2
T-Closeness	Distance metric	EMD

the traditional BA has 4724 kb and 4857 kb, ChaCha20 has 4831 kb and 4942 kb, and TF has 5785 kb and 5768 kb during encryption and decryption. From the results, the pro-

posed DBC-BA significantly reduces memory usage in encryption and decryption. Fig. 6 shows a throughput comparison of the suggested DBC-BA with existing algorithms. The proposed DBC-BA achieves a maximum throughput of 7326 kbps, and the existing algorithms, such as classical BA, ChaCha20, and TF attains 5684 kbps, 4487 kbps, and 3658 kbps, respectively. Fig. 7 presents the comparison of the proposed DBC-BA with existing algorithms in terms of security level. The proposed DBC-BA outperforms the current algorithms with the highest security level of 98.87%, and conventional BA, ChaCha20, and TF attain 95.68%, 94.19%, and 92.36%, demonstrating its greater ability to guarantee data integrity and confidentiality.

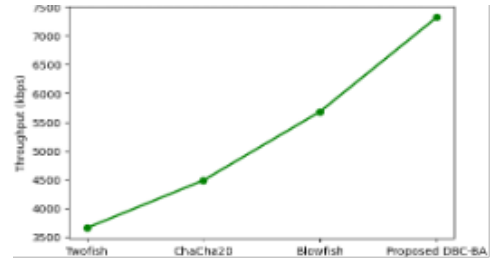


Fig. 6. Comparison of throughput.

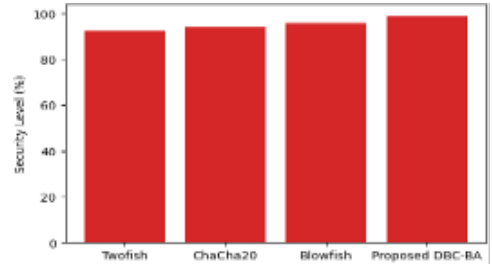


Fig. 7. Comparison of security level.

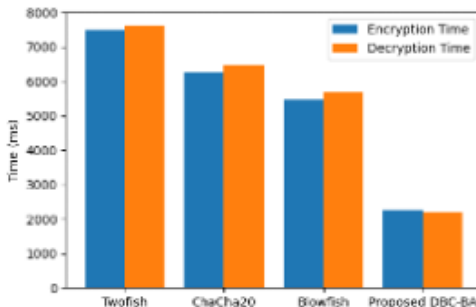


Fig. 4. Comparison of encryption and decryption time.

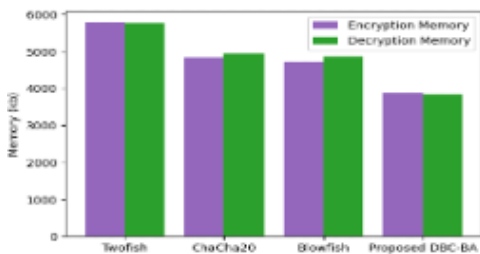


Fig. 5. Comparison of memory usage on encryption and decryption.

Overall, the proposed DBC-BA offers superior encryption performance, attaining minimum processing time, reduced

memory usage, and higher throughput and security. This improvement is attributed to the DBC-based subkeys generation, which ensures faster and secure data handling during data migration on the cloud.

4.1.2 Performance comparison of load balancing

The performance of the suggested LPF-DBSCAN algorithm is evaluated against other algorithms, including k-means clustering, CDEC (Contrastive Deep Embedded Clustering), and traditional DBSCAN, in terms of computational complexity, energy consumption, packet loss rate, throughput, execution time, network lifetime, delay, clustering accuracy, and cluster stability.

Fig. 8 presents the comparison of energy consumption for the proposed LPF-DBSCAN with existing algorithms. The LPF-DBSCAN achieves a minimum energy consumption of 41.25 J, and the DBSCAN, CDEC and K-means attain 47.65 J, 48.92J, and 51.21 J, respectively. Fig.

