

A Hybrid Extreme Learning Machine And Symmetric DNA Cryptography For Preventing Information Leakage In Cloud Containers

Mr. Sivaramkumar P^{1*}, Dr. Omprakash Sharma²

¹Assistant Professor, School of Information Technology, SRM University Sikkim

²Professor, School of Information Technology, SRM University Sikkim

Abstract

Cloud Container refers to a lightweight, portable, and executable software package and run applications in the cloud. In cloud computing, containers are widely applied for deploying and managing applications since they offer portability, scalability, and consistency across different environments, better resource utilization than traditional virtual machines based data centers. Container cyber security is a primary concern as users and information grows in exponentially. Sensitive information is highly vulnerable to cyber attacks during sharing or communication. Several solutions are available to address the information leakage security in cloud container. However, confidentiality and data integrity remain the major concern. Therefore, a novel technique called cryptosystem called Hybridized EXtreme LEarning Machine based symmetric DNA cryptography (HEXLEM-DNA) is introduced. The primary objective of the proposed HEXLEM-DNA technique is to strengthen the security of data storage in cloud container by ensuring a higher level of confidentiality and data integrity. Proposed HEXLEM-DNA technique operates through four key stages such as user registration, data encryption, authentication, and decryption. Initially, users submit their details to the cloud container for registration. Once registered, the cloud container generates a unique ID and symmetric cipher session key using Blum-Micali algorithm for each user. After that, the cloud data owner encrypts sensitive data before uploading it to the cloud container by using DNA-based encryption. The encrypted data are uploaded into the cloud container for further processing. A cloud container acts as an isolated environment within the cloud infrastructure, helping to organize, manage, and protect the data. This approach significantly improves data confidentiality and adds a strong layer of security to cloud-based storage systems. When users access the data from cloud container, they must log in to the cloud container using these proper credentials for authentication. Authentication is performed using an extreme learning machine, which verifies whether the user is authorized or not. The cloud container grants the permission to the access the data for authorized user. Finally, the authorized user performs decryption with proper symmetric cipher session key using DNA-based decryption process. This process facilitates secure data storage and communication with enhanced confidentiality and integrity in the cloud environment. The proposed method was evaluated based on various performance metrics, including data confidentiality, data integrity, and authentication accuracy, processing time, across different volumes of data and cloud users. The experimental results demonstrate that proposed outperforms traditional methods by achieving higher and confidentiality, integrity accuracy along with reduced processing time

Keywords— Cloud container, secure data storage, symmetric cipher session key, Blum-Micali algorithm, DNA-based cryptosystem, extreme learning machine, authentication, decryption.

Introduction

Cloud computing has become an essential and rapidly rising area and offers an efficient data

storage and security, greater flexibility, improved transparency within organizations. A container is a self-contained software unit to run an application, such as the code, runtime environment, system tools, and libraries. Among various cloud service models, multi-tenant container clouds have recently gained popularity as a lightweight than the traditional virtual machine (VM)-based infrastructures. This container significantly reduces both startup times and resource consumption. In cloud container environments, data security is a major concern. Therefore, efficient cryptography model in cloud container is necessary to prevent unauthorized access or data leakage.

A novel Advanced Encryption Standard with Elliptic Curve Cryptography (AES-ECC) was designed in [1] to ensure the confidentiality of data stored in cloud container. The designed AES-ECC model successfully reduced processing delay. However, it did not achieve high integrity due to the lack of user authentication within the cloud container. An integration of Attribute-Based Encryption (ABE) with the Kubernetes labeling model was proposed in [2] to enhance data protection with reduced complexity. However, it did not incorporate artificial intelligence algorithms, limiting its ability to proactively strengthen data access security

A novel active defensive security framework was developed in [3] for a container-based cloud with deep reinforcement learning. However, framework was not successful in strengthening security with minimal time overhead. To addresses critical security challenges, a new deep learning-based model was designed in [4] for real-time anomaly detection within cloud container environments. Though the model reduces response time, it did not effectively utilize cryptographic techniques to further enhance security and protect data from intruders. A kernel isolation approaches were developed in [5] for guaranteeing the container security. However, the intelligent security frameworks were not employed to mitigate emerging threats in containerized systems. An advanced and unified policy management approach called KUBEAEGIS was introduced in [6] to manage the heterogeneous security policies within the container environment. However, the method was not effective in improving confidentiality and integrity level. An innovative and automated solution was specifically designed in [7] to deal with the critical security challenge within container environments. However, it did not strengthen consistent security. A kernel isolation framework was designed in [8] aimed to achieves security guarantees with less overhead. However, the framework was not efficient to obtain promising security guarantees in the container environments. A groundbreaking approach was designed in [9] to honeypot management within cybersecurity to considerably enhance the effectiveness of threat detection. However, learning- or rule-based systems were not implemented for selective storage and analysis while preserving data privacy. An enhanced kernel SELinux framework was developed in [10] to enhance security of the container with minimal latency. However, the system's scalability remained unaddressed among various security challenges.

A Dynamic Measurement approach was developed in [11] for more complex security and trust challenges. However, it did not focus on trust measurement of container behavior and protection of container user privacy. The file I/O stacks of the representative secure container runtimes were analyzed in [12]. However, it did not conduct security evaluations for secure container runtimes to trade-off between security and overall system performance. A flexible and lightweight mitigation approach was introduced in [13] for Distributed DDOS attacks within the container-based cloud environments. However, it did not make the efforts to develop machine learning-aided container platforms. A new and rigorous container image vulnerability evaluation system was developed in [14] to ensure the potential security risks. However, the system does not focus on optimizing resource consumption, especially in large-

scale cloud environments and various container security solutions. An accelerated particle swarm optimization strategy was designed in [15] to minimize the service response time and improve the resource utilization. The strategy reduces the service transmission overhead. However, it did not optimize key performance parameters such as average waiting time, cloud throughput, and average response time.

KEY CONTRIBUTIONS

A novel HEXLEM-DNA technique is introduced to overcome the existing methods with the following new contributions.

A HEXLEM-DNA technique is developed for the security of data storage and access control in cloud container based on the integration of a symmetric DNA cryptography and extreme learning machine.

To enhance the confidentiality and integrity, DNA-based cryptography model is designed in HEXLEM-DNA technique. The cloud container generates a symmetric cipher session key for each user using Blum–Micali algorithm. Afterward, the cloud data owner encrypts data based on four DNA building blocks before uploading to the container.

To improve the accuracy of authentication accuracy, an extreme learning machine is developed to classify the authorized or unauthorized user by using Forbes index results.

To increase data integrity, a DNA-based decryption algorithm is introduced for data decryption for protecting the data from the unauthorized user in the cloud container.

Finally, comprehensive evaluations are carried out to estimate the performance of the HEXLEM-DNA technique and other techniques along with the various metrics.

Outline of paper

The paper is organized into six sections as follows: Section 2 presents a review of related work. Section 3 introduces the proposed HEXLEM-DNA technique along with a clear architectural diagram. Section 4 outlines the experimental setup, including a description of the dataset used. In Section 5, the performance of the proposed HEXLEM-DNA technique is evaluated and compared with existing techniques using various metrics. Finally, Section 6 concludes the paper.

2. RELATED WORKS

A novel, dynamic, AppArmor profile generator was developed in [16] for improving container security. However, the model was unable to demonstrate enhanced security with higher confidentiality. An automated, robust security method was designed in [17] for runtime security to verify the integrity of container. However, there was a lack of effective method to enhance the security. A deep learning approach was presented in [18] using measurements and trace-based analysis within the container environment, which effectively reduces system overhead. However, it did not achieve a high level of data integrity. A performance profiling-based proactive security model was designed in [19] to minimize the response time and overhead. However, it failed to investigate static policy analysis to identify performance various types of attacks. A Deep Dive Into Network policy was designed in [20] to guarantee the security enhancements with higher throughput. However, machine learning techniques were not developed for threat detection to handle complex interactions.

