future internet

MDPI

Article

Enhancing Design and Authentication Performance Model: A Multilevel Secure Database Management System

Hemin Sardar Abdulla and Aso M. Aladdin *

Department of Computer Science, College of Science, Charmo University, Sulaymaniyah 46023, Iraq; hemin.abdulla@chu.edu.iq
* Correspondence: aso.aladdin@chu.edu.iq

Abstract: Multilevel security (MLS) is particularly intended to secure information against unauthorized access. An MLS security DBMS allows users with different security levels to access and share a database. For this purpose, the study creates a model that includes a restricted access authentication prototype with multilevel security in a database management system (MLS/DBMS). Accordingly, the model has been designed to emphasize the highest level of authorized security. The system ensures that users can only access information that they are permitted to view, fully adhering to the newly established MLS framework. In addition, the model also integrates cryptographic algorithms, such as RSA and AES, to enhance its functionality and demonstrate the scalability and security of the model. These criteria are defined based on the perspective of the database provided to users, determined by their respective authorization levels. An informal security framework for a multilevel secure DBMS is defined. It includes a classification strategy and explains the implementation of operations like insertion and deletion, addressing the complexity of models with novel methods. The metric evaluation of this model assesses the performance of the authentication process and how operations are implemented across three authentication group types. It also calculates the key generation time and encryption types in cryptographic algorithms. The results confirm that the RSA model requires less time for evaluation while maintaining multilevel security. Furthermore, the type 2 authentication group is more complex and requires more memory and time for generation. Based on the classification, the results highlight notable differences, which designers should consider when selecting authentication methods. Lastly, the study presents various conclusions, explores possible future directions, and discusses its limitations.

Keywords: multilevel security; information system; database management system; database security; access control; system implementation

Academic Editor: Steven Furnell

Received: 13 December 2024
Revised: 26 January 2025
Accepted: 5 February 2025
Published: 8 February 2025

Citation: Abdulla, H.S.; Aladdin, A.M. Enhancing Design and Authentication Performance Model: A Multilevel Secure Database Management System. *Future Internet* **2025**, *17*, 74. https://doi.org/10.3390/fi17020074

1. Introduction

The rise of smart systems brings ethical concerns as cyber–physical infrastructure becomes vulnerable to failures. These hazards include privacy breaches, system disruptions, and significant economic and human losses [1–3]. Without strong computer and network security, technology endangers personal privacy and increases the likelihood of severe social and financial consequences [4]. Numerous global systems and databases face constant attacks, particularly targeting privacy and authentication mechanisms [5]. Thus, several models highlight the benefits of multilevel security achieved through a three-layer system, using three security evaluations to enhance protection and ensure thorough coverage. Various algorithms and models in cybersecurity, such as Rivest–Shamir–Adleman

(RSA) [6], Elliptic Curve Cryptography (ECC) [7], and Advanced Encryption Standard (AES) [8], are designed to detect security threats, anomalies, and fraud by effectively correlating information. These can be integrated with multilevel security to develop optimal authentication and standard models.

To address the question of what is most important when selecting multilevel security for authentication and data communication, as proposed by [9], several factors are considered and addressed when selecting and generating these types of models. The term "multilevel security" is used to describe the requirement of defining multiple categories or levels of data. However, the significance of a semantic for multilevel models cannot be overstated, as it has the potential to decrease the number of tuples inside relations and minimize the existence of numerous versions of reality across different security levels [9]. The primary motivation of this study is to develop a newly generated multilevel secure model to assist developers in creating systems for big data. This model aims to provide the most effective authentication methods, apply robust user restrictions, and safeguard systems from attacks and vulnerabilities across various domains, such as economics, politics, healthcare, and others.

The essential requirement for ensuring the security of a system is the existence of a clear set of guidelines that define which subjects are authorized to access exact objects [10]. "Access" is a fundamental notion, signifying the transmission of information either from a subject to an object or from an object to a subject. For instance, when a user (acting as a subject) updates the dataset (functioning as an object), information is transferred from the subject to the object. When a user retrieves a record from a dataset, information is transferred from the object to the subject, as highlighted in the following key points:

- Multilevel security facilitates the categorization of both data and users according to a hierarchical system of security levels provided.
- Database management systems are designed to facilitate the storage, organization, and modification of data in a manner that allows for efficient access and management.
- A multilevel secure approach ensures that a DBMS implements a system allowing access to the database only for authorized users.
- The information is divided into classes and users are able to be given permission to view particular types of data. Although multiple types or levels of data are established, such a requirement has been referred to using the term multilevel security.
- The approach is to reduce time consumption during the authentication process while maintaining security mechanisms.
- Multilevel points can be integrated with cybersecurity algorithms to enhance system performance during the process.

The strategy for designing a new Multilevel Security/Database Management System (MLS/DBMS) structure includes detailed approaches for distributing data models. This approach is introduced to implement multilevel security and support the development of a DBMS prototype. This model emphasizes designing a new strategy for the MLS/DBMS structure, detailing the approaches for distributing data models to implement multilevel security and achieve optimal authentication security. It also involves developing a DBMS prototype using the proposed model. Additionally, the model's performance is evaluated by comparing its effectiveness when integrated with cybersecurity algorithms during encryption and authentication processes. The main contributions of this study are discussed according to the following key points:

- The primary objective of implementing a new multilevel security model in a database system is to efficiently prevent unauthorized data access during the mechanism of the authentication process.

- This is achieved by restricting users from accessing data outside their authorized access privileges.
- Various types of multilevel security database management systems have been selected to improve the implementation of multilevel security in relational databases.
- This study involves testing applications using several example prototypes of authentication to evaluate and select the best level of security for use in database or cloud systems.
- To evaluate the best level of security during process mechanisms, measure the elapsed time during processing.
- Two strategies, "No Read-Up" and "No Write-Down", are implemented with the INSERT and DELETE operations to enhance the MLS/DBMS model.
- The model is integrated with RAS and EAS to achieve optimal performance for each prototype and assist developers in applying restrictions during implementation.
- The key generation, encryption, and decryption elapsed times are compared, along with the memory usage during the process.

The paper is organized as follows: Section 2 discusses multilevel security, categorizing data and users based on security levels. Section 3 provides an overview of database management systems (DBMS), focusing on efficient data storage, organization, and access, while ensuring only authorized users can access the database. Section 4 details the design strategy for the MLS/DBMS structure, including approaches for distributing data models and developing the DBMS prototype. Section 5 concludes the paper.

2. Background Review

Multilevel security employs a strategy that classifies data and users using a combination of hierarchical security levels and non-hierarchical security categories [11]. According to [12], a multilevel security policy has two main objectives. The controls must first prevent unauthorized users from accessing data classified above their acceptable level. Secondly, they must restrict entities from declassifying data.

In a multilevel secure system, when a subject attempts to access an object, the system must decide whether to grant access. For example, Budati et al. proposed a multilevel privacy protection approach to mitigate data compromise attacks on private cloud systems [13]. Additionally, other studies have focused on developing and deploying a functional prototype of a multilevel secure database system [14]. Lin focused on designing a multilevel blockchain architecture to uniformly distribute operations [15]. Jiang et al. also proposed an algorithm for image encryption using a multilevel permutation technique [16]. Some models are designed to integrate with distributed systems, providing a set of security patterns for large, complex systems [17]. The intelligent system's multilevel security policy model has been examined as a guide for its main application, which is building dependable distributed systems [18]. In another study, the use of a tool designed for users improved the instructional efficacy of teachers in presenting material [19]. In publication [20], the authors present a framework that addresses the security concerns of private cloud systems.

The implementation of the multilevel secure database prototype was carried out using the Microsoft SQL code program. However, performance evaluation experiments were conducted utilizing the implemented prototype [21]. Additionally, the investigations examined various aspects of multilevel security in distributed database management systems. These aspects include system design, security policy, and data/metadata distribution challenges in the context of the MLS.

Some inquiries have explored the architectural considerations and principals involved in developing a secure database system [22,23]. The current robust and secure Medium Access Control (MAC) standards have provided effective and anticipated defense against

channel attackers [24]. In artificial intelligence, the homomorphic encryption-based deep learning secure searchable blockchain system functions as a distributed database, allowing users to securely access data through searches [25]. Several strategically positioned smart contacts, including multiple access control intelligent contracts, have been used as part of the trusted and secure system that guarantees the implementation of these functionalities [26]. As highlighted in the previous section, some models in healthcare use techniques to preserve healthcare information securely. These processes ensure compliance with the security and privacy requirements of the Health Insurance Portability and Accountability Act (HIPAA) [27]. Most studies have focused on performance and security mechanisms, such as scalability and availability. Building on this prior work, our research integrates and justifies the development of an innovative model by incorporating cryptographic security algorithms.

