A Trust-Driven Optimization Model for Reliable Authorization in Hadoop Environment

BATTAT Nadia[1*] and MAKHOUL Abdallah[2]

[1*]Department of Computer Science, LIMED Laboratory, University of Bejaia, Targa Ouzemour, Bejaia, 06000, Bejaia, Algeria.
[2]Department of Computer Science, Université Marie et Louis Pasteur, CNRS, institut FEMTO-ST, Franche-Comté, F-25200, Montbéliard, France.

*Corresponding author(s). E-mail(s): nadia.battat@univ-bejaia.dz;
Contributing authors: abdallah.makhoul@univ-fcomte.fr;

Abstract

Authorization is a fundamental service in the Hadoop environment. It ensures data and protocol security while granting access exclusively to authorized users, allowing them to perform operations such as data storage, processing, and file transfer in a controlled manner. However, maintaining effective authorization remains challenging due to vulnerabilities in the Hadoop ecosystem and potential abnormal user behavior. Various authorization protocols have been developed based on different assumptions about the Hadoop environment, each aiming to ensure reliable and efficient access control through distinct techniques. While these techniques have their strengths and limitations, authorization can be further enhanced by integrating additional security mechanisms. In this work, we propose a trust-based authorization model to assist the NameNode in making access control decisions, particularly when handling uncertain or untrusted authenticated users. This model addresses two key questions: (1) should the authenticated user be authorized to perform the requested operation? (2) how should the request be executed? Our approach leverages machine learning alongside existing security tools, such as monitoring systems and intrusion detection mechanisms (IDS), to identify intrusions or unauthorized operations. By continuously analyzing incoming requests and monitoring the Hadoop environment, our method enhances security while maintaining operational efficiency. Additionally, we integrate Elephant Herding Optimization (EHO) to enable a distributed and adaptive access control mechanism, optimizing user satisfaction, minimizing security risks, and reducing resource consumption. The proposed model is evaluated against a generic authorization framework based on Role-Based Access

Control (RBAC) and Attribute-Based Access Control (ABAC) using different performance metrics. Experimental results demonstrate the effectiveness of our approach in detecting unauthorized access. Furthermore, the machine learning-based intrusion detection method is assessed using widely recognized datasets, considering key metrics such as accuracy, recall, precision, F1-score, specificity, AUC-ROC, and execution time. The results illustrate that the applied ML techniques are effective and can achieve optimal performance.

Keywords: Authentication, Authorization, Big Data, EHO, Hadoop, intrusion, machine-learning, trust.

1 Introduction

Big Data is an information-based hardware and software technology that operates through five phases (collection, storage, analytics, utilization, and destruction) to process vast amounts of data in various formats (structured, semi-structured, and unstructured). This data can be gathered from heterogeneous sources (e.g., remote sensors). The use of Big Data provides an effective solution for delivering valuable services across multiple domains, including healthcare, marketing, industry, and scientific research [1] [2].

Apache Hadoop is one of the most widespread Big Data storage and processing platforms, developed by the Apache Foundation [3]. This open-source framework consists of specific components, namely the Hadoop Distributed File System (HDFS), MapReduce, and Yet Another Resource Negotiator (YARN), which manage resources and services within at least one cluster. Each cluster consists of a Name Node, a Secondary Name Node (which takes over in case of a Name Node failure), and at least one Data Node [1][4]. The Name Node partitions files into blocks and replicates them across selected Data Nodes, ensuring fault tolerance in the Hadoop environment. It also provides users with the location of the requested Data Node containing the required data or services. Communication between the Name Node and Data Nodes is performed using a heartbeat mechanism [5].

Table 1 Features of the Hadoop ecosystem

Implementation language	Java.
Architecture Model	Master-slave.
Scheduling	Fair, FIFO, capacity.
Latency (s)	High.
Throughput	High.
Scalability	High.
Fault detection	Heartbeat mechanism.
Fault Recovery	Data replication.
Best-fit application use cases	Applications that require batch processing of very large datasets.

Table 1 presents the main features of the Hadoop ecosystem [6]. However, despite its popularity, this framework faces multiple security challenges [1]. In this paper, we focus on the issue of authorization. Authorization is a crucial process required to safeguard both the Hadoop ecosystem and the privacy of participating entities. It verifies whether a given entity (user, system, etc.) is permitted to access data and perform specific operations [7].

This process cannot be effectively carried out without first authenticating the given entity and its requested services [1] [8]. Initially, Hadoop relied on the traditional Kerberos protocol, which employs a symmetric encryption system for authentication verification [9]. Over time, various versions of the Kerberos protocol have been developed to support both authentication and authorization within the Hadoop environment [10]. The most frequently cited authentication protocols include those proposed by [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], and [22]. Authentication methods can also serve as authorization mechanisms. For instance, Kerberos allows access control management by defining ACLs to regulate user access to specific resources.