A comprehensive analysis of potential security risks and vulnerabilities was conducted in [21] to safeguard systems against evolving threats. However, advanced automation techniques were not explored to enhance the vulnerability analysis. Various data security

measures were employed in [22] within cloud environments related to data confidentiality and efficient resource distribution among users. However, a novel cryptographic model was not implemented to achieve enhanced security. Container-based applications were analyzed in [23] with respect to availability, scalability, and security in a cloud container environment. However, the analysis involved higher operational complexity. An extended Berkeley Packet Filter was developed in [24] for efficient access control within Docker containers and providing enhanced security. The issue of maintaining high confidentiality during secure data sharing was not sufficiently addressed. A container-based time synchronization model (CTSM) was designed in [25] to minimize the overhead and increases the throughput. However, the delay was not reduced.

A novel container-based complete cyber deception honeynet framework was developed in [26] with the aim of protecting the networks and capture deep-level attacks. However, the framework was found to be inefficient in identifying authorized users, to improve overall efficiency. A forecasted resource-evaluating scheduler (FoRES) was developed in [27] to ensure the resource efficiency and service reliability. However, security issues remained unaddressed. A policy-based data-audit control system was developed in [28] for container migration with minimal time consumption. However, it was unable to conduct verification in a very compact and short-range environment. A quantum random number-based genetic algorithm was designed in [29] for cloud-based storage encryption. However, the cloud-based storage was not effectively identified the various cryptographic attacks. A new EaaS architecture was designed in [30] to enhance the throughput by resolving resource allocations utilizing the cloud nodes. However, it failed to explore integrating advanced technologies to enhance the architecture's security and efficiency.

3. PROPOSAL METHODOLOGY

Cloud Computing (CC) is a rapidly developing platform that offers numerous benefits such as scalable resources, large storage capacity, and reduced operational costs. With the increasing accessibility of cloud containers, the efficiency and flexibility of cloud-based systems have further improved. The growth of big data has directs to a significant growth in data volume and complexity, demanding robust infrastructures to manage and secure information effectively. A cloud container, integrated with dynamic networking is widely used for managing big data workloads. As a result, several methods have been proposed to ensure secure data transmission and reliable isolation within containerized cloud environments. However, secured data storage in the cloud containers plays a major role to preserve sensitive data from unauthorized attacks. Based on the motivation, a novel technique called is introduced to ensure data confidentiality and integrity.

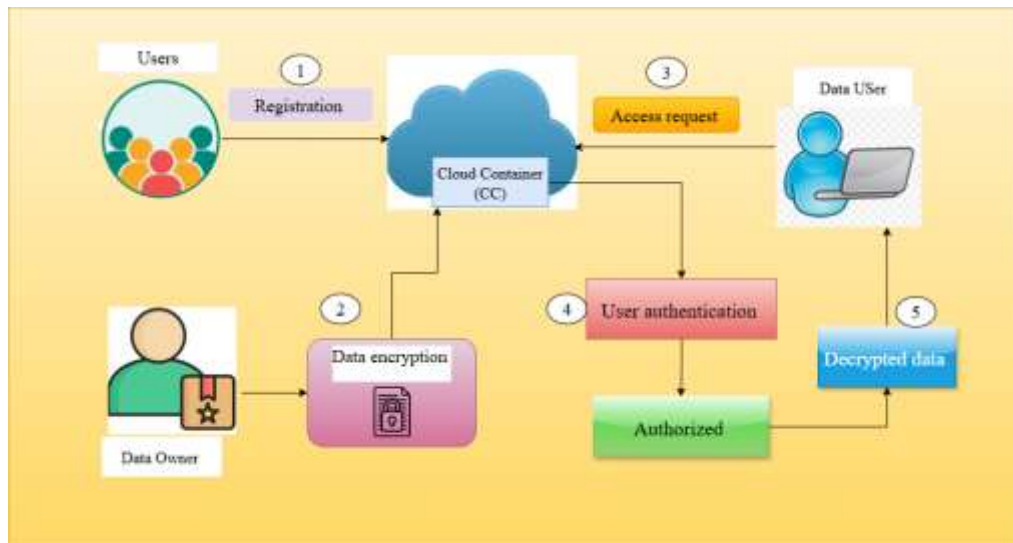


Figure 1 architecture of proposed

Figure 1 depicts the architecture diagram of the proposed method to provide efficient secure data storage in the cloud container. The cloud container architecture consists of three entities namely users, data owner and container. The users $CU = \{U_1, U_2, \dots, U_n\}$ and the data owner 'DO' who dynamically generates the data $D = \{D_1, D_2, \dots, D_n\}$ and stored into container for further processing. To begin with, both cloud users and data owners complete a registration process to enable data storage for future processing. After registration, the cloud container creates a unique key pair for each registered user. Once keys are generated, data owners encrypt their data using DNA encoding and upload the encrypted version to the cloud container. When a user requests access, the container employs an extreme learning machine to verify the user's authenticity. Access is granted to authorize users and denied access for unauthorized users. Authorized users then decrypt the data, ensuring its confidentiality and integrity.

3.1 User registration

The initial phase of the proposed system involves a user registration process. Before a user uploads data to the cloud or retrieve information from it, they are required to submit their credentials and relevant identification details to the cloud service. This registration ensures that each user is individually identified and authorized, creating a secure environment for managing data access. The cloud container maintains a record of these registered users for authentication during data transmission.

During the registration phase, the user submits essential information, including identity credentials, personal data. This step is crucial for the cloud container, as it enables the creation of secure user profiles and facilitates effective management of user accounts within the cloud infrastructure.

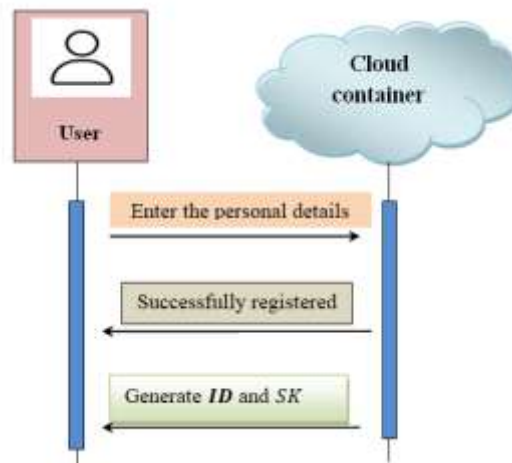


Figure 2 Flow process of new user registration

Figure 2 illustrates the detailed workflow for registering a new user within the cloud environment. To begin, the user is required to complete a registration form supplied by the cloud container. This form collects personal information such as the user's name, age, date of birth, and email address, which serves as a key element for authentication.

During registration, the user provides a valid email address that will act as their username in the cloud container. The cloud container then verifies whether this email ID is already in use by searching the existing database. If the submitted username is found to be duplicate, then the user is to select a unique email address. Once a unique username is confirmed, the container generates a symmetric cipher session key, and the registration process is completed by securely storing the new user credentials such as ID and key in the cloud container's database.

Key generation plays a significant role in establishing secure data communication within cloud environments. For each registered user, the cloud container creates a symmetric cipher session key. This symmetric cipher session key enables secure data exchange to encrypt and decrypt it, ensuring both privacy and data integrity.

The Blum–Micali algorithm is employed for symmetric cipher session key generation. A symmetric cipher session key refers to a cryptographic Pseudorandom Number Generator where the same secret key is utilized for both encrypting and decrypting data. In a DNA encoding, a session key plays a vital role and it generated for a single session and it valid only during that specific transaction.

In the key generation process, Blum–Micali algorithm is used for generating the symmetric cipher session key 'SCSK'. Let us consider the odd prime number 'P'

$$SCSK = G^x \bmod P \quad (1)$$

Where, G, x indicates a random number. In this way, user ID and symmetric cipher session key is generated for each registered users 'U_j'.

$$U_j \rightarrow (ID, SCSK) \quad (2)$$

The algorithm of user registration and key generation process is executed as follows,

Algorithm 1: user registration
Input: Number of cloud users $CU = \{U_1, U_2, \dots, U_n\}$
Output: User registration and key generation

```

Begin
\\ Registration phase
For each user 'U'
    Cloud container (CC) ask to enter their details
    Users enters their details and submit to the CC
    CC verify the availability of user name
If (submitted username is found to be duplicate)
    CC ask generate new username
End if
    CC sends 'SRM' to U'
For each registered user U'
    Generates ID and SCSK using (1) (2)
End for
End

```

The algorithm presents a different step of user registration and key generation process. Initially, the cloud container requests the user to provide personal information. Once the required details are submitted, the cloud container verifies whether the username already exists in its database. If a duplicate is found, the user is informed to select another username. Upon successful validation and acceptance of the unique credentials, the system confirms the registration by sending an acknowledgment message to the user. Followed by, Blum–Micali algorithm is applied to generate the symmetric cipher session key and ID.