Additionally, another approach has enabled fine-tuned access control across various partnership levels, creating a trusted platform for data sharing. It also designs an automated system to determine these partnership levels [28]. In study [29], the authors explore enhancing multilevel security using latent semantic indexing. This approach simplifies the management of multilevel security-based operating systems while enabling the effective utilization of extensive security measures. Wang et al. propose a fast and efficient method that requires minimal computation while effectively suppressing noise and improving accuracy. The study integrates network detection with information exchange to enhance feature extraction for detecting lens defects. It employs RSA and AES cryptography combined with multivariance techniques to strengthen authentication [30]. Various research efforts, including theses, focus on developing and optimizing systems such as blockchain, network sensor data, or healthcare data. These studies place different emphases on performance, scalability, security, and availability [31,32]. In addition to the discussion in the review, Table 1 summarizes the key steps of the approach, highlighting various aspects of the background relevant to this purpose. Each step will be described in detail, including the specific techniques or tools and the rationale behind their selection. This aims to ensure replicability and facilitate understanding of the methodology for researchers familiar with their field's conceptualization.

Table 1. Detailed background review of multilevel security in distributed systems.

Years	Ref.	Purpose	Drawbacks	Description
2024	[33]	Secure mechanism for generating password hash values	Difficult to assess validity and effectiveness.	The generated passwords are hashed using various cryptographic methods to evaluate critical security properties, such as the time needed to crack them.
2023	[13]	A secure multilevel privacy-protection scheme based on GAN	Potentially limiting the applicability of the described security patterns to contemporary complex systems.	AES is used to encrypt data transformation matrices, which are then encrypted. Level of protection for the data-transformation matrices makes the suggested method resistant to insider data-leakage violence.

Table 1. *Cont.*

Years	Ref.	Purpose	Drawbacks	Description
2023	[15]	A multilevel blockchain framework to secure information security on the internet of vehicles	Lack of practical effectiveness, scalability, and efficiency in real-world scenarios.	The non-repudiation of the exchange of data and the integrity of the transaction data are guaranteed by a new transaction block and the ECDSA digital signature. Designed with efficiency in mind, the multilevel blockchain architecture allocates tasks between intra- and inter-cluster private blockchains.
2023	[16]	A new image encryption algorithm based on the LCCM and multilevel manipulation	Difficult to objectively evaluate their effectiveness and improvement over other approaches.	By combining the Logistic map with the Chebyshev map, a novel one-dimensional Logistic–Chebyshev chaotic map (LCCM) is introduced in order to improve the inconsistency of the Chebyshev chaotic sequences.
2022	[24]	Implemented with distributed WIHFM and ODAC virtual link management features	It is vulnerable to uncertain honeypot response times due to continuously changing network dynamics.	As part of the recently created model, the IHF characteristics were recognized as innovative distributed solutions that were included in every sensor node of the WSN.
2022	[25]	Implemented a novel extended approach of homomorphic encryption in a digital healthcare system leveraging blockchain technology	Deep learning techniques for attack prediction and monitoring, without exploring the other advanced methods.	Development of a distinctive deep learning-based secure searchable blockchain as a distributed database utilizing homomorphic encryption to provide secure data access through search.
2022	[28]	Secure data sharing scheme (SDSM) for IoT-based supply chains	Not suitable for involving data sharing and collaboration across multiple supply chains.	This method facilitates the implementation of precise access control across several levels of partnerships, enabling the identification of partnerships.
2019	[34]	Encrypted password security via extended ADFGVX	It may still be vulnerable to advanced cryptographic attacks.	Proposed encryption accommodates common password characters with a random key.
2016	[17]	Integrated set of security patterns was created based on the Reference Monitor abstraction	The high cost and time required for the rigorous development and evaluation of secure systems.	TCSEC and TNI provide security frameworks for complex, distributed systems.
2014	[19]	MLSvisual, a tool that helps students learn the multilevel (Bell–LaPadula) access control model	The impact on students' understanding is influenced by the time spent using MLSvisual.	MLSvisual enhances instruction and self-study of MLS access control systems, allowing instructors to reference policies and clarify topics during class for better understanding.

Table 1. *Cont.*

Years	Ref.	Purpose	Drawbacks	Description
2014	[20]	Prevent data leaks, especially caused by mis operation and malicious inside users	The BLP-based model may not be fully representative of real-world usage, where more complex attack vectors could exist.	Access to data is managed by the security levels of subjects and objects; the present security levels of subjects may be altered by the study of sensitive data.
2012	[21]	Combination of the Multilevel Relational (MLR) model and an encryption system	Suffers from performance issues during data updates due to the need for encryption and decryption operations.	The encryption algorithm encrypts each data component in the tuple with a unique field-key based on its security classification, and decrypts each field-key separately.
2003	[14]	Belief-Consistent Multilevel Secure Relational Data Model (BCMLS) as a basis for prototype	Complex and requires substantial effort on both the server and client sides.	The prototype, developed on the software database server via a PHP web client, facilitates the input, deletion, and updating of multilevel data.

3. Methodology

Based on previous works, a multilevel secure system is essential to safeguard sensitive data. Our research addresses key subjects, particularly identifying optimal groups for an authentication system through a separate framework. This approach aligns with the deliberations in the first section. We propose a technique for developing a model that simultaneously facilitates and classifies authentication procedures. This model is designed and implemented using the new system framework, as outlined in a separate section. In addition, another section focuses on identifying cryptographic algorithms to select trusted groups based on four criteria: the time required for key generation, encryption, and decryption; the memory requirements; and integration with the MLS/DBMS model. These aspects are discussed in detail to ensure robust and efficient performance.

3.1. Trusted Framework System

It is crucial to develop new methods for controlling the framework system during the setup of new online or cloud systems. Although many existing systems provide some degree of protection, they all possess inherent weaknesses. It is essential to develop a stronger method that ensures multilevel security to protect data and resources. This is commonly used in the military, where information is categorized as unclassified (U), confidential (C), secret (S), top secret (TS), or higher. This concept can be applied across various domains, where information is systematically categorized, and users are granted access to specific types of data. Multilevel security refers to the need to define multiple categories or levels of data. In addition, the allowed data flows in an MLS system illustrate all acceptable interactions between a subject operating at the "Secret" level and various objects with different security levels, as clearly depicted in Figure 1 with assistance from [35]. Furthermore, according to the Bell–LaPadula model [36], which underpins the generation of the MLS/DBMS model, two key properties are enforced. These strategies are discussed in detail below:

- **No read-up**: A fundamental principle in information security, known as the simple security property, states that a subject is only allowed to access or read an object with an equivalent or lower security level.

- **No write-down**: A subject with a specific security level can write to an object with a security level that is equal to or higher than its own. This principle is commonly referred to as the * property, pronounced as the "star property".

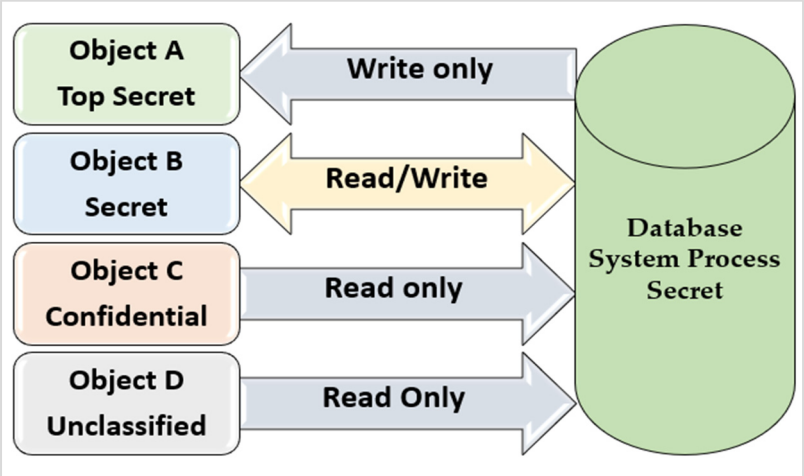

Figure 1. Permissible data flows enabled by the MLS framework.