Several protocols have been introduced in the literature to achieve reliable authorization while addressing the nature and requirements of the Hadoop environment. These protocols employ different mechanisms, each with distinct advantages and limitations. Based on their mechanisms, we classify these protocols into two categories: access control policy-based protocols and irregular behavior detection-based protocols.

To fulfill the objectives of authorization protocols and ensure compliance in a legitimate and lawful manner, we propose a novel intelligent trust model called ITM-RAH (Intelligent Trust Model-based Reliable Authorization in Hadoop). This model combines RBAC (Role-Based Access Control) and ABAC (Attribute-Based Access Control) authorization mechanisms while integrating various security tools, including intrusion detection systems (IDS), monitoring systems, and machine learning-based IDS, to classify users as trusted, uncertain, or untrusted and to detect potential intrusions. The decision-making process for service execution is optimized using the Elephant Herding Optimization (EHO) method, which considers resource consumption, security risks, and user satisfaction.

To evaluate our approach, we have compared it with a generic RBAC and ABAC authorization model rather than a specific solution like Apache Ranger. This ensures that our evaluation remains independent of any particular implementation or policy framework, making it adaptable to various authorization environments.

The key contributions of this work are as follows:

1. A classification of existing authorization protocols into two main categories: access control policy-based protocols and irregular behavior detection-based protocols.
2. The proposal of a trust-driven optimization model to enhance authorization reliability, categorize Hadoop users as trusted, uncertain, or untrusted, and compare this approach with a generic authorization service leveraging RBAC and ABAC, while considering different evaluation metrics.
3. The development of a new dataset designed to improve intrusion detection.
4. The evaluation of the proposed multi-objective optimization method based on Elephant Herding Optimization.

5. The evaluation of a machine learning-based intrusion detection system, assessing its effectiveness using metrics such as accuracy, recall, precision, F1-score, specificity, AUC-ROC, and simulation time.

The remainder of this paper is structured as follows: Section 2 discusses the authorization challenges in the Hadoop environment. Section 3 provides an overview of related work on authorization. Section 4 describes the proposed model. Section 5 presents the experimental evaluation of the model. Section 6 analyzes our proposal against several criteria. Finally, Section 7 concludes the paper and outlines future research directions.

2 Background

The Hadoop ecosystem has become one of the most powerful and widely adopted frameworks for Big Data analytics. Its flexibility and scalability make it a preferred choice for businesses seeking to process vast amounts of data efficiently while integrating additional technologies to meet their specific needs. However, the effectiveness of this environment heavily relies on the robustness of its authorization mechanisms. Authorization plays a crucial role in defining the level of access granted to an entity and determining which digital resources and services can be used.

2.1 Challenges of Authorization Protocols

Numerous authorization protocols have been developed to enhance the efficiency and reliability of Hadoop's access control mechanisms. A well-designed authorization protocol must address the following challenges:

- **Availability and Fault Tolerance:** The protocol must include mechanisms to ensure continuous availability. This can be achieved through redundancy in both policy management and software components.
- **Scalability:** A robust authorization protocol should accommodate a growing number of users without compromising performance.
- **Security:** Security is a primary concern for both industrial and academic applications [23] [24] [25]. Essential security attributes include:
  - **Authentication:** Only authenticated entities should interact with the authorization protocol.
  - **Confidentiality:** Access control policies and sensitive data must remain protected from unauthorized entities.
  - **Integrity:** Any modifications to the protocol's core functionality must be detected and addressed.
- **Correctness:** The protocol must operate within time constraints and always provide a definitive authorization decision.
- **Reliability:** It should account for malicious behavior from users and system components (Name Node, Data Nodes) to maintain trustworthiness.

- **Resource consumption and performance:** The protocol must minimize computational overhead, ensuring that additional security mechanisms do not impact Hadoop's overall performance.

2.2 Classification of Hadoop Users

In [26], Hadoop users are categorized as regular users, business users, scientific researchers, developers, or malicious users. In our model, we classify users into three distinct categories:

- **Trusted users:** Authenticated entities who correctly execute authorized Hadoop operations, such as reading, writing, processing, transferring, and storing data.
- **Untrusted users:** Entities attempting unauthorized operations to gain privileges or attack the Hadoop ecosystem.
- **Uncertain users:** Entities with inconsistent behavior, sometimes following expected usage patterns while occasionally engaging in irregular activities.

Beyond ensuring confidentiality and integrity, malicious user behavior also threatens the privacy of Hadoop users. Undetected threats can undermine the ecosystem's performance, compromise data security, and erode user trust, leading to faulty authorization decisions.

Table 2 outlines the various malicious actions that can be carried out within Hadoop environments.