3.2 Encryption

Encryption is the process of converting original data (i.e. plaintext) into unreadable form (i.e. ciphertext) to protect it from unauthorized access. The main advantage is to ensure data confidentiality and protect sensitive information during storage or transmission from unauthorized access. In secure cloud systems, data owner is the individual or organization uploading and managing sensitive data. Before storing the data into the cloud container, the data owner performs encryption to ensure confidentiality and integrity. Therefore, DNA cryptography is an emerging technique for encryption that utilizes the biological properties of DNA molecules (Deoxyribonucleic Acid) to perform cryptographic operations such as encryption and decryption.

DNA cryptography influences the inherent properties of DNA sequences and it utilizes the biological security layer to data protection than the other cryptographic methods. This DNA sequences are extremely complex and difficult to manipulate and it act as a highly secure medium for encryption.

A DNA structures consists of four building blocks called nucleotides along with the nitrogenous bases such as Adenine (A) Thymine (T), Cytosine (C), and Guanine (G) respectively. By using above said four blocks, input data is converted to cipher texts or nucleobase code.

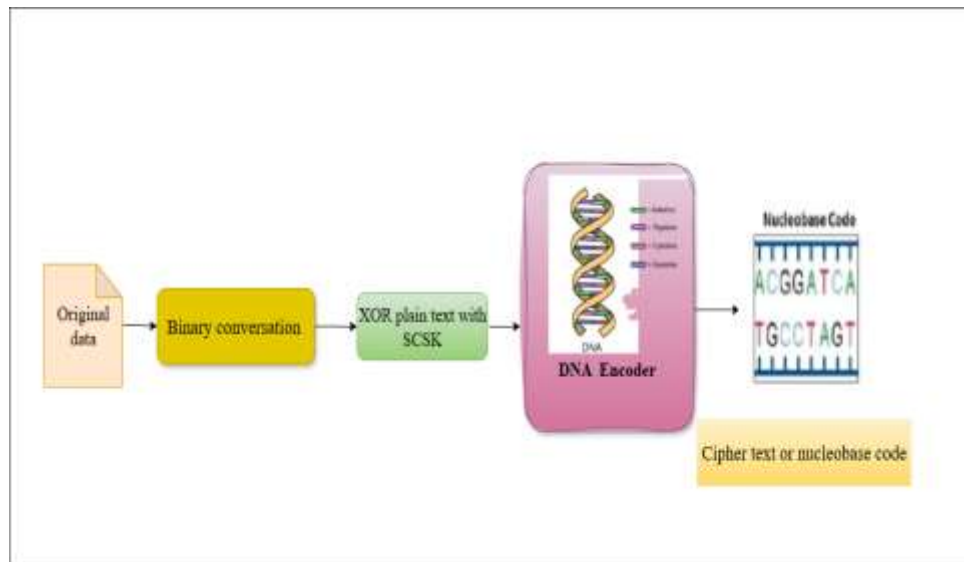


Figure 3 block diagram of the encryption

Figure 3 depicts the block diagram of the data encryption process to convert the input data into cipher text. Let us consider the number of owner data represented by $D_1, D_2, D_3, \dots D_n$. Before the uploading to cloud container, the data owner performs data encryption with the symmetric cipher session key (SCSK).

Initially, the input data is transformed into its binary form, where each character or symbol is represented using a sequence of '0's and '1's. This binary conversion is essential because DNA encoder operate on binary values. If the input is a string, it is first converted into its ASCII representation, and then each ASCII value is transformed into binary form.

$$BD = \text{Bin}(D_i) \quad (3)$$

Where, BD denotes a binary representation of data, Bin denotes a binary form of input data ' D_i '. This step ensures that the data is in a suitable format for further encryption. Followed by, the key 'SCSK' also converted into binary form 'BK'.

$$BK = \text{Bin}(\text{SCSK}_i) \quad (4)$$

After the getting the binary string, to ensure two binary lengths 'BR' and BRK are equal and aligned in even-length blocks (multiples of 2 bits). If the binary lengths are not equal, Zero padding is added to either the message or key to ensure their binary lengths for DNA encoding. The binary form of input and key are XORed bit by bit as follows,

$$Q = BD \oplus BK \quad (5)$$

Where, Q output of binary representation of data 'BD' XORed with the binary representation of key 'BK'. After that, the binary form of 'Q' is mapped into the nucleotide bases.

In order to obtain the cipher text, DNA sequences involve the four nucleotide bases such as Adenine (A) Thymine (T), Cytosine (C), and Guanine (G) as numerical values. Adenine (A) is assigned 0, T is 1, C is 2, and G is 3. Using this method, the sequence ACGT is encoded as 0123 in which each base is represented by unique binary digits as shown in table 1.

Table 1 representation of nucleotides

Four DNA components (DNA _b)	Numerical values	Binary representation
Adenine (A)	0	00
Thymine (T)	1	01
Cytosine (C)	2	10
Guanine (G)	3	11

According to table 1, the binary input is mapped to the four nucleotide bases such as A, T, C, and G based on their corresponding 2-bit binary values 'Q' observed from .

$C = \text{DNA}_b(Q)$ (6)

$$\text{DNA}_b(Q) = \begin{cases} 00 \rightarrow A \\ 01 \rightarrow T \\ 10 \rightarrow C \\ 11 \rightarrow G \end{cases} \quad (7)$$

Where, DNA_b denotes a DNA bases, Q indicates a binary values, C represents the cipher texts or nucleobase code. In this way, data encryption is performed and data owner uploads heir data to cloud container.

3.3 Authentication

Whenever a user wants to access the stored data, the cloud container employs an extreme learning machine (ELM) classifier to distinguish between authorized and unauthorized users. This classifier is a type of artificial neural network, specifically a feed-forward network, which consists of input layer, hidden layer and output layer. These layers work together to process the input data and make decisions regarding whether to grant or deny access based on learned patterns and behaviors. The advantage of extreme learning algorithm in authentication is to automatically learn huge amounts of input with minimal computational power and improving performance

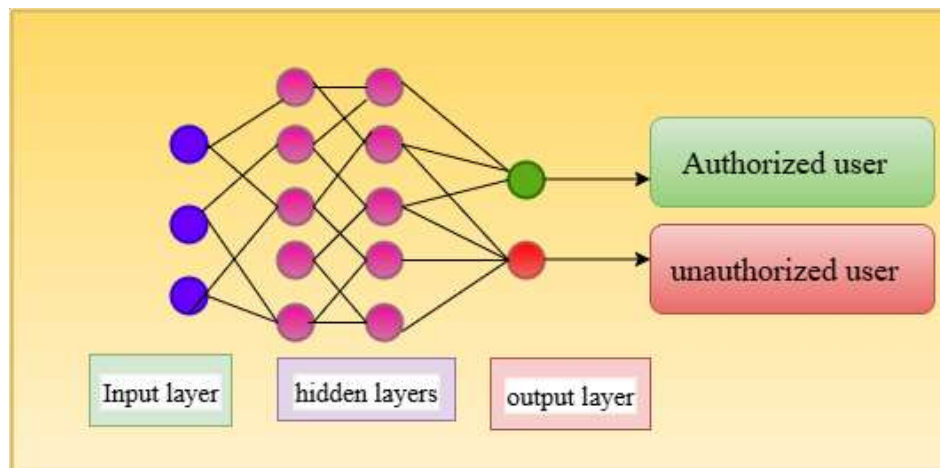


Figure 4 structures of Extreme learning machines

Figure 4 illustrates the arrangement of Extreme learning machines for classification of cloud users as authorized or unauthorized with a multiple layers of nodes. It includes different layers, namely input layer, hidden layers and output layer. The input layer receives the input

i.e. Cloud users who want to access the data. The layers located between the input and output layers are known as hidden layers. These layers consist of neurons (or nodes) that transform the input by passing it through weighted connections, enabling the network to learn complex patterns. Finally, the output layer provides the overall output.