We focused on these two strategies as novel guidelines for determining authentication classifications. When applied precisely and strictly, they provide multiple levels of security. The strategies implemented, extensively researched and developed, are based on the reference monitor concept as described in [37]. This approach was chosen for its well-established foundation in research and development. It serves as a critical component of operating systems and computer hardware, overseeing and managing access authorizations between subjects and objects. Consequently, the reference monitor in MLS/DBMS enables access to a designated file, known as the operating system security database, which records each subject's security clearances and each object's classification levels, as shown in Figure 2. As a result, our model employs this regulation, which is based on the security parameters of both the subject and the object involved.

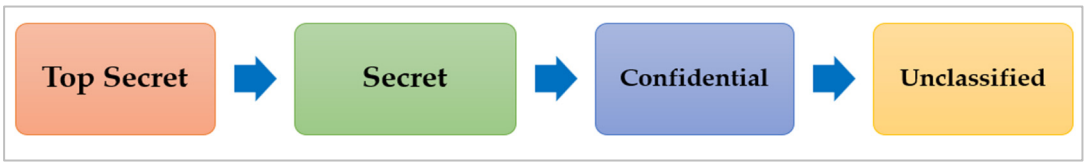

Figure 2. Initial establishment of multiple clearance levels.

3.2. System Design Methodology

The goal of this method is to design and implement the best technique for solving authentication access challenges in big data. The approach focuses on increasing user access to data while balancing cost efficiency and utilizing query languages, all while enhancing data security. It ensures higher data independence from application programs, maintains integrity during authentication, and guides developers in restricting unauthorized access processes.

A main aspect of the study is the exploration of character variations, analyzing the use of uppercase, lowercase, and special symbols to strengthen password security. This investigation highlights how these variations enhance password complexity, improve resistance to unauthorized access efforts, and boost overall security performance and scalability.

3.2.1. Mandatory Access Control

In a relational DBMS, unauthorized access to the database represents a failure to maintain confidentiality. Similarly, unauthorized modifications to the data indicate a breach of data integrity. Furthermore, a lack of access to database system services reflects a failure to ensure availability. The designed aspects of MLS/DBMS are explored further in the following subsection.

In the MLS/DBMS model, discretionary access control is combined with MAC to classify users and data into specific security levels, as defined by the Bell–LaPadula model. This popular framework for MLS assigns every subject and object a security classification, such as Top Secret (TS), Secret (S), Confidential (C), or Unclassified (U), following the hierarchy $TS \geq S \geq C \geq U$. Two key restrictions are imposed on data access to ensure security. Addressing these restrictions, the model identifies and classifies each access authentication within a relational DBMS during access control.

As designed in this novel model, Figure 3 illustrates the allowable data transfers within the MLS framework. It provides a detailed analysis of permitted data interactions between a subject operating at the (S) level and various objects with different security levels. These interactions as previously discussed are governed by the Bell–LaPadula model's two key strategies: "no read-up" and "no write-down".

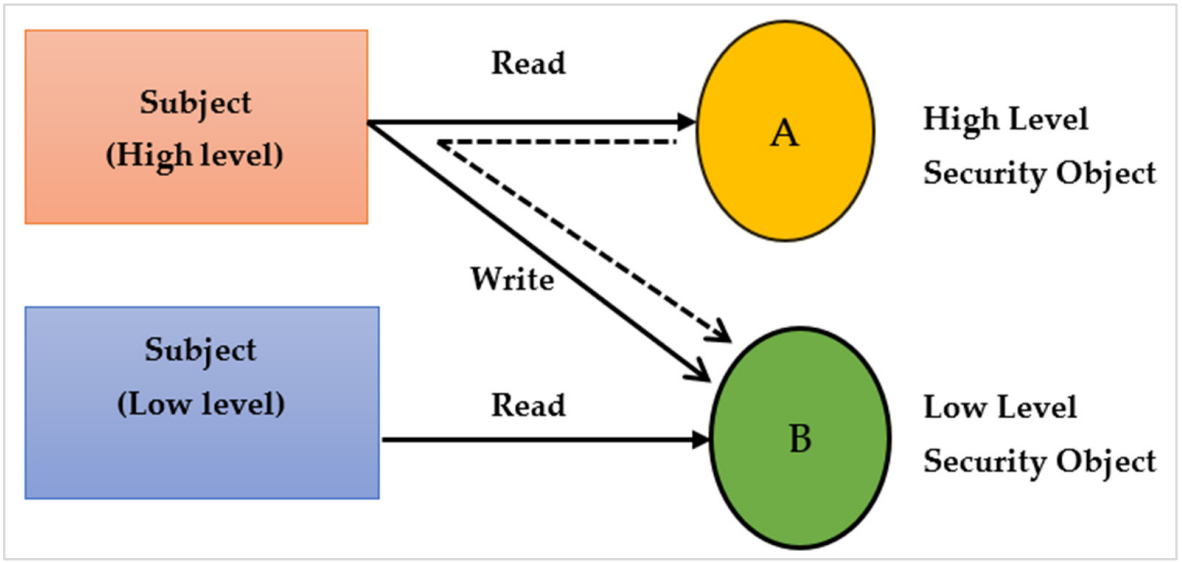

Figure 3. The MLS data model for the illegal information exchange.

3.2.2. Multilevel Relational Model

In multilevel relational databases, attribute values and tuples are generally treated as data objects with assigned security classifications. In the MLS/DBMS model, each attribute in the schema must be linked to a classification attribute (K), and every value within a tuple must be associated with its corresponding security classification. Additionally, certain models include a tuple classification attribute (TK) to assign an overall security level to each tuple.

Therefore, a multilevel relational system is represented as $R(A_1, K_1, A_2, K_2, \ldots, A_n, K_n, TK)$, where each A_i is a data attribute, each K_i categorizes A_i , and TK classifies the tuple. Relation instances consist of distinct tuples in the form $(a_1, k_1, a_2, k_2, \ldots, a_n, k_n, tk)$. The TK attribute indicates the highest classification within a tuple, while each K_i ensures the security classification of its respective attribute.

The primary key in an MLS relationship is derived from attributes that form the primary key in single-level relations. Depending on their authorization levels, users

(subjects) view different data in a multilevel relation. With a filtering method, a single tuple can be kept at a higher classification level, with corresponding tuples generated at lower levels when required. This relationship in the MLS/DBMS design assists developers in maintaining consistent categories within relations and selecting the most appropriate authentication restriction type.

The model emphasizes authentication, so the process begins with creating an account, as illustrated in Figure 4. To clarify how the MLS/DBMS model operates, a flowchart can be created. The process starts by prompting the user to confirm their username and password. The flowchart then includes a decision point with the question: "has an account?", which leads to two possible paths:

- No: If the user does not have an account yet, the process continues to the next step.
- Yes: If the user already has an account, the process skips to the step labeled "Login".

After a successful login, the user proceeds to the step labeled "Check the databases for existing accounts based on user accessibility". However, the subsequent step appears redundant, as the system has already determined that the user has not created an account (as indicated in decision step 2, Figure 4). This redundancy may be a leftover from a previous version of the flowchart, potentially representing additional internal verification. The final box, labeled "User Management", indicates that the account was successfully created and that the user now has access to the user management system.

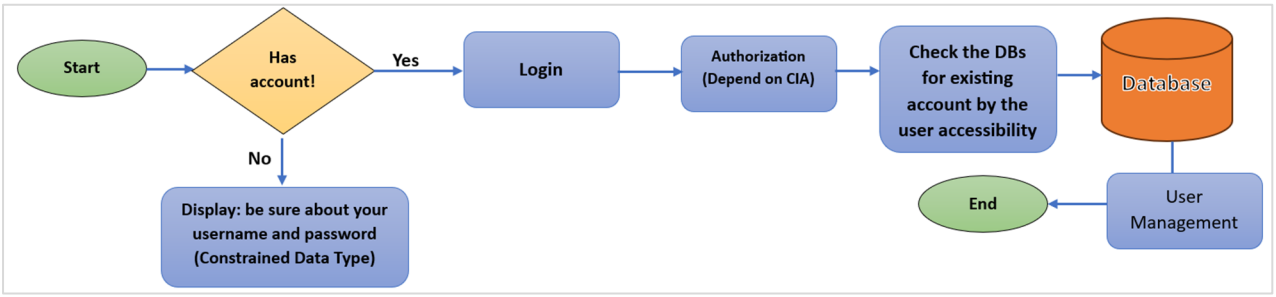

Figure 4. Simplified MLS/DBMS model process for checking an account in the database.