Table 2 Malicious behaviors of Hadoop users

Tasks	Malicious behaviors
Authentication task	Perform failed authentication attempts, achieve unauthorized access, etc.
HDFS-related tasks	Reading sensitive metadata (e.g., accessing via HTTP protocol), changing file permissions, accessing a file via HTTP protocols, disrupting the operation of HDFS, etc.
MapReduce-related tasks	Execute dfsadmin commands, perform unauthorized operations via remote procedure call (RPC), execute/install malware, escalate privileges, modify job priorities, delete or update job code, disrupt the operation of MapReduce, etc.
Communication tasks	Intercept traffic exchanged between participating entities (Name Node, Data Nodes, users), perform replay attacks, etc.
Name Node tasks	Man-in-the-middle and message fabrication attacks, injection, etc.
Other malicious tasks	Erase or modify log entries, use developer tools, modify the security system or installed software, etc.

2.3 Traditional Security Tools

Several conventional security tools have been proposed in the literature to detect threats and attacks, including monitoring systems, anti-malware programs, antivirus solutions, firewalls, and intrusion detection systems (IDS).

- Monitoring systems (e.g., Ganglia, Nagios) continuously analyze system activity to improve the reliability of authorization mechanisms by tracking user behavior.
- IDS solutions, which detect security breaches, generally employ one of two techniques:
  - **Signature-based detection:** Identifies known threats by matching observed patterns with predefined attack signatures.
  - **Anomaly-based detection:** Flags unusual activities based on deviations from established behavioral norms.
- Firewalls complement IDS solutions by preventing unauthorized access to network resources.

However, traditional security measures often struggle to detect evolving threats such as Distributed Denial-of-Service (DDoS) attacks or sophisticated new malware. In fact, advanced techniques, such as machine learning-based approaches, have been developed to enhance threat detection.

2.4 Authorization Process Using an Intelligent Trust Model

Machine learning (ML) is a subset of artificial intelligence that enables systems to learn patterns from data and make informed decisions without explicit programming [27]. ML-based intrusion detection systems (IDS) analyze historical network traffic to identify and flag security threats in real-time. Unlike signature-based systems, ML-based IDS can detect previously unseen attacks by continuously improving through new data and feedback from security analysts.

In this paper, we propose an intelligent trust model to enhance authorization service in Hadoop environments. Figure 1 illustrates how our proposed model assists the Name Node in making authorization decisions when receiving a service request message.

The proposed model operates at the Name Node level and leverages both traditional security tools (monitoring systems, IDS) and ML-based IDS for dynamic trust assessment. ML-based IDS continuously evaluate incoming service requests to update trust and untrust values, while traditional security tools refine these values based on detected anomalies.

The increasing complexity of Hadoop authorization necessitates decision-making processes that balance three key factors: security risk, resource consumption, and user satisfaction. Security risks stem from potential threats, while resource consumption is related to computational overhead. At the same time, usability considerations improve user trust and efficiency.

We employ the Elephant Herding Optimization (EHO) algorithm to achieve an optimal balance between these competing objectives. EHO's iterative search capabilities enable dynamic resource allocation, ensuring that Hadoop's authorization process remains secure, scalable, and efficient while preserving system performance.