9 presents the throughput comparison of LPF-DBSCAN with DBSCAN, CDEC and k-means. The LPF-DBSCAN achieves a maximum throughput of 4215 kbps, and the DBSCAN, CDEC, and K-means attain 3565 kbps, 3297 kbps, and 2451 kbps, respectively. Fig. 10 presents the comparison of the proposed LPF-DBSCAN with existing algorithms in terms of execution time, delay, and network lifetime. The LPF-DBSCAN gains a minimum of 7584 ms of execution time, whereas the classical DBSCAN, CDEC and K-means gain 13547 ms, 15129 ms, and 17658 ms, respectively. When considering network lifetime, the presented approach maintains a consistently maximum network lifetime of 13685 ms. whereas DBSCAN exhibits a network lifetime of 9545 ms, CDEC exhibits a network lifetime of 8972 ms, and K-means exhibits a minimum network lifetime of 6698 ms. This reflects the strength of the suggested approach in sustaining network stability while the network is being scaled. Furthermore, the LPF-DBSCAN has a minimum delay of 3145 ms, compared to classical DBSCAN, CDEC and K-means, which have 4365 ms, 4825 ms, and 5968 ms, respectively. These results illustrate the robustness of the suggested LPF-DBSCAN and reflect its efficiency for load balancing in cloud data migration.

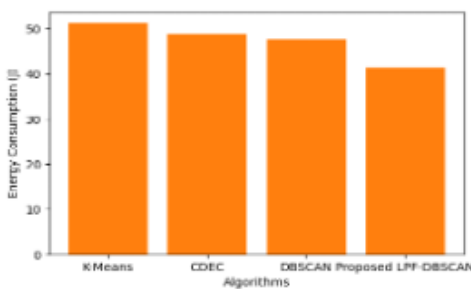


Fig. 8. Comparison of energy consumption.

Fig. 11 illustrates the cluster sta-

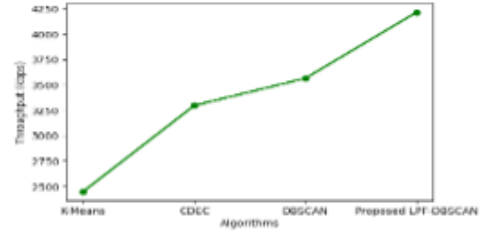


Fig. 9. Comparison of throughput.

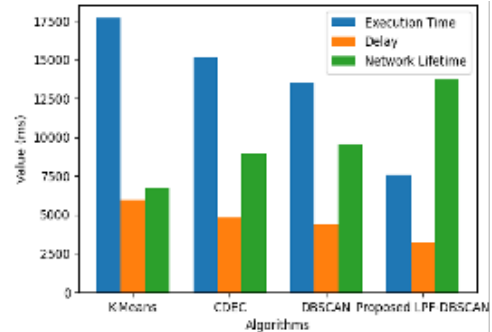


Fig. 10. Comparison of execution time, delay and network lifetime.

bility of the proposed LPF-DBSCAN in comparison to existing algorithms. LPF-DBSCAN exhibits maximum stability of 124.32, while DBSCAN, CDEC, and K-means achieve 92.45, 91.82 and 76.65, respectively. Classical DBSCAN generates unstable clusters, which rapidly collapse due to fixed parameters, and K-means clustering is unable to produce robust clusters due to its computationally demanding nature, leading to lower stability. The proposed method employs LPF-based density estimation, with a focus on robust cluster formation, resulting in improved stability and suitability in dynamic networks.

Fig. 12 presents a comparison of the proposed LPF-DBSCAN with traditional DBSCAN, CDEC, and K-means methods, based on clustering accuracy and packet loss rate. The LPF-DBSCAN gains the highest clustering accuracy of 98.78%, while classical DBSCAN, CDEC, and K-

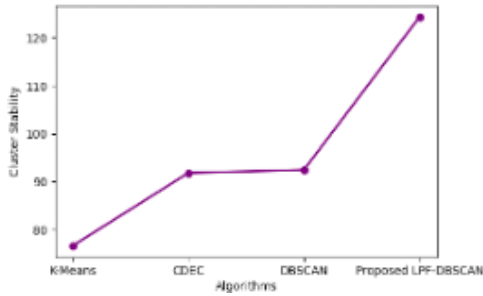


Fig. 11. Comparison of cluster stability.

means gain 94.87%, 94.21%, and 94.78%, respectively. Additionally, the proposed algorithm achieves minimum packet loss rate of 7.568%. In contrast, the traditional DBSCAN, CDEC, and K-means record higher packet loss rates of 11.45%, 12.43%, and 15.54%, respectively. These findings indicate that the proposed method is more efficient in data balancing, optimizing clustering accuracy while reducing packet loss, even with an increase in network size.

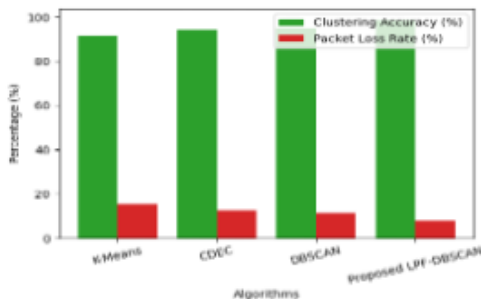


Fig. 12. Comparison of clustering accuracy and packet loss rate.