3.3. Multilevel Security Techniques

To propose an MLS integrated with a database system for implementing the projected mechanism efficiently, a strong database system is required. Figure 5 provides a flowchart detailing the proposed system administration framework, which administrators use to manage user accounts for system access. The proposed paradigm includes several data manipulation declarations. The model manages data through INSERT and DELETE operations, allowing for data manipulation based on standard SQL queries. The MLS data model explains how these INSERT and DELETE operations function, as outlined below:

I. INSERT operation: This operation performed by a subject at the class level L follows the general form outlined in Equation (1).

$$\boldsymbol{INSERT\ INTO} R\left[\left(A_{j1}\left[, A_{j2}\right]\ldots\right)\right]\boldsymbol{VALUES}\left(a_{j1}\left[, a_{j2}\right]\ldots\right) \quad (1)$$

According to this equation, R represents the relation's name, and $\left(a_{j1}\left[, a_{j2}\right]\ldots\right)$ are the corresponding attribute names where $\left[\left(A_{j1}\left[, A_{j2}\right]\ldots\right)\right]$ are the relative attribute names and $1 \leq j_1, j_2 \ldots \leq n$. Each INSERT statement is limited to inserting a single tuple into the relation R. The process of inserting a tuple t is carried out through the following steps, ensuring all attributes in the database relation are addressed:

Step 1: Obtain the security level from the subject running the insert operation.

Step 2: When the attribute is included in the attribute list of the insert statement, its value will be obtained from the value list of the insert statement.
Step 3: The security level of all attributes will match that of the subject performing the insert action.
Step 4: The goal is to insert values from a list into relations with a single security level, ensuring the user's defined security level matches the values and aligns with the relations' characteristics.
Step 5: A_i should have the value set to null because it is not an attribute in the INTO clause's list.
Step 6: The subject whose class level is used to perform the insert statement will be set to the tuple-class.

II. **DELETE operation**: This operation, known as DELETE, performed by a topic with class level L, is typically represented in Equation (2).

$$DELETE\ FROM\ R \tag{2}$$

According to our model, R represents the relational name, assuming the relation R includes data attributes $A_1, \ldots, A_n$; P is an expression that may include the delete operation as a predicate according to Figure 5. Only the tuples $t \in r$ with $t[TC] = L$, where L is the classification level of the subject executing the delete statement, are affected.

For tuples $t \in r$ that satisfy the predicate expression P, r is modified through the following steps to address the issue until an optimal solution is achieved:

Step 1: Check the security level of the subject for the deletion operation.
Step 2: Remove all of the tuples that are contained within single-level relations and that are in accordance with the deletion prerequisites, with P being able to delete operations and having a security level that matches that user's level.
Step 3: Deletion will occur for tuples that fulfill the predicate expression.
Step 4: The value of any attribute in a higher-level tuple that depends on an attribute in the deleted tuple will be set to null if found.

The process starts with i to 1 as shown in Figure 5, traversing a user list. Users meeting criterion P are stored in $t(A)$. This denotes a provisional array or list containing these selected users. The loop iterates through users in $t(A)$, evaluating standards and setting Cpk to $L(user)$. Parameter A_i, within range $R[A_i]$, filters entries (A_i) meeting condition P. After processing, i increments, advancing the loop to the next user in $L(user)$.

The process checks if A_i has a defined value ($A_i = a_i$) or is null. If null, the access fails the criteria, requiring adjustments; otherwise, progression continues. After this check, i increments, iterating through accesses. The value $C_i = L(user)$ is assigned for further processing, followed by executing an "INSERT" operation to update the user class.

After the initialization is finalized, the system retrieves entries (A_i) that meet condition P, filtering data to identify accesses matching specific criteria within the user class. After retrieving these entries, the counter i increments, indicating a loop structure where each iteration processes a new entry in $L(user)$. A decision point then checks for additional entries satisfying P. If more entries exist, the loop continues; otherwise, the process deletes (A_i), removing the processed data and concluding the progression.

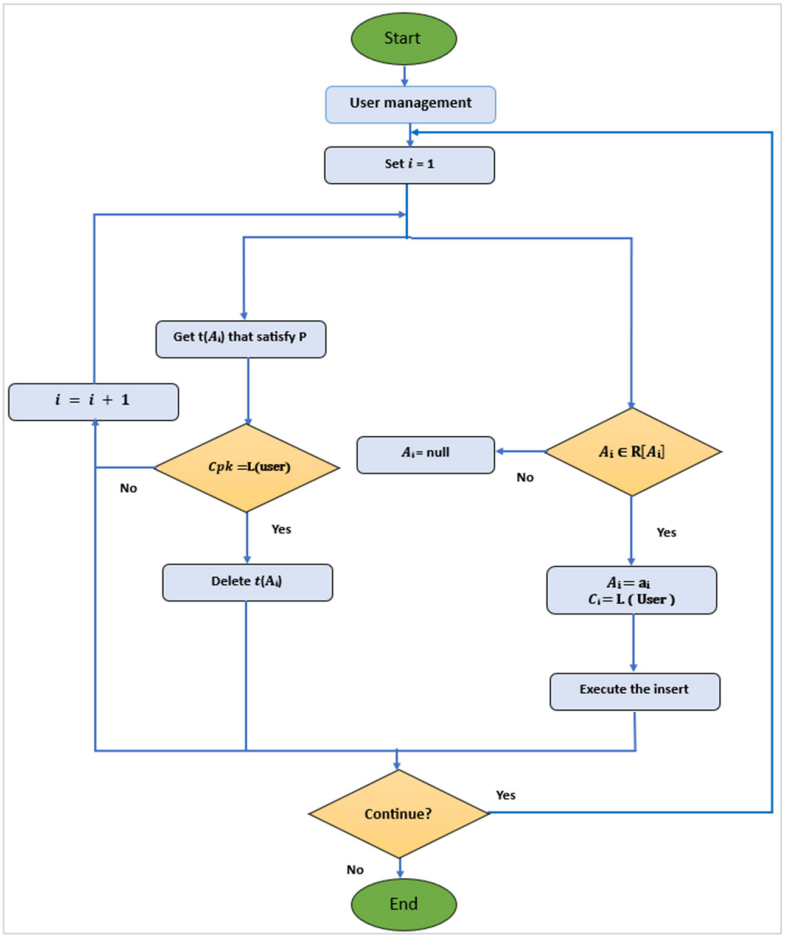

Figure 5. The flowchart provides a basic outline for a user management process.

3.4. Cryptographic Techniques

To evaluate this issue in the model, we focused on two cryptographic algorithms that secure data using mathematical algorithms and protocols. While these models address various aspects, in MLS/DBMS they are specifically used for evaluating access authentication. In authentication, cryptography aids in verifying the identity of users and systems, ensuring that only authorized individuals can access specific resources. Techniques such as password hashing and two-factor authentication are commonly used. The two cryptographic algorithms used in this model are RSA and AES, which are briefly described in the following subsections.

3.4.1. Rivest–Shamir–Adleman Algorithm

The RSA cryptosystem, developed by Ronald Rivest, Adi Shamir, and Leonard Adlemanin [6], is the earliest and most extensively used asymmetric encryption system. Since its inception, it has been employed in various cryptographic applications, including digital signatures, financial services, e-commerce, and email privacy. The algorithm's security relies on the difficulty of factoring large prime numbers.

The RSA process has four basic techniques: key generation, encryption, decryption, and memory usage. The focus is on scalable and secure authentication access. During key generation, we measure the time for the first three techniques in milliseconds (ms) and track memory usage in bytes. This technique is implemented in the MLS/DBMS model for authentication. To compute the modulus n, two large prime numbers, p and q, are chosen. The message to be encrypted or decrypted is denoted as m. The encrypted message ccc

is generated using the public key (n, e), and the received message q is decoded using the private key (n, d) with Equation (3).

$$m = c^d \ mod \ n \tag{3}$$

3.4.2. Advanced Encryption Standard Algorithm

AES is a symmetric encryption algorithm commonly used to secure data, particularly in authentication processes, such as encrypting passwords or email addresses. AES operates on fixed 128-bit blocks and supports key sizes of 128, 192, and 256 bits. It uses multiple stages of substitution, permutation, and mixing to convert plaintext into ciphertext. AES is highly efficient, secure, and used in various database applications like banking, government, and telecommunications. Similar to RSA, this technique is used to address four cryptographic issues, as discussed in the MLS/DBMS model.