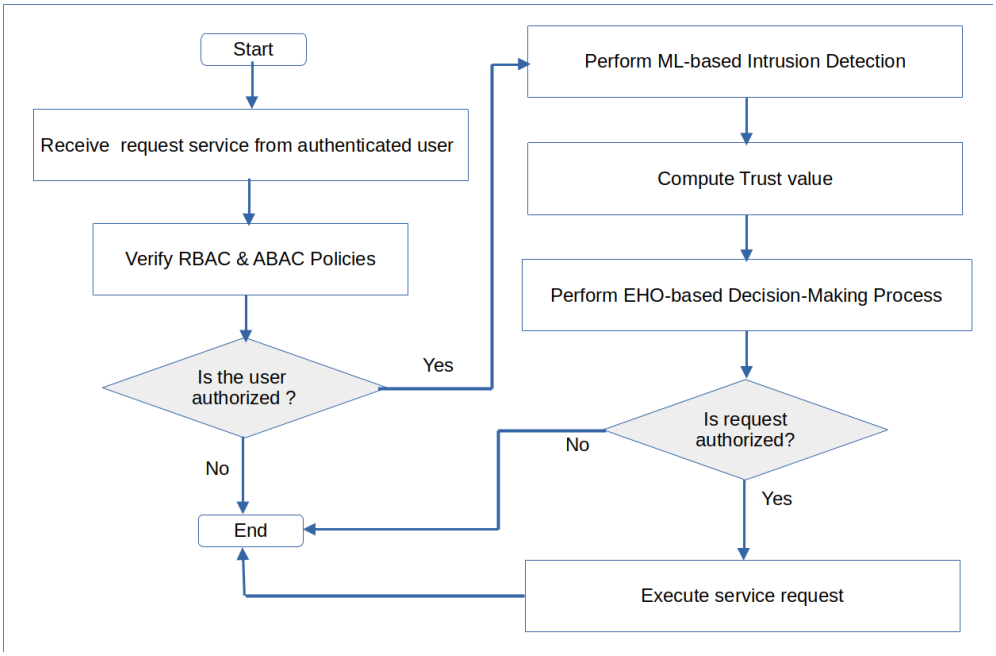

Fig. 1 Authorization service using the proposed model

3 Related Work

Ensuring secure authorization in the Hadoop ecosystem remains a critical challenge. Initially, Hadoop security mechanisms focused primarily on preventing unintentional data loss, neglecting the risks associated with unauthorized access to HDFS, resources, and services. The introduction of file permission mechanisms in HDFS provided a foundational layer of security by defining user-specific access rights, but this approach was insufficient against sophisticated attacks, insider threats, and advanced persistent threats (APTs).

To address these concerns, several authorization protocols have been developed with the aim of establishing secure and reliable access control in Hadoop environments. These protocols rely on different mechanisms and present different advantages and disadvantages. Based on the authorization mechanism used, we propose to classify these protocols as follows.

- **Access Control Policies-Based Protocols**: These protocols verify whether an entity has permission to access specific servers or resources by cross-referencing an access control list or credentials [28]. Researchers primarily focus on the following access control policy models [29]:
  - **Role-Based Access Control (RBAC)**: This model bases decisions about resource access on the roles that different users perform within the Hadoop ecosystem. RBAC was initially considered an effective way to control user permissions.

By using roles, administrative work is reduced and organizations gain greater flexibility in defining access restrictions. However, due to the rapid expansion of data and users in Big Data applications, RBAC alone is no longer sufficient.
  - **Attribute-Based Access Control (ABAC)**: This model assesses attributes to determine who can access resources. Traditional models such as Mandatory Access Control (MAC), Discretionary Access Control (DAC), and RBAC often lack the granularity and flexibility needed for modern systems. ABAC overcomes these limitations by leveraging real-time data and enabling more accurate and context-aware decisions based on various attributes. However, improper handling of attributes can lead to confusion and complexity.

- **Irregular Behavior Detection-Based Protocols**: In addition to predefined access control policies, the NameNode considers the irregular behavior of participating entities to control access to resources and services. Malicious entities can perform unauthorized operations, such as injecting malicious code, falsifying unencrypted data, intercepting, deleting, or modifying Hadoop ecosystem functionalities. Such irregular behaviors pose significant threats, potentially impacting system performance, compromising user privacy, and causing economic damage.

3.1 Access Control Policies-Based Protocols

Traditional authorization mechanisms in Hadoop focus on policy enforcement using predefined rules. Although these models provide a structured approach to access control, they are often static and struggle to deal with evolving security threats. Below, we review key policy-based authorization mechanisms.

Apache Knox: Apache Knox Gateway provides a centralized authentication and authorization framework for Hadoop clusters. It supports LDAP and Active Directory integration, offering identity federation and secure HTTP-based access to Hadoop services. While Knox simplifies user management and secures perimeter access, it does not implement fine-grained access control at the resource level, making it insufficient for granular security policies.

Apache Sentry: Apache Sentry enforces fine-grained authorization by applying predefined access control policies at the database, table, and column levels. It is widely used in Hadoop distributions such as Cloudera CDH, allowing administrators to define role-based policies. However, Sentry lacks dynamic adaptability, meaning unauthorized queries or privilege escalation attacks cannot be detected in real time [5].

Apache Ranger: Apache Ranger provides a centralized policy management framework for Hadoop, integrating role-based access control (RBAC) and Kerberos authentication. It enables administrators to define granular access policies, supporting fine-grained authorization across HDFS, Hive, HBase, and other Hadoop components. However, Ranger's reliance on static policies makes it vulnerable to insider threats, as it lacks real-time behavioral monitoring to detect anomalous access patterns.

To overcome the limitations of RBAC-based models, recent works have explored hybrid approaches that integrate attribute-based policies.

HBD-Authority (Hybrid Access Control Model): Proposed in [30], HBD-Authority combines the Truman and Non-Truman models to enforce separation-of-duty (SoD)

principles. This hybrid approach enhances access transparency but does not incorporate adaptive trust assessment, making it less effective against insider attacks.

H-RABAC (Role-Attribute-Based Access Control): Introduced in [29], H-RABAC merges RBAC and ABAC to enable dynamic access control based on user attributes and data sensitivity. While it improves flexibility, it still lacks real-time threat detection capabilities, leaving it susceptible to compromised credential misuse.