From the results, the proposed LPF-DBSCAN demonstrates superior performance in data load balancing, achieving the minimum execution time, delay, packet loss, and energy consumption, while also yielding greater throughput, network lifetime, clustering stability, and accuracy.

4.1.3 Performance comparison of vm privacy preservation

The effectiveness of the suggested T-Closeness Anonymity is contrasted with L-Diversity, Differential Privacy (DP), and K-Anonymization in terms of information gain and anonymity loss.

Fig. 13 shows the comparison of the T-Closeness with L-Diversity, K-Anonymization, and DP in terms of information gain and anonymity loss. The T-Closeness achieves the maximum information gain of 87.23% and minimum anonymity loss of 7.65%. In comparison, L-Diversity achieves an information gain of 81.54%, K-Anonymization achieves an information gain of 75.23%, and DP achieves an information gain of 71.42%, with anonymity losses of 13.54% 19.65%, and 21.58%, respectively. From these results, T-Closeness outperforms the other methods and offers stronger protection of sensitive VM information.

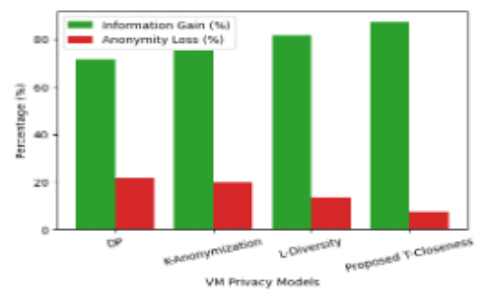


Fig. 13. Comparison of information gain and anonymity loss.

4.1.4 Performance comparison of vm selection and data migration

The performance of the proposed HW-AOBOA is compared with existing algorithms such as AOBOA, ACO, and Hippopotamus Optimization (HO) in terms of energy consumption, resource utilization, memory utilization, CPU utilization, load, response time, and delay.

The energy usage comparison between the suggested HW-AOBOA and existing algorithms is shown in Fig. 14. The proposed HW-AOBOA outperforms existing algorithms by achieving a minimum energy consumption of 38.56 J. In comparison, the AOBOA, ACO, and HO attain 45.84 J, 53.65 J, and 59.87 J, respectively. The reduced energy consumption in VM selection is directly influenced by optimized clustering in LPF-DBSCAN, which reduces computational overhead during allocation. The suggested approach decreased the energy required for VM selection and large-scale data movement by creating the most effective energy model.

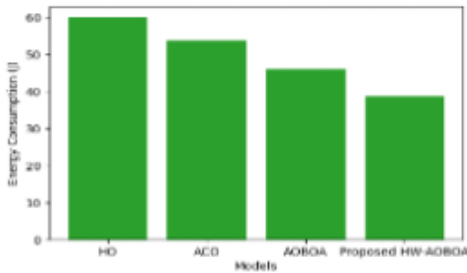


Fig. 14. Comparison of energy consumption.

Fig. 15 shows a comparison of resource, memory, and CPU utilization for the proposed HW-AOBOA, AOBOA, ACO, and HO. The HW-AOBOA achieves the maximum resource utilization of 89.48%, compared to 76.78%, 64.56%, and 62.57% for AOBOA, ACO, and HO, illustrating more effective resource management. Furthermore, the proposed algorithm requires a minimum memory utilization of 63.59%, compared to 71.23%, 78.65%, and 79.52% for the AOBOA, ACO, and HO, indicating optimized memory usage and reduced overhead. Similarly, the proposed HW-AOBOA achieves a minimum CPU utilization of 71.22%, compared to 79.84%, 83.56%, and 85.91% for

the AOBOA, ACO, and HO, respectively. According to the results, the suggested HW-AOBOA performs better than the AOBOA, ACO, and HO in terms of CPU, memory, and resource usage, ensuring balanced and efficient VM selection and data migration.

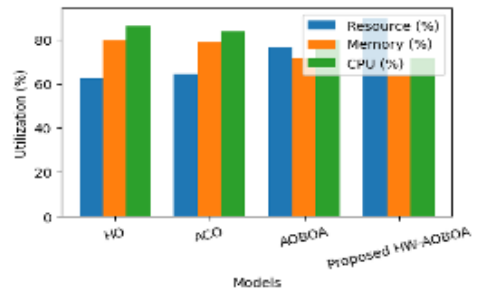


Fig. 15. Comparison of resource, memory, and CPU utilization.