Scientifically, AES operates on a block of plaintext P and key K to generate ciphertext C. The encryption process includes several steps (rounds) of substitution, shifting, mixing, and key addition. For a given round r, the transformation can be expressed in Equation (4), where C is the resulting ciphertext after applying the AES operations.

$$C = AES(P, \ K) \tag{4}$$

4. Manual Implementation of the Model

The demonstration of the MLS/DBMS model is generated using various comparative programs, aiming to identify several multilevel relational database security models. However, the evaluation of the cryptographic technique is implemented separately from the generated model to compare its results with those of the hybridized technique. This helps identify the outcomes and provide more specific experimental evaluations, which are detailed in next section. Thus, all of the steps needed to implement this model are identified to provide complete knowledge based on its structure, including method usage, software development, and graphical user interface with console-based implementations.

4.1. Performance Study

All models must evaluate the performance of both software and hardware. This model is designed to analyze the performance evaluation of continuous multilevel database models while maintaining belief consistency. It highlights the impact of varying sizes and structures of relational multilevel databases on the effectiveness of this model.

The hardware used for conducting the performance study includes a central processing unit (CPU) with a 2.1 GHz Core i7 processor, 16 GB of physical random-access memory (RAM), and a 1 TB hard disk drive (HDD) for storage.

Initially, this study was implemented using Microsoft Visual Studio 2008 and Microsoft SQL Server 2008 as software tools. However, in the second repetition, the implementation was redesigned using Microsoft Visual Studio 2022, incorporating cryptographic techniques. The experiments conducted in this study aim to evaluate the effectiveness of hierarchical multilevel database models by analyzing the impact of varying the number of tuples, the number of attributes, and the levels of security. The primary objective is to measure the elapsed time performance and ensure accurate problem detection.

Despite the comparative cryptographic results, mostly for RSA and AES used separately in Python, the 2022 version (Python 3.11.0) supports these algorithms through libraries like PyCryptodome and Cryptography. Both libraries deliver secure and efficient encryption, decryption, and key generation.

4.2. Login Systems: User Authentication Interface with Validation

Cybersecurity Login involves the process by which a user confirms their identity, validates their authenticity, and accesses a system with computers. User login information typically includes a "password" corresponding with a "username", and this login information typically becomes known as a login. The user enters the confidential information and the software package authenticates the user's username and password. The log archive is considered crucial given that it serves as the system's access point.

Ensuring that only authorized users can access a login session is a critical component of any security system. To keep access and authorizations secure, users should log out after each session. Conspicuously, MLS/DBMS systems address this issue by managing read and write operations during the process, classifying all usernames and passwords within the model. After the administrator enters the username and password successfully (see Figure 6), the system grants access to all options and permissions. Once logged in, the main system window appears, screening forms linked to a database for each user authentication level.

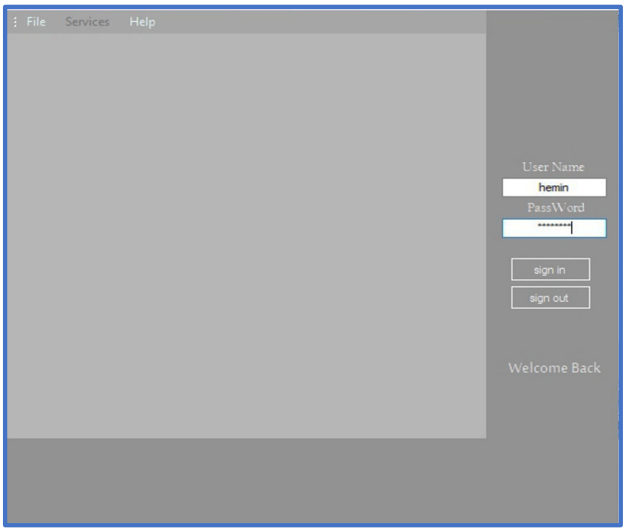

Figure 6. Authentication via username and password, accompanied by an alert message.

4.3. User Interface Design for Secure Information Workflow Management

This study presents a novel policy for implementing an MLS/DBMS system, offering confidential authentication for each user at the highest level when connecting to a remote server. As a result, users must collaborate with others at the same level, based on their operational priorities (INSERT, UPDATE, DELETE, SELECT), to generate a final confirmation that ensures the completion of the distribution process.

Figure 7 illustrates the administrator's ability to add or remove users in the user table, where each user has an ID, Name, Nickname, and Password. Additionally, the model allows for actions such as ADD, PRINT, DELETE, EDIT, MENU, ADMIN, AND STOP USER, along with permissions (U, C, S, T) approved by the administrator. The evaluation of problems can also be addressed using RSA and ASE when confirming the command, depending on the integration of the MLS/DBMS model.

The overwhelming majority of database security research focuses on database administrators (DBAs), which possess the authority to oversee and regulate database transactions. The administrator must be identified to oversee all of the database administrators and regulate what privileges they have. The initial policy establishes an administrator responsible for overseeing and monitoring all databases, their users, and all data transmissions throughout the database server.

All actions demonstrate that the generated model enables adding, modifying, removing, and navigating user records, while also supporting administrative tasks and other operations. The design highlights a focus on user administration and the management of secure or confidential data.

Figure 7. Administrator functionality performance.

4.4. Agent Information Management for Personal Data and License Control

The normal security level ensures each user can access the rights already granted in the system. For example, if a user has permission to add or use other options, they can perform actions like adding, saving, editing, or deleting. All issues are identified in the form shown in Figure 8. To secure the data, the user logically needs an identifier, and this form includes all necessary clarifications. It provides fields for entering general information, such as the name of user, ID number, and location, as well as license details like the license number and expiration date.

Figure 8. The normal security level identifies user access to the system.

Figure 9 illustrates that, following the "no read-up, no write-down" principle, users can access important functionalities only at their designated level. For example, a user at the "unclassified" level can perform all actions within their access rights. The information is managed through a software interface titled "Control Panel Information", designed for administering classified data or records. The interface is separated into fields with specialized functionalities, focusing on data management and classification. It is considered for administrative or security information management, ensuring efficient navigation and functionality.

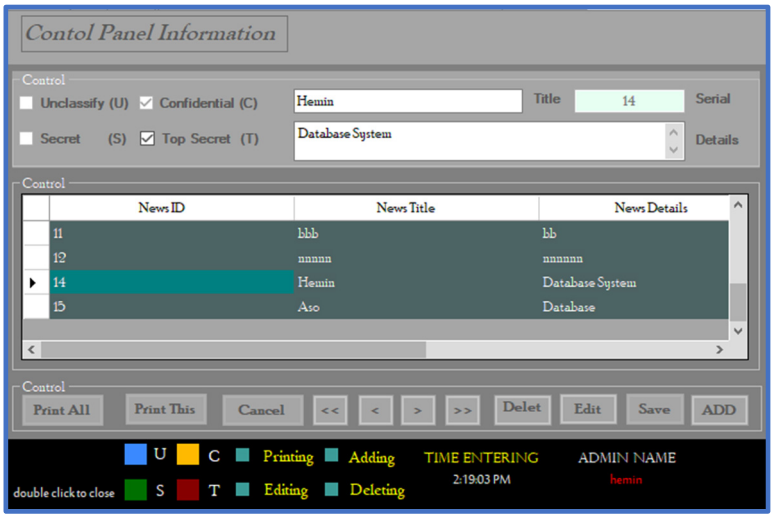

Figure 9. Control panel access information.

5. Results and Discussion

To evaluate this model's effectiveness in establishing a secure framework for database systems or any access information system, it is essential to implement a classification mechanism. This mechanism must efficiently prevent unauthorized data access through a structured authentication process that considers access levels, characteristic sizes, and performance requirements.

The evaluation involves testing the model with three distinct authentication types, as outlined in Appendix A. Each type comprises 10 unique usernames and passwords, set up to simulate real-world access scenarios. The characteristics of each type of authentication tested in the MLS/DBMS models are specified as follows:

Type 1: Usernames are usually alphanumeric, combining letters and numbers. They are generally simple and easy to remember but must be unique for each user to prevent conflicts. Passwords vary in complexity based on security needs. Simple passwords may consist of common words or short sequences, making them easy to recall but less secure.