Each layer includes a small individual unit called node or neuron. Neurons in each layer are connected to neurons in the next layer, and each connection has a weight. The connections between neurons are associated with weights that are adjusted during the training process. Initially, number of cloud users and their login activities is collected at an input layer. The activity of a neuron in each layer computes a linear function of the weighted sum of its inputs, summed by the bias term.

$$F = \sum_{i=1}^n U_i \beta_i + H \quad (8)$$

Where, F indicates an activity of neuron, U_i denotes a 'n' number of users and their activities, β_i synaptic weight, 'H' indicates a bias term that has the integer value of '1'.

The input is transferred into the hidden layer. With the input users, cloud container performs the user verification through the Forbes index. The Forbes index is a metric used to compare login activities of the users and the login credentials stored during registration. It measures similarity based on two sets. The mathematical formulation of the Forbes index coefficient is presented as follows.

$$FI = N * \frac{|LC \cap SC|}{|LC| |SC|} \quad (9)$$

Where, FI denotes a Forbes index results, N indicates a total number of elements in the set (the length of the vectors), LC denotes a user login credentials, SC stored credentials at the time of registration, $|LC \cap SC|$ denotes a mutual dependence between the two sets, $|LC|$ denotes a size of the sets 'LC', $|SC|$ denotes a size of the sets 'SC'. The Forbes index provides the output ranges between 0 and 1. The coefficient 'FI' attains its maximum value of '1' when the two sets are matched, meaning all elements matched exactly. If there is no overlap between the sets, the similarity score is 0, indicating no similarity.

The similarity coefficient values are given to the sigmoid activation function to analyze the results and provide the binary classification results as authentic or illegitimate users (i.e. attacks).

$$f(S) = \frac{1}{(1+\exp(-FC))} \quad (10)$$

Where, $f(S)$ denotes a sigmoid activation function, FI denotes a Forbes similarity. The output of the sigmoid function is lies between 0 and 1.

$$f(S) = \begin{cases} 1 ; & \text{attack or unauthorized} \\ 0 ; & \text{normal or authorized} \end{cases} \quad (11)$$

If the output value reaches 1, the user is classified as attacks or suspicious. Otherwise, they are classified as legitimate users or normal. The classification results are obtained at the output layer.

3.4 Decryption

After distinguishing between legitimate and unauthorized users, the cloud server grants access to normal users while denying access to those identified as illegitimate. This approach strengthens data integrity by preventing unauthorized access. Verified users then decrypt the ciphertext to retrieve the original data. The decryption process utilizes the DNA decoding cryptosystem and is carried out using the symmetric cipher session key associated with the authorized user.

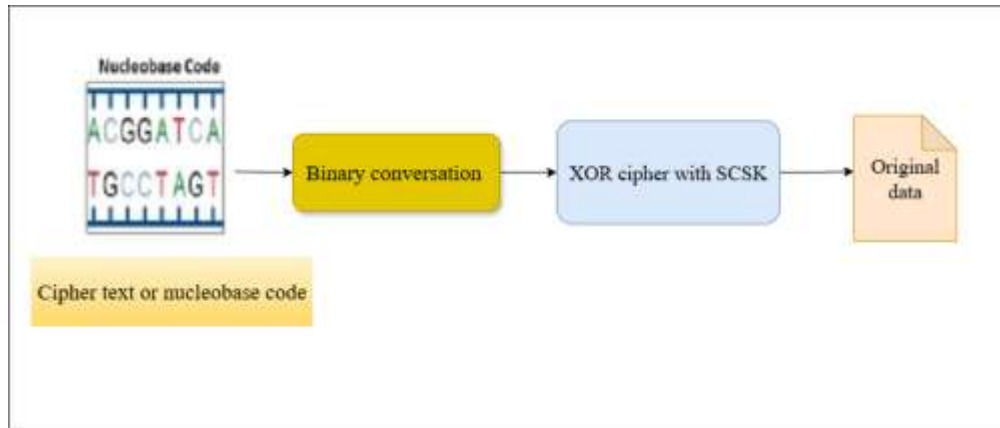


Figure 5 block diagram of the decryption

Figure 5 depicts the block diagram of the data decryption from the cloud container by converting cipher text into original input data. Let us consider the cipher text is 'C'. The authorized user performs data decryption with the symmetric cipher session key (SCSK). First input cipher text 'C' with DNA bases is converted into binary representation according to table 1. The binary representation of the cipher text 'Q (C)' is obtained as follows,

$$Q(C) = \begin{cases} A \rightarrow 00 \\ T \rightarrow 01 \\ C \rightarrow 10 \\ G \rightarrow 11 \end{cases} \quad (12)$$

These binary values are then XORed with the bits of the symmetric cipher session key (SCSK) to reconstruct the original plaintext in binary form.

$$D = Q(C) \oplus BK \quad (13)$$

Where, D denotes a original data, Q (C) binary representation of the cipher text 'Q (C)', BK denotes a bits of the symmetric cipher session key (SCSK). Finally, the binary of the input data is translated into its equivalent decimal or string characters to recover the original message. For each cipher text, the original data gets obtained through the decryption with symmetric key. In this way secure data storage and access control is performed in cloud container

Algorithm 2: Extreme Learning Machine Based Symmetric DNA Cryptography

Input: Number of cloud users $CU = \{U_1, U_2, \dots, U_n\}$, cloud owner 'CW' data $D_1, D_2, D_3, \dots, D_m$, cloud container 'CC', symmetric cipher session key (SCSK)

Output : Increase security of data transaction

```

Begin
Encryption
1: For each owner data 'Di'
2:   Convert binary representation 'BD' using (3)
3:   Convert SCSK into binary representation 'BK' using (4)
4:   Perform XOR operation 'Q' using (5)
5:   Map the binary input 'Q' into four DNA bases using (6) (7)
6:   Upload encrypted data 'C' into cloud container 'CC'
7: End for
Authentication
8: For each access request from user
9:   CC performs the user authorization
10: Collect the number of users in input layer
11: for each user ---hidden layer
12:   for each user login credentials
13:     Measure the similarity index using (9)
14:   End for
15: End for
16:   Apply sigmoid activation function using (10)
17: If ( $f(S) = 1$ ) then ---output layer
18:   User classified as attacks or suspicious
19: else
20:   User classified normal or not suspicious
21: End if
22: End for
Decryption
23: Authentic user perform decryption
24: for each DNA bases
25:   convert into binary form 'Q (C)' using (12)
26:   Perform XOR operation Q (C) and key BK (13)
27:   Transform binary into data or string
28:   Obtain the original data
29: End for
End

```

Algorithm 1 outlines a secure method for managing data storage and access control between cloud users and the container. Initially, cloud owner encrypt their data using four DNA bases to maintain confidentiality before uploading it to the cloud container, ensuring that unauthorized access is prevented. When access requests arrived, the cloud c uses an extreme learning machine Classifier to identify and verify the user authorization. The classifier receives the number of users as input in the input layer. To assess access authenticity, the Forbes index compares the user's login credentials and stored credentials in the container database. A sigmoid activation function processes this similarity, categorizing the user as either authorized or unauthorized (i.e. attacks). The authentication results are produced at the output layer. If authenticated, the users then decrypt the data and retrieve its original form through the DNA decoding process. This method effectively protects data privacy and integrity during both transmission and access in the cloud container environment.