Access Control Policies-Based Protocols, including Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC), provide well-structured and predefined mechanisms for regulating access permissions. These models rely on explicit policies that assign rights to users or roles, ensuring a clear and deterministic access management framework. While policy-based approaches are widely adopted due to their simplicity and administrative control, they exhibit critical limitations in modern, large-scale environments like Hadoop:

- Lack of adaptability: These mechanisms do not dynamically adjust access rights based on evolving user behavior or changing threat landscapes. Once permissions are granted, they remain static unless manually modified, making them vulnerable to account compromise or privilege escalation attacks.
- Complex policy management: In large distributed systems, defining and maintaining fine-grained access control rules becomes increasingly complex, especially when dealing with a high volume of users, datasets, and services. Misconfigurations or outdated policies may inadvertently create security gaps.
- Inability to detect malicious insiders: Policy-based mechanisms primarily focus on enforcing predefined rules rather than detecting deviations in user behavior. As a result, they fail to prevent unauthorized access attempts by legitimate but compromised accounts.

3.2 Irregular Behavior Detection-Based Protocols

The security of distributed environments, such as Hadoop, relies heavily on effective authorization mechanisms to regulate access and prevent unauthorized operations. Traditional access control mechanisms operate on predefined policies and static authentication procedures, which fail to detect unauthorized activities once initial access is granted. To mitigate this limitation, researchers have explored dynamic, anomaly-based authorization models capable of adapting to evolving user behaviors.

In [31], the authors introduce an authorization model integrating Attribute-Based Access Control (ABAC) with an intermediary access proxy. This proxy dynamically assigns trust threshold values to each resource based on its sensitivity level, thereby enforcing fine-grained access restrictions. While this approach enhances access control granularity, it relies primarily on access attempts for trust evaluation, which remains insufficient. Incorporating additional behavioral attributes—such as access frequency, command execution patterns, or deviation from historical user activity—could significantly improve its robustness.

A security framework proposed in [32] combines Kerberos authentication with Transport Layer Security (TLS) to secure HDFS data against replay and data injection attacks. While this integration strengthens communication security, it inherits the

inherent weaknesses of both Kerberos and TLS. These include complex key management, susceptibility to credential theft, and exposure to sophisticated attack vectors such as Kerberos ticket forgery and TLS downgrade attacks.

In [5], a sandboxing mechanism is implemented to restrict the operations performed by authenticated users, effectively containing potential threats within controlled execution environments. This approach limits unauthorized privilege escalation and unintended data modifications. However, its effectiveness could be further improved by integrating runtime monitoring and predictive security analytics to detect and mitigate emerging attack patterns.

In [26], the author proposes an intrusion detection system (IDS) that leverages artificial neural networks (ANNs) and an on-demand honeypot mechanism to detect unauthorized operations. This approach suggests dynamically deploying honeypots in response to identified anomalous behaviors, with the aim of improving threat detection and system security.

In [33], the authors propose a security framework for the Hadoop ecosystem based on Attribute-Based Access Control (ABAC) to safeguard sensitive and private data stored in datasets while mitigating various cybersecurity threats. Their analysis highlights three essential components required to establish robust operational security: the mandatory use of a secure data analyst notebook, the integration of a dedicated data security service layer, and the deployment of a centralized management console for policy enforcement and continuous monitoring.

Irregular behavior-based approaches represent a promising direction. By integrating access control policies with dynamic anomaly detection, these protocols aim to provide a more adaptive and resilient security mechanism. However, they also face key challenges, particularly their strong dependency on the selected irregular behavior detection mechanism.

Through the analysis of existing protocols, we identify opportunities to enhance authorization mechanisms by integrating advanced irregular behavior detection techniques for authenticated users. In fact, we propose a novel approach that combines RBAC, ABAC, and adaptive authorization methods, creating a more resilient security framework for Hadoop environments. By continuously evaluating user trust levels and leveraging machine learning for anomaly detection, this solution aims to strengthen protection against unauthorized access while ensuring adaptability to evolving threats.

- Dynamic trust evaluation: Unlike static policy-based methods, our model continuously assesses user behavior and assigns trust levels (trusted, uncertain, untrusted) to dynamically adjust access privileges. This helps mitigate risks associated with compromised accounts and unauthorized privilege escalation.
- Machine learning-based anomaly detection: By training a classification model on user interaction patterns, our approach identifies deviations that may indicate malicious activities. This allows for early threat detection and proactive security enforcement.

4 ITM-RAH

The concept of trust has been proposed as a means of managing malevolent users [34]. Generally, trust is defined as a strong belief in each participating entity's ability to correctly perform the required tasks [35]. In our context, we assume that the Name Node evaluates the trustworthiness of other users in the Hadoop environment. We define trust as a strong conviction in a user's ability to consistently, securely, and reliably perform necessary tasks (such as authentication) while ensuring that transmitted messages remain free of malicious code.

In [35], the authors cited the most commonly used trust management models (TMM), which are: Policy-based TMM, Reputation-based TMM, Network-based TMM, Game theory-based TMM, Fuzzy-based TMM, and ML-based TMM. In our context, we propose to use Reputation-based and ML-based TMM.