Type 2: Usernames are typically alphanumeric and can include letters, numbers, and special characters such as hyphens or underscores. They are usually simple and easy to remember but must be unique to avoid conflicts. Passwords vary in complexity based on security needs, providing stronger protection against unauthorized access.

Type 3: Usernames are usually alphanumeric and may consist of a mix of letters, numbers, and unusual characters like hyphens, underscores, or periods. They are designed to be unique, often incorporating a combination of characters and numbers. Complex passwords typically combine uppercase and lowercase letters, numbers, and special characters to support security and prevent unauthorized access.

After addressing any issues in the setup, the model assesses the impact of each authentication type. Results indicate that the influence of these different authentication types

is substantial, validating the model's efficacy in securing access across varied authentication levels.

Table 2 presents a statistical evaluation of three distinct authentication models, assessing their performance in terms of size and time metrics. The analysis highlights the efficiency of each model, with size measured by storage requirements and time reflecting processing speed. This comparison offers valuable insights into the trade-offs between performance and security in multilevel authentication systems. Each type is assessed by cumulative (sum) and average values for these indicators, with results reflecting access constraints detailed in the Appendices. Type 2, exhibiting the highest cumulative size (464) and time (14.67 s), suggests more substantial resource usage, likely due to more intricate username–password setups. Type 1 demonstrates the lowest time averages (0.319 s), indicating a streamlined model with minimal access issues. Type 3, balanced between resource demands and response time (size of 432, time of 15.46 s), represents a moderate solution. Despite its difficulty, Type 3 may be suitable for environments valuing security and efficiency. This analysis helps inform the optimal authentication choice, factoring in resource limitations and user access needs.

Table 2. A statistical experiment evaluating three specific authentication types.

Statistical Functionality	Type1		Type2		Type3	
	Size	**Time**	**Size**	**Time**	**Size**	**Time**
Sum	408	3.1874548	464	14.6744821	432	15.4596281
Average	40.8	0.31874548	46.4	1.46744821	43.2	1.54596281

Table 3 presents a classification analysis of three authentication types, organized into four security levels: U, C, S, and T. These levels are clearly illustrated in Figure 10, highlighting the distribution of each authentication type across the security categories. The analysis emphasizes the authentication model's performance, as each appendix table verifies and supports the effectiveness of each type at different security levels. Each type demonstrates a varying distribution of levels, suggesting differences in suitability for specific security needs. Type 1 demonstrates a balanced distribution across all security levels, with the highest occurrences at U and S levels (seven authentications each). This suggests adaptability for both moderate and highly secure environments, producing an unexpectedly robust performance across varied security requirements. Type 2 also displays versatility, with slightly higher scores in the C level (7), but has the lowest frequency at Top Secret (5), which may imply moderate suitability for highly sensitive data. Type 3, in contrast, places a strong emphasis on security, achieving the highest score at the S level (8). However, it maintains relatively consistent performance across the other levels, except for the T level (5), which is an unexpected result. This outcome seems unusual, especially considering previous research on case sensitivity and improvements, where all levels, including T level, should have demonstrated stronger performance. This pattern suggests Type 3's potential for environments prioritizing data sensitivity and robust security. Overall, each authentication type shows distinct patterns in security level suitability, guiding their use based on confidentiality and resource requirements.

For further experimental evaluations, the model was analyzed as described in the methodology section. The model was integrated with cryptographic algorithms to assess four key features: key generation elapsed time, encryption elapsed time, decryption elapsed time, and memory usage during the process. For this evaluation, these features were calculated using the RSA and AES algorithms in the application without employing the MLS model. Additionally, for comparison and accuracy, the results were obtained using Python programming, as detailed in the previous section.

Table 3. Classification levels of three specific authentication types.

Authentications	Levels			
	Unclassified	**Confidential**	**Secret**	**Top Secret**
Type1	7	6	7	6
Type2	6	7	6	5
Type3	6	6	8	5

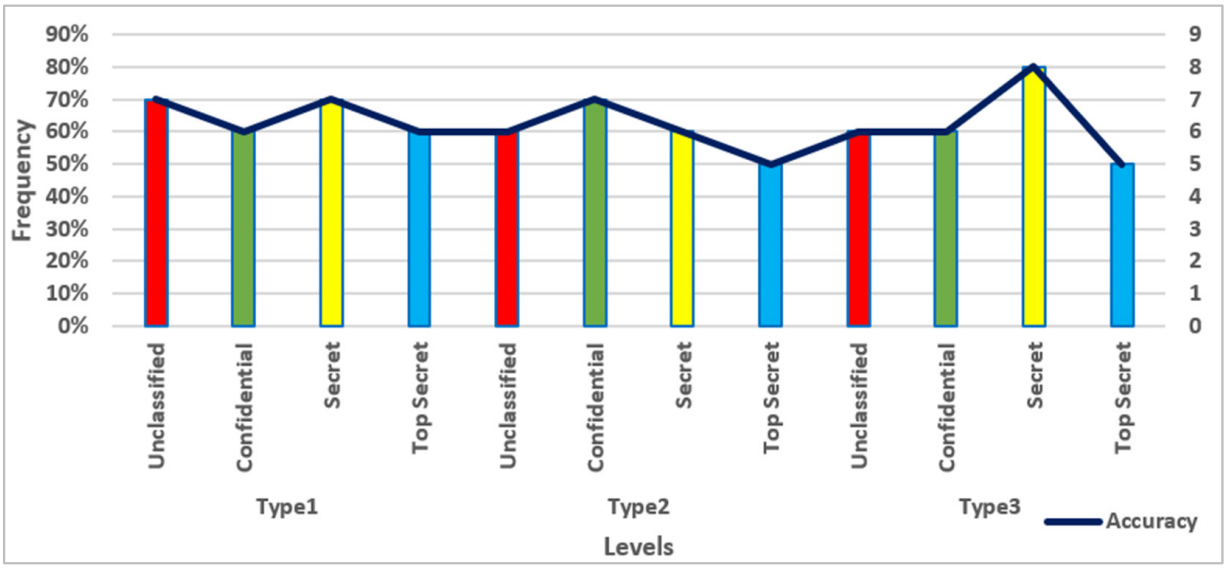

Figure 10. The level classification for accessing authentications.

Table 4 presents the results obtained without utilizing the MLS model. The results for both selected algorithms indicate the average authentication times for username and password authentication and are detailed in Appendix B. It is observed that RSA generally requires more elapsed time compared to AES. This is primarily because RSA involves more complex mathematical computations, making its key generation process more time-consuming. However, RSA is also more scalable and secure as an algorithm.

Table 4. Average performance of cryptographic algorithms without using MLS.

Features	Type1		Type2		Type2	
	RSA	**AES**	**RSA**	**AES**	**RSA**	**AES**
Key Gen Time (ms)	98.52702	0.00828	60762.8900	6.0300	153.4295	0.0117
Encryption Time (ms)	0.13183	0.07721	109.5900	69.4800	0.2448	0.1128
Decryption Time (ms)	1.34033	0.03865	1143.7800	35.3200	3.2106	0.0741
Memory Usage (bytes)	97,816,986	97,859,584	97,644,544	97,684,685	97,673,215	97,673,215

Furthermore, Type 2 authentication requires more time than the other two types due to the added complexity in managing the relationship between the access username and password. While there are slight differences in average memory usage across the algorithms, these differences are minimal and do not show significant gaps.

Table 5 presents the evaluation results using our generated model (MLS/DBMS). The results show that the AES algorithm requires less time for key generation compared to RSA, though significantly. Type 2 authentication involves more calculations but offers higher

security and is acceptable based on our model. Type 3 is also suitable for adding restrictions and aiding developers in generating DBMS or other systems.

Table 5. Average performance of cryptographic algorithms with MLS.

Features	Type1		Type2		Type2	
	RSA	AES	RSA	AES	RSA	AES
Key Gen Time (ms)	91.63013	0.007783	60674.18	5.903371	151.9719	0.011665
Encryption Time (ms)	0.127875	0.076438	108.1653	68.02092	0.190944	0.10716
Decryption Time (ms)	1.353733	0.037877	1127.767	33.55047	3.563766	0.071877
Memory Usage (bytes)	95,860,646	96,783,129	97,546,899	92,800,451	96,696,483	92,789,554

Concerning elapsed time, Type 2 in RSA shows the highest average key generation time at 60,674.18 ms, compared to Type 1 and Type 3, which have key generation times of 91.63 ms and 151.97 ms, respectively. Similarly, RSA in Type 2 requires considerably more time than AES, where RSA takes 60,674.18 ms, and AES requires only 5.90 ms.