EXPERIMENTAL SETUP

Experimental evaluation of proposed HEXLEM-DNA technique and existing AES-ECC [1] [1], ABE with Kubernetes [2] are implemented in using python programming language with ContainerCloudSim simulator. For the experimental consideration, the BETH Dataset is considered and taken from <https://www.kaggle.com/datasets/katehighnam/beth-dataset> for Real Cybersecurity Data for Anomaly Detection Research in cloud data centers by detecting the intruders or attacks. The BETH dataset consists of many labeled data folder including training, testing and validation. Among them, training data folder is considered for implementing on a major cloud provider. The dataset consists of 16 raw features and 763144 instances or data samples are taken as input to perform secure data storage in cloud container. The attributes description of dataset is listed in table 1

Table 1 dataset description

S.No	Attributes	Description
1	timestamp	Indicates the time an event or system call occurred..
2	processId	Unique ID of the process that triggered the event.
3	threadId	Thread ID within the process.
4	parentProcessId	The ID of the parent process that spawned this process.
5	userId	Identifier of the user executing the process
6	mountNamespace	Identifies the namespace of the process
7	processName	Name of executable/process. Used to identify suspicious or known malicious binaries.
8	hostName	Name of honeypot or host machine where the event occurred.
9	eventId	Unique identifier for the system event
10	eventName	Name of event that was triggered.
11	stackAddresses	Memory value addresses where the event occurred.
12	argsNum	Length of arguments passed to the event
13	returnValue	Return value of the event
14	args	List of arguments passed to the event
15	sus	Indicates whether the event is suspicious (based on behavior or signature) 1 = suspicious, 0 = benign
16	evil	Indicates malicious intent 1 = attack behavior, 0 = not malicious

4.1 Implementation results and discussions

The implementation of the HEXLEM-DNA technique is effectively analyzed to evaluate its effectiveness for preventing information leakage in cloud containers. The analysis focused on several key processes including registration, key generation, data encryption, authentication and classification.

First, User registration is the process to create an account or access credentials to a cloud container. In cloud container environments, secure user registration is crucial to ensure that only legitimate users access resources.



Figure 6 user registrations in cloud container

After registration, key generation is the process of creating unique ID and symmetric cipher session key using Blum–Micali algorithm used for encrypting and decrypting data to ensure confidentiality and integrity.



Figure 7 user id and symmetric cipher session key

After key generation, the data owner performs the data encryption using four SNA sequences such as Adenine (A) Thymine (T), Cytosine (C), and Guanine (G).

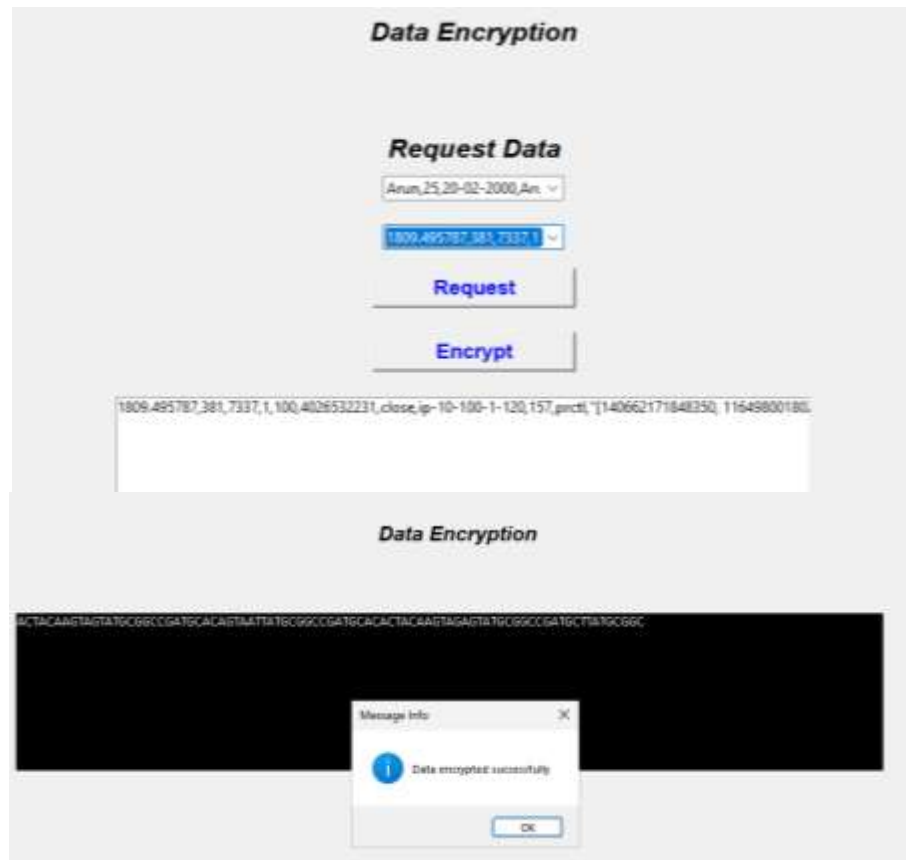


Figure 8 data encryption

The HEXLEM-DNA technique performs the User authentication using extreme learning machine for verifying the identity of a user before accessing the data from cloud container. It ensures that only authorized users access protected resources.

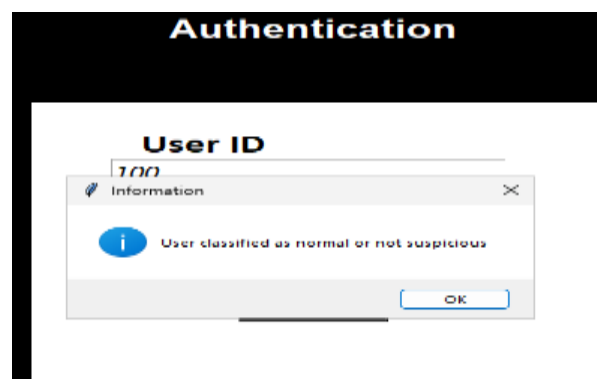


Figure 9 user authentication

Finally, HEXLEM-DNA technique performs decryption for converting encrypted data (ciphertext) back into its original, readable using a symmetric cipher session key. It is the reverse operation of encryption and allows authorized users to access the original

information from the cloud container.

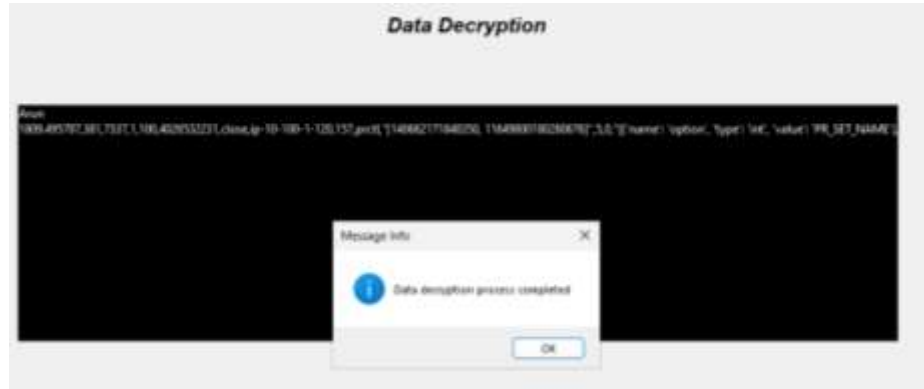


Figure 10 data decryption

Comparative Analysis and Results Discussion

The comparative result analysis of proposed HEXLEM-DNA and existing AES-ECC [1] [1], ABE with Kubernetes [2] are discussed in this section with different parameters such as Data confidentiality rate, data integrity, accuracy and computation time. The comparative performance analyses are done with the help of either table or graphical representation.

Impact of data confidentiality rate

Data confidentiality rate is calculated as the proportion of data accessed by authorized users relative to the total amount of data. It is evaluated using the following formula:

$$DCR = \sum_{i=1}^n \frac{DAA}{D_i} * 100 \quad (14)$$

Where, DCR denotes the data confidentiality rate measured as number of data 'D_i' and number of data accessed by authorized user 'DAA'. The confidentiality rate is measured in percentage (%).