The load comparison between the proposed method with AOBOA, ACO, and HO is displayed in Fig. 16. The suggested HW-AOBOA achieves the lowest load of 3564 kbps, compared to 5678 kbps, 8323 kbps, and 9259 kbps for the AOBOA, ACO, and HO, respectively. This reduction in load demonstrates that the HW-AOBOA manages data migration more efficiently, resulting in smoother and faster data migration. Fig. 17 shows the comparison of response time and delay for the proposed HW-AOBOA with AOBOA, ACO, and HO. The proposed algorithm achieves the lowest response time of 2745 ms, while AOBOA, ACO, and HO attain 4587 ms, 6359 ms, and 7252 ms, respectively. Moreover, the proposed approach achieves a minimum delay of 1857 ms, while the AOBOA, ACO, and HO attain delays of 3695 ms, 5847 ms, and 6840 ms, respectively. By reducing response time and delay, these results demonstrate the excellent performance of the HW-AOBOA, which enhances the overall effectiveness of VM selection and data migration.

tion. Overall, the proposed HW-AOBOA consistently surpasses AOBOA, ACO, and HO across all performance metrics, due to its integration of the HW principle with AOBOA. This contribution enhances energy efficiency and resource utilization, and minimizes load and delay, thereby ensuring faster, cost-efficient, and more reliable VM selection and cloud data migration.

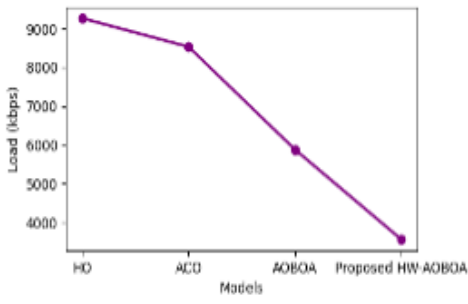


Fig. 16. Comparison of load.

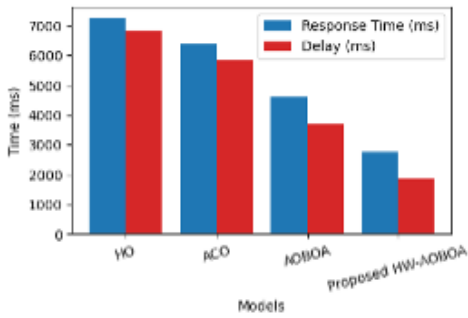


Fig. 17. Comparison of response time and delay.

4.2 Statistical analysis

To validate the reliability and consistency of the proposed framework, the experiments were independently repeated for $K = 5$ runs under identical simulation settings. The mean, standard deviation (SD) values, and 95% confidence intervals (CI) were computed.

The statistical analysis results presented in Table 2 demonstrate the stabil-

ity and consistency of the proposed framework across five independent runs. The proposed DBC-BA achieved an average encryption time of 2255.6 ± 7.1 ms, throughput of 7326.0 ± 20.5 kbps, and security level of $98.87 \pm 0.06\%$, indicating efficient and secure encryption performance with very low variation among repeated executions. Similarly, the proposed LPF-DBSCAN achieved an average clustering accuracy of $98.78 \pm 0.05\%$, packet loss rate of $7.57 \pm 0.04\%$, and cluster stability of 124.32 ± 0.24 , demonstrating robust and reliable load balancing performance under varying workload conditions. Furthermore, the proposed T-Closeness anonymity model attained an average information gain of $87.24 \pm 0.05\%$ and anonymity loss of $7.66 \pm 0.04\%$, confirming stable privacy preservation performance with minimal information leakage. In addition, the proposed HW-AOBOA achieved an average energy consumption of 38.57 ± 0.06 J, resource utilization of $89.49 \pm 0.06\%$, and delay of 1857 ± 4.9 ms, illustrating stable VM selection and efficient data migration performance. Overall, the low SD and narrow CI intervals confirm that the proposed secure cloud data migration framework exhibits high statistical stability and robustness across multiple independent runs.

4.3 Computational complexity analysis

This section evaluates the computational complexity of the proposed framework.