The discussion and comparison of the results from Table 4 (with the generated model) and Table 5 (without the model) demonstrate significant improvements and optimizations. The generated model effectively reduces elapsed time and memory usage while enhancing security and accessibility for the selected algorithms.

In addition, the results using MLS show significant improvements compared to those without using the MLS/DBMS model. Thus, for RSA, key generation time reduced by 7.01% (98.53 ms to 91.63 ms in Type 1) and memory usage decreased by 2.0% (from 97,816,986 bytes to 95,860,646 bytes in Type 1). Likewise, AES encryption and decryption times improved by 1.0–5.0% across all types, enhancing efficiency.

The comparison shows that integrating MLS reduces elapsed time and memory usage, enhancing cryptographic efficiency and aligning with prior studies emphasizing MLS's optimization in secure systems. The comprehensive background review, as discussed in the review section, confirms that the evaluation of our model has demonstrated significant improvements, providing an optimal solution for the selected cryptographic algorithms. This aligns with the findings from prior research, highlighting the model's effectiveness in enhancing performance, security, and efficiency.

6. Conclusions

The created model is designed to evaluate the impact of multilevel security on database system functionality while addressing some of the fundamental challenges associated with multilevel database security. The architecture of system security, along with its legal regulations and access control strategies, is both significant and challenging. This has been analyzed through multilevel data, distributed queries, and the importance of security policies. Additionally, several challenges related to developing a secure system prototype are summarized. Accordingly, this model, known as MLS/DBMS, is designed to evaluate authentication samples and assist developers in creating a restriction prototype for a system.

Database security operates at the tuple level when using the mandatory access control paradigm. This approach combines user-level and data-level security with database security principles. By enforcing secure access standards, it ensures higher security, providing better data protection. To enhance security, two cryptographic algorithms were integrated: one based on stream ciphers and the other on block ciphers, including RSA and AES. While these algorithms provide a high level of security, they also impact performance in terms of elapsed time and memory usage.

In conclusion, performance metrics like cumulative size and elapsed time were key to the evaluation, revealing clear differences among authentication types. One type showed the highest resource usage, suitable for resource-rich environments, while another offered the fastest response times, ideal for speed-focused scenarios. The third type balanced resource demands and efficiency, making it adaptable. Additionally, incorporating metrics during the generation of cryptographic algorithms enhances the MLS/DBMS model. AES encryption and decryption times showed a 1.0–5.0% improvement across all types, boosting overall efficiency.

The developed model has shown improvement in its limitations and can be further enhanced in future studies, particularly in areas such as user experience and security robustness. Future work could focus on applying metaheuristic algorithms to find the optimal solution for the best constraints while integrating the MLS/DBMS model. Various metaheuristic algorithms, such as Fitness Dependent Optimizer (FDO) [38,39] and Lagrange Elementary Optimization (LEO) [40], could be incorporated into the model. Additionally, researchers could explore hybridizing these techniques for security enhancement within game applications or statistical algorithms [41]. Furthermore, integrating MLS/DBMS with neural networks could improve accuracy in solving authentication problems [42].

Author Contributions: Conceptualization, A.M.A.; methodology, H.S.A. and A.M.A.; software, H.S.A.; validation, H.S.A. and A.M.A.; formal analysis, A.M.A.; data curation, A.M.A.; writing—original draft preparation, H.S.A. and A.M.A.; writing—review and editing, H.S.A. and A.M.A.; visualization, A.M.A.; supervision, A.M.A.; project administration, H.S.A. and A.M.A.; funding acquisition, H.S.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

No.	Type1		Type2		Type3	
	Username	**Password**	**Username**	**Password**	**Username**	**Password**
1	aso	qwertyui	john-2024	1234@@qwer	SOPHIA	1234QWER
2	hemin	1234qwer	ali-14555	123@@@###qwe	MARIA_192024	QWER@1234
3	ahmed	12345678	william_123	qwertyasdfgh	ALICE-15433	qwER1234
4	ari	12345qwert	thomas145145	1234qwe@@@	OLIVER-2024	QWER@1234
5	sara	123456qwerty	mark-8888	qw###12345	AHMED-19822024	HHhh##1234
6	sozan	1234567qwertyu	alixander-aa777	qwer1234@@@@	nikola	qwER1234@@
7	Sali	12345678qwertyui	david98766	4321@@qqww	david_822024	2233#@ASDFgh
8	nali	(@@##qwer)	ashly_1988	rewq4321###	fatima-242482	ttrreewwqq&&1212
9	adam	12@@##qwer	maureen2018	2233@@qqwweerr	peter-12	##HHhh1234@@
10	soran	1234qwer	catherine_20242024	ttrreewwqq&&1212	ibrahim-433	112233@@##Qwer

Appendix B

No.	Type1		Type2		Type3	
	Size	Time	Size	Time	Size	Time
1	32	0.4956346	36	2.8201329	32	1.954768
2	32	0.3034645	44	2.4113905	36	1.7133282
3	32	0.2915782	48	1.8997364	32	1.4596774
4	40	0.2940378	40	2.7126597	36	1.3951995
5	48	0.299474	44	1.8111584	40	1.3558706
6	56	0.3047116	48	0.2528152	40	1.6801124
7	64	0.3021764	40	0.2929596	48	1.5526438
8	32	0.2924147	44	0.2538759	64	1.4470435
9	40	0.3018205	56	0.2497982	48	1.3205507
10	32	0.3021425	64	1.9699553	56	1.580434

References

1. Goyal, S.; Desai, P.; Swaminathan, V. Multi-Level Security Embedded with Surveillance System. *IEEE Sens. J.* **2017**, *17*, 7497–7501. [CrossRef]
2. Mishchenko, D.; Oleinikova, I.; Erdődi, L.; Pokhrel, B.R. Multidomain Cyber-Physical Testbed for Power System Vulnerability Assessment. *IEEE Access* **2024**, *12*, 38135–38149. [CrossRef]
3. Amachaghi, E.N.; Shojafar, M.; Foh, C.H.; Moessner, K. A Survey for Intrusion Detection Systems in Open RAN. *IEEE Access* **2024**, *12*, 88146–88173. [CrossRef]
4. Malik, M.S. IoT Malware: A Comprehensive Survey of Threats, Vulnerabilities, and Mitigation Strategies. *Int. J. Electron. Crime Investig.* **2024**, *8*, 57–66. [CrossRef]
5. Wang, X.; Yan, Z.; Zhang, R.; Zhang, P. Attacks and Defenses in User Authentication Systems: A Survey. *J. Netw. Comput. Appl.* **2021**, *188*, 103080. [CrossRef]
6. Liu, K.; Xu, G.; Cao, Q.; Wang, C.; Jia, J.; Gao, Y.; Xu, G. A Rivest–Shamir–Adleman-Based Robust and Effective Three-Factor User Authentication Protocol for Healthcare Use in Wireless Body Area Networks. *Sensors* **2023**, *23*, 8992. [CrossRef] [PubMed]
7. Banerjee, S.; Patil, A. ECC Based Encryption Algorithm for Lightweight Cryptography. In *Intelligent Systems Design and Applications*; Springer: Cham, Switzerland, 2020; pp. 600–609.
8. Ghosh, A. Comparison of Encryption Algorithms: AES, Blowfish and Twofish for Security of Wireless Networks. *Int. Res. J. Eng. Technol.* **2020**, *7*, 4656–4658.
9. Pranjic, M.; Fertalj, K.; JukiC, N. Importance of Semantics in MLS Database Models. In Proceedings of the ITI 2002. Proceedings of the 24th International Conference on Information Technology Interfaces (IEEE Cat. No. 02EX534), Cavtat, Croatia, 24–27 June 2002; pp. 51–56.
10. Singla, D.; Verma, N. Performance Analysis of Authentication System: A Systematic Literature Review. *Recent Adv. Comput. Sci. Commun.* **2024**, *17*, 47–67. [CrossRef]
11. Kavitha, G.; Prasannakumar, V.; Pranav, R.P. Enhancing Digital Security: A Comprehensive Multi-Model Authentication Framework Leveraging Cryptography and Biometrics. In Proceedings of the 2024 8th International Conference on Inventive Systems and Control (ICISC), Coimbatore, India, 29–30 July 2024; pp. 476–486.
12. Dorairaj, S.D.; Kaliannan, T. An Adaptive Multilevel Security Framework for the Data Stored in Cloud Environment. *Sci. World J.* **2015**, *2015*, 601017. [CrossRef] [PubMed]
13. Budati, A.K.; Vulapula, S.R.; Shah, S.B.H.; Al-Tirawi, A.; Carie, A. Secure Multi-Level Privacy-Protection Scheme for Securing Private Data over 5G-Enabled Hybrid Cloud IoT Networks. *Electronics* **2023**, *12*, 1638. [CrossRef]
14. Fertalj, K.; Jukić, N.; Pranjić, M. Implementing Belief-Consistent Multilevel Secure Relational Data Model: Issues and Solutions. *J. Comput. Inf. Technol.* **2003**, *11*, 225–232.
15. Lin, H.Y. Secure Data Transfer Based on a Multi-Level Blockchain for Internet of Vehicles. *Sensors* **2023**, *23*, 2664. [CrossRef] [PubMed]
16. Jiang, M.; Yang, H. Image Encryption Algorithm Using Multi-Level Permutation and Improved Logistic–Chebyshev Coupled Map. *Information* **2023**, *14*, 456. [CrossRef]
17. Heckman, M.R.; Schell, R.R. Using Proven Reference Monitor Patterns for Security Evaluation. *Information* **2016**, *7*, 23. [CrossRef]