Table 3 comparison of data confidentiality rate

Number of data instances	Data confidentiality rate (%)		
	Proposed HEXLEM-DNA	AES-ECC [1]	ABE with Kubernetes [2]
10000	96.85	94.21	92.14
20000	96.58	93.65	91.32
30000	97.02	93.74	91.05
40000	96.89	93.45	91.45
50000	97.12	93.65	91.03
60000	96.35	94.05	92.45
70000	97.33	94.22	92.34
80000	97.22	94.56	92.63
90000	97.65	94.33	92.1
100000	96.85	93.63	92.32

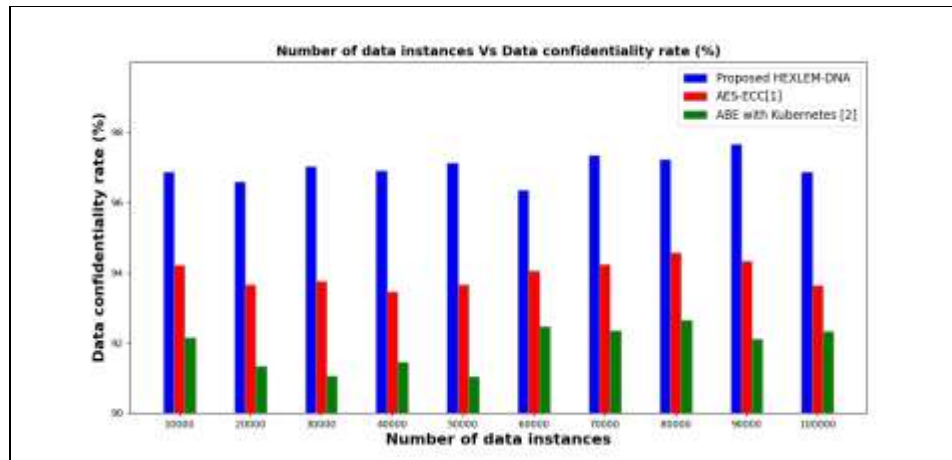


Figure 11 graphical analysis of data confidentiality rate

Figure 11 illustrates the graphical analysis of the data confidentiality rate versus volume of data using three methods HEXLEM-DNA and existing AES-ECC [1], ABE with Kubernetes [2]. The horizontal axis represents the number of instances, while the vertical axis shows the corresponding confidentiality rate. The results clearly indicate that the proposed HEXLEM-DNA approach achieves superior performance in maintaining data confidentiality compared to both baseline methods. For example, when tested with 10,000 data instances, the HEXLEM-DNA technique attained a confidentiality rate of 96.85%, whereas [1] and [2] achieved 94.21% and 92.14%, respectively. The observed results of HEXLEM-DNA are compared to the results of existing methods. The comparison results illustrate that the proposed HEXLEM-DNA increases the performance of confidentiality by 3% and 5% when compared to existing methods. Across ten experimental runs, the HEXLEM-DNA consistently delivered higher, showing an improvement of approximately 3% compared to [1] and 6% compared to [2]. This enhanced performance is primarily achieved due to the integration of the Hybridized extreme learning machine based symmetric DNA cryptography. The symmetric DNA cryptography is to strengthen the security of data storage in cloud container. Additionally, the extreme learning machine contributes to security by evaluating the relationship of user login credentials and categorizing users as either authorized or unauthorized. Only authorized user's received decrypted data, thereby improves the overall confidentiality.

Impact of Data integrity rate:

Integrity rate refers to the number of data items that remain unmodified by unauthorized users. It is quantitatively evaluated using the following formula:

$$DIR = \sum_{i=1}^n \frac{DNA}{D_i} * 100 \quad (15)$$

Where, DIR denotes a data integrity rate, n indicates the number of data instances that is measured in the unit of percentage (%).

Table 3 comparison of data integrity rate

Number of data instances	Data integrity rate (%)		
	Proposed HEXLEM-DNA	AES-ECC [1]	ABE with Kubernetes [2]

10000	95.86	93.89	91.36
20000	95.23	92.65	90.05
30000	96.05	92.56	90.45
40000	95.74	92.74	90.74
50000	96.65	92.65	90.36
60000	95.66	93.02	91.45
70000	96.41	93.45	91.03
80000	96.78	93.74	91.45
90000	96.13	93.65	91.41
100000	95.85	92.86	91.04

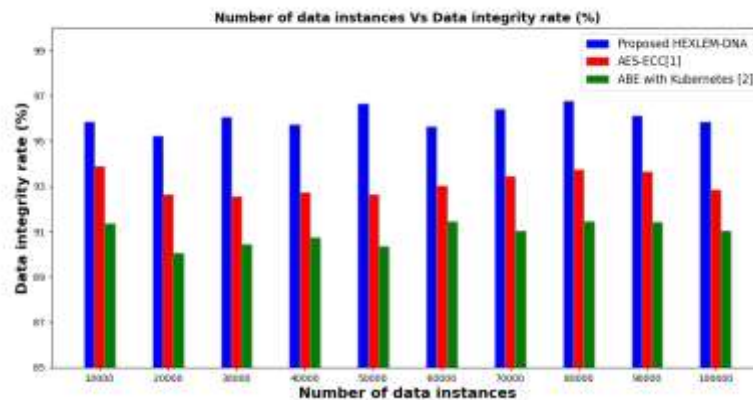


Figure 12 graphical analysis of data integrity rate

Figure 12 present the performance evaluation of the data integrity rate with respect to varying amounts of collected data, ranging from 10,000 to 100,000 records. The data integrity rate was measured using three different methods using HEXLEM-DNA and existing AES-ECC [1], ABE with Kubernetes [2]. In the graph, the horizontal axis represents the number of input data points, while the vertical axis displays the resulting data integrity rate. Among the three methods, the proposed HEXLEM-DNA technique consistently demonstrated improved performance in terms of data integrity. Let us consider the scenario with 10,000 data instances to compute the integrity. The integrity rate of the HEXLEM-DNA technique is 95.86%, while the integrity rates of the existing methods [1] and [2] are observed to be 93.89% and 91.36%, respectively. The observed result shows that the data integrity rate of HEXLEM-DNA technique is considerably increased by 2% and 5% when compared to [1] [2] respectively. Similarly, various data integrity rates were observed for each method with different inputs. The overall observed results of proposed model are compared to existing methods. The average value of comparison results confirms that the HEXLEM-DNA technique notably increases the data integrity rate by 3% and 7% compared to [1] [2] respectively. This improvement is achieved by using DNA cryptography for secure data encoding that utilizes the properties of DNA bases (A, T, C, G) to encode and secure digital data by using symmetric cipher session key generated by Blum–Micali algorithm. During the decoding process, the proper symmetric cipher session key is crucial to get the original data from the cloud

container. This process helps reduce data alteration or modification, ensuring high integrity and protecting sensitive information from potential attackers.

Impact of the authentication accuracy

Authentication accuracy (AA) represents the proportion of users correctly identified as authorized or attacks by the system to the total number of users. The formula for calculating the accuracy is given below,

$$AA = \left(\frac{TRP+TRN}{TRP+TRN+FLP+FLN} \right) * 100 \quad (16)$$

Where, AA denotes an authentication accuracy, true positive 'TRP' represents the legitimate users correctly authenticated, true negative 'TRN' illegitimate users correctly identified, false positives 'FLP' illegitimate users incorrectly authenticated, false negatives 'FLN' denotes a legitimate users incorrectly identified. The accuracy is measured in the unit of percentages (%).

Table 4 comparison of authentication accuracy

Number of users	Authentication accuracy (%)		
	Proposed HEXLEM-DNA	AES-ECC [1]	ABE with Kuberbetes [2]
1000	97.5	93	91
2000	97.23	92.56	90.63
3000	97.63	93.36	91.23
4000	97.05	92.05	90.41
5000	96.78	92.74	90.32
6000	97.03	92.96	90.74
7000	96.74	93.05	91.2
8000	97.63	92.56	90.36
9000	97.05	93.22	91.22
10000	96.89	92.36	90.36