- Reputation-based TMM: Trust is estimated based on the reputation of participating users while monitoring their behaviors and actions.
- ML-based TMM: Trust is estimated based on the outputs of machine learning-based techniques.

4.1 Trust value evaluation

Effective access control in large-scale distributed systems requires a trust-based evaluation that dynamically adapts to user behavior. The proposed model assigns a trust value to each user, serving as a decision making criterion for granting or restricting access to system resources. This trust value is computed based on three key metrics: authentication success, behavioral consistency (honesty), and historical trust scores. These components ensure a scalable, real-time assessment mechanism that seamlessly integrates into enterprise security frameworks.

1. **Successful Authentication (SA)**: Authentication success rate is a primary determinant of trust, reflecting the user's ability to provide valid credentials efficiently. Within a defined evaluation window (epoch e), authentication reliability is computed as:

$$SA_e = \begin{cases} 1, & \text{if } ET = 0 \\ 1 - \frac{ET-1}{ET}, & \text{if } 1 \leq ET \leq 3 \\ 0, & \text{if } ET > 3 \end{cases} \quad (1)$$

   where ET represents the number of authentication attempts required before success. A threshold-based policy enforces a maximum of three attempts per epoch, after which authentication is considered failed, setting SA_e to zero.

   To gather authentication process data, the proposal can continuously process authentication logs generated by integrated security services, such as centralized access control policies, Kerberos-based authentication, and extended identity management modules. These logs provide real-time data on authentication attempts, including timestamps, source IP addresses, and authentication outcomes.

2. **Honesty (H)**: Beyond authentication success, assessing user trust requires monitoring behavioral patterns over time. The honesty metric (H) is derived from a

combination of anomaly detection algorithms that classify user actions based on historical and contextual activity.

The computation of H involves aggregating detection outputs from multiple machine learning models operating in parallel. The honesty score is normalized between 0 (highly suspicious or irregular behavior) and 1 (legitimate and expected activity).

The calculation process is outlined in Algorithm 1.

Algorithm 1 Honesty Metric Calculation

Input: N_{ML} (Number of ML algorithms)
Initialize: $Max \Leftarrow 0$
for each subset do
$N_{DA} \Leftarrow 0$
for each ML algorithm **do**
    Compute ML output MLO
    if $MLO = 1$ **then**
        $N_{DA} \Leftarrow N_{DA} + 1$
    end if
end for
$Max \Leftarrow \max(Max, N_{DA})$
end for
Compute honesty metric: $H \Leftarrow 1 - (Max/N_{ML})$
Output: H

3. **Previous Trust Value (PTV)**: To ensure stability in access control decisions, the model integrates historical trust scores. This prevents abrupt fluctuations in trust assessments due to isolated anomalies. The previous trust value (PTV) acts as a stabilizing factor, ensuring continuity in security policies while mitigating the risk of transient misbehavior unfairly impacting access privileges.

The overall trust value for an evaluation epoch e is computed as:

$$TV_e = a \cdot SA_e + b \cdot H_e + c \cdot PTV \tag{2}$$

where a, b, c are weight coefficients such that $a + b + c = 1$. The values of these coefficients can be dynamically adjusted based on system policies.

Additionally, we compute the untrust value (UTV_e) as : ($UTV_e = 1 - TV_e$).

4.2 Trust value update and storage

The trust value can evolve based on real-time monitoring of user activities and security alerts. Various defense mechanisms, such as intrusion detection systems (IDS) and

monitoring tools continuously assess system activity. When these mechanisms detect potential threats linked to a specific user, the trust value is updated accordingly.

To ensure effective trust management, the system relies on the continuous analysis of system logs, which provide crucial insights into authentication attempts, resource access patterns, and detected security incidents. These logs are collected from various components of the Hadoop ecosystem, including:

- Authentication logs : Tracking login attempts, successful sessions, and failed authentication events.
- Access control logs: Recording interactions with HDFS, YARN, and other Hadoop services.
- Intrusion detection system (IDS) logs: Detecting network anomalies, exploit attempts, and unauthorized actions.
- System resource monitoring logs: Identifying abnormal CPU, memory, or disk usage that may indicate malicious activity.

By analyzing these logs, the system detects security-relevant events that influence the trust score. Specifically, the following scenarios lead to trust value updates:

- If an IDS detects an attempt to exploit a known Hadoop vulnerability, the user's trust score is significantly lowered.
- If authentication logs reveal multiple failed login attempts within a short time window, indicating a possible brute-force attack, the trust value is reduced.
- If system resource logs show anomalies in resource consumption patterns or generate suspicious data traffic, a moderate reduction is applied.
- In cases of severe infractions (e.g., detected malware execution as reported in security logs), the trust score is set to zero, resulting in an immediate account block.

The model also takes into account trust rehabilitation mechanisms. If a previously flagged user demonstrates consistent legitimate behavior over multiple epochs, their trust score can gradually increase, allowing reintegration into the system without requiring full manual intervention.