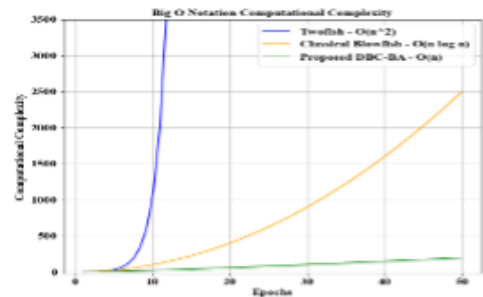
Fig. 18 presents the computational complexity comparison of the proposed DBC-BA with existing algorithms. The proposed DBC-BA achieves a linear complexity of $O(n)$, whereas the classical BA exhibits $O(n \log n)$ complexity and TF demonstrates higher exponential growth complexity $O(n^2)$. As the number of

Table 2. Statistical analysis of proposed framework.

Proposed Model	Run	Encryption Time	Throughput	Security level
DBC-BA	1	2264	7298	98.79
	2	2249	7341	98.91
	3	2256	7326	98.87
	4	2261	7318	98.83
	5	2248	7347	98.95
	Mean $\pm$ SD	2255.6 $\pm$ 7.1	7326.0 $\pm$ 20.5	98.87 $\pm$ 0.06
	95% CI	2255.6 $\pm$ 8.8	7326.0 $\pm$ 25.4	98.87 $\pm$ 0.07
LPF-DBSCAN	Run	Clustering Accuracy	Packet Loss Rate	Cluster Stability
	1	98.71	7.63	123.94
	2	98.84	7.52	124.58
	3	98.78	7.52	124.32
	4	98.81	7.54	124.47
	5	98.76	7.59	124.29
	Mean $\pm$ SD	98.78 $\pm$ 0.05	7.57 $\pm$ 0.04	124.32 $\pm$ 0.24
	95% CI	98.78 $\pm$ 0.06	7.57 $\pm$ 0.05	124.32 $\pm$ 0.30
T-Closeness Anonymity	Run	Information Gain	Anonymity Loss	
	1	87.18	7.71	
	2	87.31	7.61	
	3	87.23	7.66	
	4	87.27	7.63	
	5	87.19	7.69	
	Mean $\pm$ SD	87.24 $\pm$ 0.05	7.66 $\pm$ 0.04	
	95% CI	87.24 $\pm$ 0.06	7.66 $\pm$ 0.05	
HW-AOBOA	Run	Energy Consumption	Resource Utilization	Delay
	1	38.64	89.41	1864
	2	38.49	89.56	1851
	3	38.57	89.49	1857
	4	38.61	89.52	1855
	5	38.54	89.47	1860
	Mean $\pm$ SD	38.57 $\pm$ 0.06	89.49 $\pm$ 0.06	1857 $\pm$ 4.9
	95% CI	38.57 $\pm$ 0.07	89.49 $\pm$ 0.07	1857 $\pm$ 6.1

epochs increases, the computational cost of existing algorithms increases rapidly, while the proposed DBC-BA maintains stable and scalable performance with lower processing overhead, enabling faster encryption and decryption for secure cloud data migration.

Fig. 19 illustrates the computational complexity comparison of the proposed LPF-DBSCAN existing algorithms. The results demonstrate that the proposed LPF-DBSCAN achieves lower computational complexity with a near-linear growth rate of $O(n)$, whereas the traditional DBSCAN and k-means exhibits $O(n \log n)$ and

**Fig. 18.** Computational complexity comparison of DBC-BA with existing algorithms.

$O(n^2)$ complexity. The proposed LPF-DBSCAN provides better scalability, faster execution, and improved suitability for dy-

dynamic cloud load balancing and secure data migration applications.

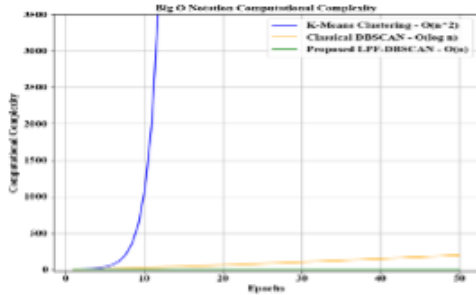


Fig. 19. Computational complexity comparison of LPF-DBSCAN with existing algorithms.

The computational complexity comparison of the proposed HW-AOBOA with existing optimization algorithms is illustrated in Fig. 20. From the graph, it is observed that the proposed HW-AOBOA achieves lower computational complexity with a near-linear growth rate of ($O(n)$). The reduced complexity enables faster optimization with lower processing overhead during VM selection and migration.

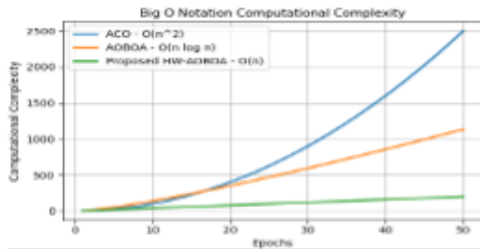


Fig. 20. Computational complexity comparison of HW-AOBOA with existing algorithms.