18. Anderson, R. *Security Engineering: A Guide to Building Dependable Distributed Systems*, 3rd ed.; John Wiley & Sons: Hoboken, NJ, USA, 2020; ISBN 978-0-470-06852-6.
19. Wang, M.; Carr, S.; Mayo, J.; Shene, C.-K.; Wang, C. MLSvisual: A Visualization Tool for Teaching Access Control Using Multi-Level Security. In Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education, Uppsala, Sweden, 21–25 June 2014; pp. 93–98.
20. Xue, H.; Zhang, Y.; Guo, Z. A Multilevel Security Model for Private Cloud. *Chin. J. Electron.* **2014**, *23*, 232–235. [CrossRef]
21. Sallam, A.I.; El-Rabaie, E.-S.; Faragallah, O.S. Encryption-Based Multilevel Model for DBMS. *Comput. Secur.* **2012**, *31*, 437–446. [CrossRef]
22. Kumar, A.; Verma, G. Multi-Level Authentication for Security in Cloud Using Improved Quantum Key Distribution. *Netw. Comput. Neural Syst.* 2024; *in press*. [CrossRef]
23. Sunil Kumar Aithal, S.; Rajashree. Multi-Privacy Level Encryption Technique For Secure Cloud-Based Data Repositories. In Proceedings of the 2024 International Conference on Knowledge Engineering and Communication Systems (ICKECS), Chikkaballapur, India, 18–19 April 2024; pp. 1–6.
24. Soundararajan, R.; Rajagopal, M.; Muthuramalingam, A.; Hossain, E.; Lloret, J. Interleaved Honeypot-Framing Model with Secure MAC Policies for Wireless Sensor Networks. *Sensors* **2022**, *22*, 8046. [CrossRef]
25. Ali, A.; Pasha, M.F.; Ali, J.; Fang, O.H.; Masud, M.; Jurcut, A.D.; Alzain, M.A. Deep Learning Based Homomorphic Secure Search-Able Encryption for Keyword Search in Blockchain Healthcare System: A Novel Approach to Cryptography. *Sensors* **2022**, *22*, 528. [CrossRef] [PubMed]
26. Mhamdi, H.; Ayadi, M.; Ksibi, A.; Al-Rasheed, A.; Soufiene, B.O.; Hedi, S. SEMRAchain: A Secure Electronic Medical Record Based on Blockchain Technology. *Electronics* **2022**, *11*, 3617. [CrossRef]
27. Lee, T.-F.; Chang, I.-P.; Kung, T.-S. Blockchain-Based Healthcare Information Preservation Using Extended Chaotic Maps for HIPAA Privacy/Security Regulations. *Appl. Sci.* **2021**, *11*, 10576. [CrossRef]
28. Yu, C.; Zhan, Y.; Sohail, M. SDSM: Secure Data Sharing for Multilevel Partnerships in IoT Based Supply Chain. *Symmetry* **2022**, *14*, 2656. [CrossRef]
29. Thorleuchter, D.; Van den Poel, D. Improved Multilevel Security with Latent Semantic Indexing. *Expert. Syst. Appl.* **2012**, *39*, 13462–13471. [CrossRef]
30. Muhammed, R.K.; Aziz, R.R.; Hassan, A.A.; Aladdin, A.M.; Saydah, S.J.; Rashid, T.A.; Hassan, B.A. Comparative Analysis of AES, Blowfish, Twofish, Salsa20, and ChaCha20 for Image Encryption. *Kurd. J. Appl. Res.* **2024**, *9*, 52–65. [CrossRef]
31. Akogun, D.N. Enhancing Data Security in Cloud Storage Using Residue Number System and Advanced Encryption Standard. Master's Thesis, ProQuest Dissertations & Theses, Kwara State University (Nigeria), Kwara, Nigeria, 2020.
32. Modugula, R.S.R. A Hybrid Approach for Augmenting Password Security Using Argon2i Hashing and AES Scheme. Master's Thesis, National College of Ireland, Dublin, Ireland, 2020.
33. Sharma, A.; Thapliyal, S.; Wazid, M.; Mishra, A.K.; Kumar, P.; Giri, D. A Secure Mechanism for Password Hash Value Generator with the Security Analysis of Various Hashing Algorithms. In Proceedings of the 2024 4th International Conference on Computer, Communication, Control & Information Technology (C3IT), Hooghly, India, 28–29 September 2024; pp. 1–6.
34. Venkateswarlu, I.B.; Kakarla, J. Password Security by Encryption Using an Extended ADFGVX Cipher. *Int. J. Inf. Comput. Secur.* **2019**, *11*, 510. [CrossRef]
35. Linux, E.; Linux, R.H.E. Red Hat Enterprise Linux 8. *Europe* **2019**, *800*, 2835.
36. Cristiá, M.; Rossi, G. Automated Proof of Bell–LaPadula Security Properties. *J. Autom. Reason.* **2021**, *65*, 463–478. [CrossRef]
37. Gorbachov, V.; Batiaa, A.K.; Ponomarenko, O.; Kulak, E. Securing Computer Hardware on the Base of Reference Monitor Obfuscation. In Proceedings of the 2018 International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T), Kharkiv, Ukraine, 9–12 October 2018; pp. 406–410.
38. Abdullah, J.M.; Ahmed, T. Fitness Dependent Optimizer: Inspired by the Bee Swarming Reproductive Process. *IEEE Access* **2019**, *7*, 43473–43486. [CrossRef]
39. Aladdin, A.M.; Abdullah, J.M.; Salih, K.O.M.; Rashid, T.A.; Sagban, R.; Alsaddon, A.; Bacanin, N.; Chhabra, A.; Vimal, S.; Banerjee, I. Fitness-Dependent Optimizer for IoT Healthcare Using Adapted Parameters: A Case Study Implementation. In *Practical Artificial Intelligence for Internet of Medical Things*; CRC Press: Boca Raton, FL, USA, 2023; pp. 45–61.
40. Aladdin, A.M.; Rashid, T.A. Leo: Lagrange Elementary Optimization. *arXiv* **2023**, arXiv:2304.05346.

41. Amin, A.A.H.; Aladdin, A.M.; Hasan, D.O.; Mohammed-Taha, S.R.; Rashid, T.A. Enhancing Algorithm Selection through Comprehensive Performance Evaluation: Statistical Analysis of Stochastic Algorithms. *Computation* **2023**, *11*, 231. [CrossRef]
42. Polat, O.; Türkoğlu, M.; Polat, H.; Oyucu, S.; Üzen, H.; Yardımcı, F.; Aksöz, A. Multi-Stage Learning Framework Using Convolutional Neural Network and Decision Tree-Based Classification for Detection of DDoS Pandemic Attacks in SDN-Based SCADA Systems. *Sensors* **2024**, *24*, 1040. [CrossRef]