The trust value can be both increased and decreased by a chosen step value, depending on the users' behavior. In this paper, we use the same step value referenced in [34].

To protect trust-related data from unauthorized modifications, the proposed model can integrate a blockchain-based storage mechanism. Trust values, along with user statuses, can be stored as immutable records in the blockchain. The stored data follows the structure:

$$\{ID_{user}, TV_e, Status_{user}, Time\_of\_storage\} \tag{3}$$

Only the Hadoop administrator has the authority to update this information. By leveraging blockchain, the model prevents unauthorized changes, ensuring transparency and accountability in trust management.

4.3 Security of machine learning models

Attacks against the accuracy, integrity, or security of machine learning (ML) models are referred to as adversarial machine learning (AML) attacks. Malicious actors may use AML assaults to achieve a variety of goals, including obtaining personal data, influencing judgment calls, or jeopardizing the privacy and secrecy of machine learning systems [36]. To mitigate such attacks, we propose the use of multiple datasets and classifiers for intrusion detection. This approach is inspired by the work of Paya et al. [27], who advocate for the use of multiple classifiers instead of relying on a single model. Employing a diverse ensemble of classifiers increases the difficulty for potential attackers to replicate or bypass the Intrusion Detection System (IDS), as it becomes harder to anticipate which classifier will be used to evaluate a specific request. This strategy not only introduces uncertainty for adversaries but also enhances the system's robustness and improves detection accuracy.

We believe that using another new DataSet that containing features of malicious entities can also enhance the performance of our model. In fact, we define different items of new DataSet as described in table 3. Nevertheless, preparing this dataset remains an important issue to be dealt in the future.

Table 3 The proposed DataSet items

N ∘	**Item**	**Description**
1	Age	Age $\geq$ 18 or not.
2	Genre	Male or female.
3	Has a psychological or mental disorder	Yes or No.
4	Has a job	Yes or No.
5	Has a family	Yes or No.
6	Quality of life	Good or Poor.
7	Has self motivation (Vengeance, religion, business interests, etc.)	Yes or No.
8	Has friendship with administrator, analysts or developers	Yes or No.
9	Has friendship with hackers or is a hacker	Yes or No.
10	Social reputation	Honest or not.
11	Is an authenticated user manipulating any system using Hadoop	Yes or No .
12	Successive attacks	Number of successive attacks $\geq$ 0 or not.
13	Date of last detected attack	Is an active attacker ?
14	Dangerous attacks	Number of dangerous attacks $\geq$ 0 or not .
15	Manipulated operating system	Linux, Mac, Android, etc.
16	Coding level	Advanced or good.
17	Used Hardware	Advanced or good.
18	Location	Location information (country, longitude, latitude, etc.)

4.4 EHO-based multi-objective optimization method

ITM-RAH is used to classify participating users based on their behaviors and to make appropriate decisions regarding the execution of service requests while enforcing a set of new policies described in Table 4. We note that the proposed model is also extensible to other policies to protect against additional potential attacks. For example, to counter a Man-in-the-Middle attack, a new policy (Policy 7) can be introduced to verify the user's location and the device used (laptop, smartphone, etc.). Any change would require an email notification to the corresponding user. Upon receiving confirmation, the user would be authorized to proceed with the request.

Table 4 Proposed policies

Policy number	**Purpose**
1	By applying policy number 1, the Name Node can directly perform the request service of the current user.
2	By applying policy number 2, the Name Node can perform the request service on another environment (Honey pot server, sandbox, etc.).
3	By applying policy number 3, the Name Node can perform some limited tasks in isolated environment.
4	By applying policy number 4, the Name Node can deny the request service for short delay.
5	By applying policy number 5, the Name Node can deny the request service for long delay.
6	By applying policy number 6, the Name Node can delete this user from its dataSets.

Algorithm 2 indicate how users are classified according to their behaviors. This classification can be used in risk management process that includes : risk prevention, detection and evaluation.

4.4.1 Description of the Proposed Method

In this paper, we use Elephant Herding Optimization (EHO) [37][38] to provide an optimal solution for making appropriate decisions regarding the execution of user requests. This method aims to maximize user satisfaction while reducing resource consumption costs and minimizing security risks. The choice of EHO is justified by its reputation as one of the most effective evolutionary algorithms, recognized for its versatility in engineering applications, its ability to thoroughly explore the search space, and its successful implementation in various optimization domains [39][40].