Fig. 21 presents the convergence analysis of the proposed HW-AOBOA in comparison with AOBOA and ACO algorithms in terms of fitness value (energy consumption). The proposed HW-AOBOA converges faster and achieves a more stable optimal solution compared to the existing algorithms. Initially, the fitness values are high for all methods, but the proposed

HW-AOBOA rapidly decreases the fitness value within fewer iterations and stabilizes earlier than AOBOA and ACO. The improved convergence performance is mainly due to the integration of the HW principle into AOBOA, which enhances solution diversity and improves the global search capability, thereby preventing premature convergence and enhancing optimization performance for energy-efficient VM selection in secure cloud data migration.

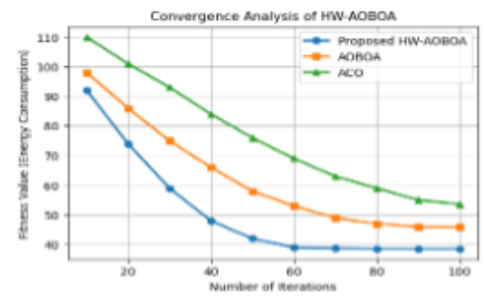


Fig. 21. Convergence analysis of HW-AOBOA with existing algorithms.

4.4 Security analysis of the DBC-BA

The security performance of the proposed DBC-BA is analyzed in terms of confidentiality, attack resistance, key sensitivity, and secure key generation capability. The proposed method enhances the conventional BA by integrating DBC into the subkey generation process, thereby improving both encryption security and processing efficiency during cloud data migration. The proposed DBC-BA generates dynamic subkeys using document-dependent chunking and SHA-256-based hash generation. Since the generated keys depend on the document structure, chunk index, and base encryption key, the proposed method increases key randomness and reduces the possibility of key prediction. This dynamic key generation mechanism enhances resistance against brute-force and key-related at-

tacks. In addition, the use of chunk-specific hash generation ensures that even minor modifications in the input data produce significantly different subkeys, thereby improving avalanche characteristics and encryption sensitivity. The proposed DBC-BA also strengthens confidentiality during cloud migration by encrypting data using a 16-round Feistel network structure with dynamically updated P-arrays and S-boxes. The integration of DBC into the subkey initialization process improves diffusion and confusion properties, making statistical analysis and differential attacks more difficult. Another important security advantage of the proposed DBC-BA is its resistance to unauthorized data access during migration. Since the encryption and decryption processes require the dynamically generated DBC-based subkeys, attackers cannot easily reconstruct the original plaintext without access to the exact chunk-based key generation mechanism. Therefore, the proposed DBC-BA provides secure and reliable encryption for cloud data migration environments.

4.5 Comparative analysis of proposed with state-of-the-art methods

This section presents a comparative analysis of the proposed model with previously published works [11-17] to demonstrate the efficiency of the proposed system. These previously published studies are reviewed in detail in the literature section. Table 3 compares the performance of the suggested method with that of state-of-the-art methods.

According to Table 3, the suggested framework outperforms the current state-of-the-art techniques [11-17] across all performance metrics. In the encryption phase, the proposed DBC-BA achieves the lowest encryption time of 2256 ms and the high-

est security level of 98.87%, outperforming [12, 15, 17]. Notably, [15] exhibits a higher processing time of 3459 ms, and [17] attains only 92% reliability. This indicates that the proposed DBC-BA ensures stronger data confidentiality with efficient processing and high throughput. Similarly, in the load balancing phase, LPF-DBSCAN achieves higher throughput, network lifetime, cluster stability, and accuracy while minimizing energy consumption, delay, execution time, and packet loss. For VM selection and migration, the HW-AOBOA achieves higher resource usage and minimum CPU and memory usage, load, energy consumption, delay, and response time, outperforming [11-14, 16, 17]. Unlike existing state-of-the-art methods that focus on only one or two data migration objectives, the proposed framework simultaneously addresses confidentiality, workload balancing, VM optimization, privacy preservation, and secure data migration within a single architecture. This integrated design improves migration reliability, scalability, and security.