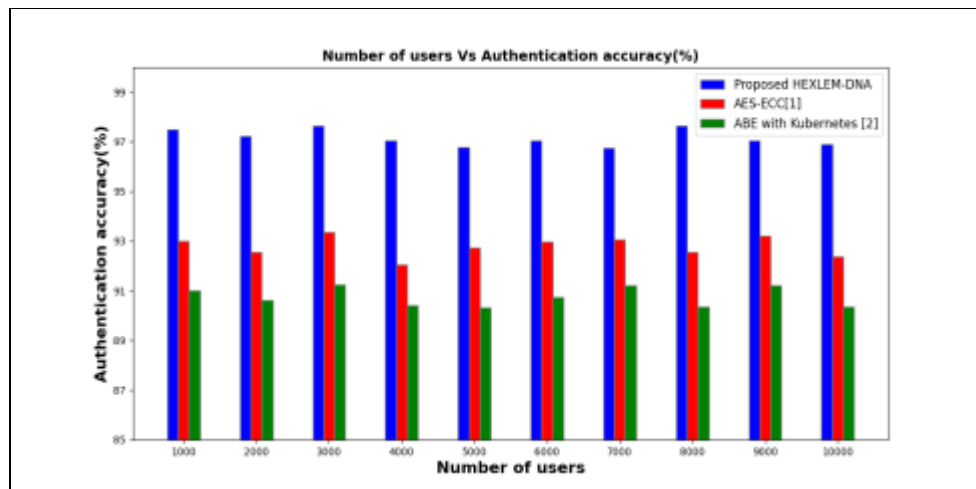


Figure 8 graphical analysis of authentication accuracy

Figure 13 present a graphical representation of authentication accuracy performance across three different methods. In Figure 13, the x-axis denotes the number of users ranging from 1,000 to 10,000, while the y-axis displays the corresponding accuracy values. The results clearly indicate that the HEXLEM-DNA technique consistently outperforms the existing approaches in terms of accuracy. During the initial test involving 1000 users, the proposed HEXLEM-DNA technique achieved an accuracy of 97.5%. The accuracy levels of other methods were recorded as 93% using [1], 91% using [2]. Similar various observed results were observed for different user counts. The observed results of HEXLEM-DNA indicate that accuracy is improved by 5% and 7% compared to methods [1] and [2], respectively. Similarly, various counts of input such as 2000, 3000 ...10000 are considered in experimentation to compute the authentication accuracy. For each method, ten various results were obtained. After collecting these results, the performance of the proposed HEXLEM-DNA technique is compared to that of existing methods. The overall comparison is conducted using the ratio of the absolute difference between the results of the proposed and existing methods to the results of the existing methods. Finally, the average of these ten comparison ratios was calculated. The analysis indicates that the HEXLEM-DNA technique increases the authentication accuracy by 5% and 7% compared to the existing techniques. This enhanced performance is achieved by the integration of extreme learning machine to effectively distinguish between authorized users and unauthorized during cloud data transmission. The Forbes index results helps to performs classification of in the authorized users and unauthorized user at output layer. This process notably increases the accuracy of authenticating the user in the cloud container environment.

Impact of processing time

Processing time is defined as an amount of time considered by the algorithm to perform secure data communication which includes key generation, encryption, authentication and decryption. The formula for calculating the time is measured as follows,

$$PT = \sum_{i=1}^n D_i * \text{Time}(KG + EN + A + DN) \quad (17)$$

Where, PT indicates processing time, D_i denotes a number of data, Time denotes a time for key generation 'KG', encryption 'EN', authentication 'A', Decryption 'DN'. Processing time is measured in terms of milliseconds (ms).

Table 5 comparison of processing time

Number of data instances	processing time (ms)		
	Proposed HEXLEM-DNA	AES-ECC [1]	ABE with Kubernetes [2]
10000	56	63	72.5
20000	60.3	66.8	76.2
30000	63.8	70.5	80.6
40000	67.8	73.6	84.4
50000	71.2	80.2	86.3
60000	74.6	83.6	90.2
70000	78.3	86.4	92.7
80000	82.2	90.2	94.8
90000	86.6	92.4	97.3

100000	90.2	95.8	101.5
--------	------	------	-------

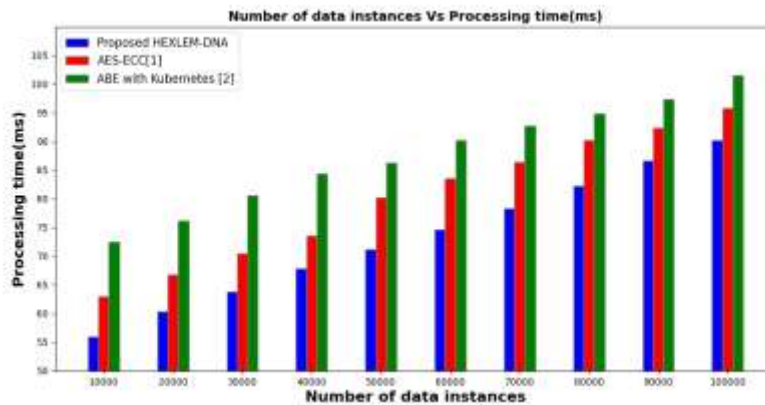


Figure 14 graphical analysis of processing time

Figure 14 illustrates the graphical comparison of processing time across three different techniques namely HEXLEM-DNA and existing AES-ECC [1], ABE with Kubernetes [2]. In the figure, the x-axis represents the number of data instances, while the y-axis indicates the corresponding processing time. As shown in figure 14, an increase in the volume of data leads to a proportional increase in processing time. For assessment, experiments were conducted using 10,000 data instances. The proposed HEXLEM-DNA required a processing time of 56 ms, whereas [1] required 63 ms, [2] required 72.5ms respectively. The observed results of HEXLEM-DNA indicate that processing time is minimized by 11% and 23% compared to methods [1] and [2], respectively. Similarly, multiple results were observed for each method with varying numbers of input data instances. A total of ten results were conducted for each method, and the results were compared. The comparison reveals that the average processing time of the proposed HEXLEM-DNA technique is noticeably reduced by 9% and 17% compared to conventional cryptography methods [1] [2]. This is because of proposed DNA cryptographic model effectively performs the encoding and decoding with symmetric key, thereby notable reduction in the overall processing time. Moreover, the extreme learning machine in HEXLEM-DNA technique accurately identifies the authorized and unauthorized users with minimal time consumption.

CONCLUSION

Data security and privacy are major concerns for users in cloud container to protect the data confidentiality and integrity. In this paper, a robust HEXLEM-DNA technique tailored for secure data storage and access control. The system ensures data security by allowing the data owner to encrypt the data using DNA encryption before uploading it to the cloud container. The cloud container efficiently generates symmetric cipher session key using Blum–Micali algorithm to preserve the shared data and prevent unauthorized access by malicious users during the sharing process. Furthermore, the integration of an extreme learning machine enhances the system's ability to distinguish between authorized and unauthorized users, thereby enabling effective access control. A series of comprehensive experiments were conducted with datasets of varying sizes, and the performance of the proposed model was compared against two conventional cryptography methods. The results

demonstrate that the HEXLEM-DNA technique achieves superior performance in terms of confidentiality, integrity, authentication accuracy and reduction in processing time, outperforming the existing cryptography approaches.