5. Discussion

The experimental results demonstrate that the proposed multi-tiered framework effectively improves secure cloud data migration by integrating encryption, load balancing, VM selection, privacy preservation, and migration management into a unified architecture. In the encryption phase, the proposed DBC-BA achieved lower encryption and decryption time, reduced memory usage, and higher throughput. The proposed method also achieved a higher security level, indicating stronger confidentiality protection during migration. This improvement is mainly due to the integration of DBC during subkey generation, which accelerates key process-

Table 3. Performance comparison of the proposed method with state-of-the-art methods.

Ref. No	Contribution	Methods used	Performance results
[HTML]D3DFEE [11]	VM Allocation, Migration	Branch-and-Price, MDVMM	CPU usage-74%, Energy consumption-80 KW
[12]	Load Balancing, Encryption, Migration	JDBO, DEC	Resource usage-83.4%, Load-0.070
[HTML]D3DFEE [13]	Load Balancing, Resource Scheduling, Migration	ACO, GA, Fuzzy logic, K Means, Bi-LSTM, RNN	Delay-3267ms
[14]	VM Selection, Migration	Adaptive dynamic threshold, IEABF	Energy consumption-106 KWH
[HTML]D3DFEE [15]	Encryption	NLCA	Processing time-3459 ms
[16]	Migration	EAMA	Resource usage-84%, Energy consumption-8 KW/H
[HTML]D3DFEE [17]	Resource Allocation, Data Transmission	AFFA, SDN based CAC	Energy consumption -16, Reliability-92%
Proposed	Encryption, Load Balancing, VM Selection, VM Privacy Preservation, Migration	DBC-BA, LPF-DBSCAN, T-Closeness, HW-AOBOA	DBC-BA: Encryption time-2256ms, Memory Usage-3875 kb, Throughput- 7326 kbps, Security level- 98.87%

ing and enhances security during migration. However, computational overhead may still increase when handling large-scale cloud data with frequent migration requests. For load balancing, the LPF-DBSCAN achieved higher clustering accuracy, improved cluster stability, lower packet loss, and reduced delay compared to traditional DBSCAN and k-means algorithm. The LPF-based density estimation enabled efficient workload distribution in dynamic cloud environments. Nevertheless, minor packet loss and execution delay still occurred due to network fluctuations and varying workloads. Moreover, maintaining higher clustering stability may increase computational requirements during large-scale deployment. In the VM privacy preservation phase, the T-Closeness approach provided better protection for sensitive VM information by maximizing information gain and minimizing anonymity loss. However, increasing privacy constraints may reduce data utility and increase anonymization complexity for heterogeneous cloud datasets. The proposed HW-AOBOA also achieved superior VM selection and migration performance by improving resource utilization and reducing energy consumption, response time, CPU

usage, and migration delay. The adaptive queen-selection mechanism improved the global search capability and enabled efficient VM selection during migration. However, combining multiple optimization strategies increases algorithmic complexity, especially in distributed cloud infrastructures with large numbers of VMs and continuous migration requests.

Another important aspect of the proposed framework is the presence of trade-offs among conflicting performance objectives. In cloud environments, metrics such as energy consumption, delay, throughput, security, and resource utilization are inherently interdependent. Improving one objective may negatively impact another; for instance, stronger encryption and enhanced privacy preservation can increase computational overhead and latency, while aggressive load balancing may introduce additional processing cost. The proposed framework addresses these conflicting objectives through a coordinated multi-objective optimization strategy, which aims to balance security, efficiency, and resource utilization. This enables a more realistic and practical trade-off between performance metrics in dynamic cloud environments. Furthermore, the proposed framework was evaluated in

a simulated cloud environment, while real-world cloud systems may experience dynamic workloads, network instability, and scalability challenges. Therefore, real-time implementation and scalability validation are necessary before industrial deployment. In future work, lightweight encryption models and AI-driven adaptive migration strategies can be integrated to further improve scalability, security, and real-time deployment performance.

6. Conclusion

This study proposed a multi-tiered security framework for secure data migration in a cloud environment by integrating encryption, load balancing, VM selection, privacy preservation, and migration management into a unified architecture. Initially, user tasks were collected and assigned to the container. The proposed DBC-BA was used to encrypt the data to ensure confidentiality during migration. Load balancing was then carried out using the proposed LPF-DBSCAN algorithm. The proposed HW-AOBOA was employed for VM selection and data migration. In addition, T-Closeness was used to protect sensitive VM information. Experimental analysis showed that the proposed DBC-BA achieved the higher security level of 98.87%, the LPF-DBSCAN achieved a maximum clustering accuracy of 98.78%, and the HW-AOBOA achieves the maximum resource usage of 89.48%, indicating the effectiveness of the proposed framework. The proposed framework significantly outperformed the existing techniques, ensuring a smooth transition from on-premise systems to cloud environments. The study further demonstrated that secure cloud data migration requires not only robust encryption but also effective workload management, intelligent VM selection,

and privacy-aware migration mechanisms to achieve reliable and stable migration performance. Overall, the proposed framework demonstrates the importance of integrating multiple security and optimization techniques to achieve secure, efficient, and reliable cloud data migration.