REFERENCES

1. Mohammad A. Altahat, Tariq Daradkeh, Anjali Agarwal, "Optimized Encryption-Integrated Strategy for Containers Scheduling and Secure Migration in Multi-Cloud Data Centers", *IEEE Access*, Volume 12, 2024, Pages 51330 – 51345. DOI: 10.1109/ACCESS.2024.3386169
2. Mauro Femminella, Martina Palmucci, Gianluca Reali, Mattia Rengo, "Attribute-Based Management of Secure Kubernetes Cloud Bursting", *IEEE Open Journal of the Communications Society*, Volume 5, 2024, Pages 1276 – 1298. DOI: 10.1109/OJCOMS.2024.3367461
3. Yuanbo Li, Hongchao Hu, Wenyan Liu and Xiaohan Yang, "An Optimal Active Defensive Security Framework for the Container-Based Cloud with Deep Reinforcement Learning", *Electronics*, Volume 12, Issue 7, 2023, Pages 1-18. <https://doi.org/10.3390/electronics12071598>
4. Ke Xiong¹, Zhonghao Wu^{1,2}, Xuzhong Jia, "DeepContainer: A Deep Learning-based Framework for Real-time Anomaly Detection in Cloud-Native Container Environments", *Journal of Advanced Computing Systems*, Volume 5, Issue 1, Pages 1-17. <https://doi.org/10.69987/JACS.2025.50101>
5. Sehar Zehra, Hassan Jamil Syed, Fahad Samad, Ummay Faseeha, Hamza Ahmed, Muhammad Khurram Khan, "Securing the Shared Kernel: Exploring Kernel Isolation and Emerging Challenges in Modern Cloud Computing", *IEEE Access*, Volume 12, 2024, Pages 179281 – 179317. DOI: 10.1109/ACCESS.2024.3507215
6. Bom Kim and Seungsoo Lee, "KubeAegis: A Unified Security Policy Management Framework for Containerized Environments", *IEEE Access*, Volume 12, 2024, Pages 160636 – 160652. DOI: 10.1109/ACCESS.2024.3487990
7. Seungsoo Lee, Jaehyun Nam, "Kunerva: Automated Network Policy Discovery Framework for Containers", *IEEE Access*, Volume 11, 2023, Pages 95616 – 95631. DOI: 10.1109/ACCESS.2023.3310281
8. Hang Huang, Honglei Wang, Jia Rao, Song Wu, Hao Fan, Chen Yu, "vKernel: Enhancing Container Isolation via Private Code and Data", *IEEE Transactions on Computers*, Volume 73, Issue 7, 2024, Pages 1711 – 1723. DOI: 10.1109/TC.2024.3383988
9. Yurii Tulashvili, Viktor Kosheliuk, "Orchestrating honeypot deployment in lightweight container platforms to improve security", *International Science Journal of Engineering & Agriculture*, Volume 4, Issue 1, 2025, Pages 1-13. <https://doi.org/10.46299/j.isjea.20250401.01>
10. Hao Liang, Gongshan He, Jinhu Jiang, "MSpace: Multi security namespace management for containers", *Journal of Engineering Research*, Elsevier, 2025, Pages 1-7. <https://doi.org/10.1016/j.jer.2025.04.007>
11. Yufei Hana, Chao Li, Jianbiao Zhang, Yifan Wang, Lehao Yu, Yihao Cao, Hong Shen, Weixing Hou, Hailin Luo, "DMSCTS: Dynamic measurement scheme for the containers-hybrid-deployment based on trusted subsystem", *Computers & Security*, Elsevier, Volume 148, 2025, Pages 1-27. <https://doi.org/10.1016/j.cose.2024.104158>
12. Kyungwoon Lee, Jeongsu Kim, Ik-Hyeon Kwon, Hyunchan Park and Cheol-Ho Hong,

- "Impact of Secure Container Runtimes on File I/O Performance in Edge Computing", Applied Sciences, Volume 13, Issue 24, 2023, Pages 1-16.
<https://doi.org/10.3390/app132413329>
13. Sarp Koksall, Ferhat Ozgur Catak, Yaser Dalveren, "Flexible and Lightweight Mitigation Framework for Distributed Denial-of-Service Attacks in Container-Based Edge Networks Using Kubernetes", IEEE Access, Volume 12, 2024, Pages 172980 – 172991. DOI: 10.1109/ACCESS.2024.3501192
 14. Youngsoo Kim, Cheolhee Park, Dowon Hong, "A Robust Framework for Comprehensive Container Image Vulnerability Assessment", IEEE Access, Volume 13, 2025, Pages 35837 – 35847. DOI: 10.1109/ACCESS.2025.3545124
 15. Mingxue Ouyang, Jianqing Xi, Weihua Bai, Keqin Li, "Band-Area Resource Management Platform and Accelerated Particle Swarm Optimization Algorithm for Container Deployment in Internet-of-Things Cloud", IEEE Access, Volume 10, 2022, Pages 86844 – 86863. DOI: 10.1109/ACCESS.2022.3198971
 16. Hui Zhu, Christian Gehrmann & Paula Roth, "Access Security Policy Generation for Containers as a Cloud Service", SN Computer Science, Springer, Volume 4, 2023, Pages 1-15. <https://doi.org/10.1007/s42979-023-02186-1>
 17. [Santosh Ugale and Amol Potgantwar, "Container Security in Cloud Environments: A Comprehensive Analysis and Future Directions for DevSecOps", Engineering Proceedings, Volume 59, Issue 1, 2023, Pages 1-11.
<https://doi.org/10.3390/engproc2023059057>
 18. Soyeon Park and Hyokyung Bahn, "Performance Analysis of Container Effect in Deep Learning Workloads and Implications", Applied Sciences, Volume 13, Issue 21, 2023, Pages 1-15. <https://doi.org/10.3390/app132111654>
 19. Hugo Kermabon-Bobinnec, Sima Bagheri, Mahmood GholipourChoubbeh, Suryadipta Majumdar, Yosr Jarraya, Lingyu Wang and Makan Pourzandi, "PerfSPEC: Performance Profiling-based Proactive Security Policy Enforcement for Containers", IEEE Transactions on Dependable and Secure Computing, Volume 22, 2025, Pages 919-938. DOI 10.1109/TDSC.2024.3420712
 20. Bom Kim, Jinwoo Kim, Seungsoo Lee, "Exploring Security Enhancements in Kubernetes CNI: A Deep Dive Into Network Policies", IEEE Access, Volume 13, 2025, Pages 35322 – 35338. DOI: 10.1109/ACCESS.2025.3543841
 21. Bipin Gajbhiye, Om Goel, Pandi Kirupa Gopalakrishna Pandian, "Managing Vulnerabilities in Containerized and Kubernetes Environments", Journal of Quantum Science and Technology, Volume 1, Issue 2, 2024, Pages 59-71.
<http://dx.doi.org/10.2139/ssrn.4982847>
 22. Faya Safa, Raddad Al King, "Data Security in Cloud Computing", International Journal of Wireless and Ad Hoc Communication (IJWAC), Volume 07, Issue 01, 2023, Pages 50-61, <https://doi.org/10.54216/IJWAC.070105>
 23. Bruno Nascimento, Rui Santos, João Henriques, Marco V. Bernardo and Filipe Caldeir, "Availability, Scalability, and Security in the Migration from Container-Based to Cloud-Native Applications", Computers, Volume 13, Issue 8, 2024, Pages 1-13.
<https://doi.org/10.3390/computers13080192>
 24. Juan Martinez Delbugio, Vijay K. Madiseti, "Enhanced Memory-Safe Linux Security Modules (eLSMs) for Improving Security of Docker Containers for Data Centers", Journal of Software Engineering and Applications, Volume 17, Issue 5, 2024, Pages 259-269. doi: 10.4236/jsea.2024.175015.

25. Kennedy Chinedu Okafor, Wisdom Onyema Okafor, Omowunmi Mary Longe, Ikechukwu Ignatius Ayogu, Kelvin Anoh and Bamidele Adebisi, "Scalable Container-Based Time Synchronization for Smart Grid Data Center Networks", *Technologies*, Volume 13, Issue 3, 2025, <https://doi.org/10.3390/technologies13030105>
26. Tianxiang Yu, Yang Xin and Chunyong Zhang, "HoneyFactory: Container-Based Comprehensive Cyber Deception Honeynet Architecture", *Electronics*, Volume 13, Issue 2, 2024, Pages 1-27. <https://doi.org/10.3390/electronics13020361>
27. Jueun Jeon, Sihyun Park, Byeonghui Jeong, Young-Sik Jeong, "Efficient Container Scheduling With Hybrid Deep Learning Model for Improved Service Reliability in Cloud Computing", *IEEE Access*, Volume 12, 2024, Pages 65166 – 65177. DOI: 10.1109/ACCESS.2024.3396652
28. Toshihiro Uchibayashi, Bernady Apduhan, Takuo Suganuma and Masahiro Hiji, "Experiments and Evaluation of a Container Migration Data-Auditing System on Edge Computing Environment", *Computers*, Volume 12, Issue 2, 2023, Pages 1-18. <https://doi.org/10.3390/computers12020027>
29. Anish Saini, Athanasios Tsokanos and Raimund Kirner, "QuantumGS-Box—A Key-Dependent GA and QRNG-Based S-Box for High-Speed Cloud-Based Storage Encryption", *Sci*, Volume 6, Issue 4, 2024, Pages 1- ; <https://doi.org/10.3390/sci6040086>
30. Amir Javadpour, Forough Ja'fari, Tarik Taleb, Chafika Benzaïd, Yang Bin, Yue Zhao, "Encryption as a Service (EaaS): Introducing the Full-Cloud-Fog Architecture for Enhanced Performance and Security", *IEEE Internet of Things Journal*, Volume 11, Issue 24, 2024, Pages 39744 – 39766. DOI: 10.1109/JIOT.2024.3450192