References

- [1] S. Daniel, G. Olaoye, and U. Ejaz, Data migration in the cloud database: A review of vendor solutions and challenges, 2025.
- [2] A. Karunamurthy, M. Yuvaraj, J. Shahithya, and V. Thenmozhi, Cloud database: Empowering scalable and flexible data management, *Quing: International Journal of Innovative Research in Science and Engineering*, 2023.
- [3] J. R. Arunkumar, Study analysis of cloud security challenges and issues in cloud computing technologies, *Journal of Science, Computing and Engineering Research*, vol. 6, no. 8, pp. 6-10, 2023.
- [4] M. Azam, F. Nasim, J. Ahmad, and S. M. Bhatti, A security framework for data migration over the cloud, *Journal of Computing & Biomedical Informatics*, vol. 7, no. 2, 2024.
- [5] A. Alelyani, Cloud computing efficiency: Optimizing resource utilization, energy consumption, latency, availability, and reliability using intelligent algorithms, 2024.
- [6] P. Udayasankaran and S. J. J. Thangaraj, Energy efficient resource utilization and load balancing in virtual machines using prediction algorithms, *International Journal of Cognitive Computing in Engineering*, vol. 4, pp. 127-34, 2023.
- [7] A. Katal, S. Dahiya, and T. Choudhury, Energy efficiency in cloud computing data centers: A survey on software technologies, *Cluster Computing*, vol. 26, no. 3, pp. 1845-75, 2023.

- [8] O. Van Geet and D. Sickinger, *Best Practices Guide for Energy-Efficient Data Center Design*. Golden, CO, USA: National Renewable Energy Laboratory (NREL), 2024.
- [9] O. B. Johnson, J. Olamijuwon, Y. W. Weldegeorgise, and O. Soji, *Designing a comprehensive cloud migration framework for high-revenue financial services: A case study on efficiency and cost management*, *Open Access Research Journal of Science and Technology*, vol. 12, no. 2, pp. 58-69, 2024.
- [10] R. M. Haris, M. Barhamgi, A. Badawy, A. Nhlabatsi, and K. M. Khan, *Enhancing security and performance in live VM migration: A machine learning-driven framework with selective encryption for enhanced security and performance in cloud computing environments*, *Expert Systems*, vol. 42, no. 2, p. e13823, 2025.
- [11] N. K. Sharma, S. Bojjagani, Y. P. Reddy, M. Vivekanandan, J. Srinivasan, and A. K. Maurya, *A novel energy-efficient multi-dimensional virtual machines allocation and migration at the cloud data center*, *IEEE Access*, vol. 11, pp. 107480-95, 2023.
- [12] Y. H. Reddy, D. J. Rao, and V. Polepally, *Jaya dung beetle optimization-based load balancing and VM migration for cloud data security in DevOps*, *Computers and Electrical Engineering*, vol. 124, p. 110400, 2025.
- [13] M. G. Brahmam, *VMMISD: An efficient load balancing model for virtual machine migrations via fused metaheuristics with iterative security measures and deep learning optimizations*, *IEEE Access*, vol. 12, pp. 39351-74, 2024.
- [14] Z. Ma, D. Ma, M. Lv, and Y. Liu, *Virtual machine migration techniques for optimizing energy consumption in cloud data centers*, *IEEE Access*, vol. 11, pp. 86739-53, 2023.
- [15] 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. 91-9, 2021.
- [16] M. Ibrahim, M. Imran, F. Jamil, Y. J. Lee, and D. H. Kim, *EAMA: Efficient adaptive migration algorithm for cloud data centers (CDCs)*, *Symmetry*, vol. 13, no. 4, p. 690, 2021.
- [17] R. Mangalagowri and R. Venkataraman, *Ensure secured data transmission during virtual machine migration over cloud computing environment*, *International Journal of System Assurance Engineering and Management*, pp. 1-12, 2023.
- [18] S. S. Tyagi, *Enhancing the security of cloud data through encryption with AES and Fernet algorithm through convolutional neural networks (CNN)*, *International Journal of Computer Networks and Applications*, vol. 8, no. 4, pp. 288-99, 2021.