The proposed method consists of the following steps:

1. Solution Encoding: Each solution in EHO corresponds to an individual elephant within a clan. In the context of authorization in Hadoop, a solution represents a decision or strategy for managing access permissions based on:
   - Security Risk (SR $\in [0, 1]$): This metric is calculated according to the selected policy using trust and distrust values, error rates (false positives or negatives in

Algorithm 2 Classification of users

```
1: TV_e: Trust value of the current user for this session e;
2: UTV_e: Untrust value of the current user for this session e;
3: H_e: Honesty metrics of the current user for this session e;
4: Begin
5: for each received service request sent by unblocked user do
6: Calculate metrics;
7: Calculate TV_e;
8: Calculate UTV_e;
9: if (H = 0) ) then
10:     User status will be untrusted;
11: else
12:     if (H = 1) then
13:         if (TV_e > UTV_e) then
14:             User status will be trusted;
15:         else
16:             User status will be uncertain;
17:         end if
18:     else
19:         if (TV_e >= UTV_e) then
20:             User status will be uncertain;
21:         else
22:             User status will be untrusted;
23:         end if
24:     end if
25: end if
26: end for
27: End
```

trust value estimation), and impact assessments (potential severity of security breaches caused by executing the requested service, such as financial loss, data confidentiality compromise, or reputational damage).

- Resource Consumption Cost (RCC $\in [0, 1]$): This metric quantifies the execution cost of a service request based on the selected policy, taking into account factors such as energy consumption, RAM usage, and CPU load.
- User Satisfaction (US $\in [0, 1]$): This metric is estimated under the selected policy by considering response time (which can be normalized according to a predefined maximum acceptable time) and the probability of a positive response.

Note: A weight can be assigned to each metric or to individual factors used in their estimation, reflecting their relative importance in decision-making.

2. Fitness Function: EHO evaluates all potential solutions according to a predefined fitness function to determine the most optimal one. This function integrates multiple objectives, including minimizing security risks, maximizing user satisfaction, and reducing resource consumption costs. It serves as a measure of solution quality.

3. EHO Iterations: EHO iteratively updates the population of solutions by guiding each elephant contender toward a "herd center," which is influenced by the best-performing solution so far. At each iteration, the best solution is determined by evaluating each candidate's fitness and selecting the one that optimally balances satisfaction, risk minimization, and cost efficiency. This solution drives the herd towards improved results.

   EHO plays a key role in refining the optimization process through its clan update operation, where solutions evolve by exchanging knowledge across different policies, and through separation, which ensures a broad exploration of possibilities to avoid premature convergence on suboptimal solutions. Together, these mechanisms help identify the most effective access control decisions in Hadoop environments, optimizing security, performance, and user experience.

5 Performance evaluation

This section presents our methodology and outlines our results. All experiments were conducted on an Ubuntu 22.04 LTS machine with an Intel(R) Core(TM) i5-3337U CPU @ 1.80GHz and 6 GB of RAM. Python 3 was used to implement the proposed approach.

5.1 Evaluation of the used ML based IDS

Developing an effective machine learning model requires thorough evaluation to assess its performance and reliability.

5.1.1 Methology

To detect intrusions while mitigating adversarial machine learning (AML) attacks, we use two widely recognized datasets for IDS evaluation: CIC-IDS-2017 and CIC-DDoS-2019 [27]. These datasets were sourced from Kaggle [41] [42].

- CIC-IDS-2017 was generated in a simulated corporate network environment by the Canadian Institute for Cybersecurity (CIC). It includes approximately 80 significant features and simulates the behavior of 25 users. CIC-IDS-2017 can be used alone or in combination with other datasets and is considered a realistic representation of typical network traffic [43].
- CIC-DDoS-2019 contains more than 80 essential features and 13 different types of DDoS attacks. Although the dataset includes simulated network traffic, the benign data is real [44].

For our evaluation, we use the data subsets listed in Tables 5 and 6 after performing data preprocessing. The preprocessing module includes the following steps [45] [46]: data identification, alphanumeric-to-numeric conversion, normalization, and feature selection.

- Data identification: Each dataset is analyzed to detect missing values, duplicate records, and inconsistencies.

Table 5 Subsets of CIC-IDS-2017

Subsets of CIC-IDS-2017	**Number of instances of 'BENIGN' class**	**Number of instances of 'ATTACk' classes**	**Total**
Friday-WorkingHours-Afternoon-DDos (DS1-1)	97718	128027 (DDoS)	225745
Friday-WorkingHours-Afternoon-PortScan (DS1-2)	127537	158930 (PortScan)	286467
Friday-WorkingHours-Morning (DS1-3)	189067	1966 (Bot)	191033
Thursday-WorkingHours-Morning-WebAttacks (DS1-4)	168186	652(XSS) + 1507 (Brute Force) + 21 (Sql Injection)	170366
Tuesday-WorkingHours (DS1-5)	432074	7938(FTP-Patator) + 5897 (SSH-Patator)	445909
Wednesday-workingHours (DS1-6)	63496	5796 (DoS slowloris)+ 623411 (DoS Slowhttptest)	692703

Table 6 Subsets of CIC-DDoS-2019

Subsets of DDoS-2019	**Number of instances of 'BENIGN' class**	**Number of instances of 'ATTACk' classes**	**Total**
LDAP (DS2-1)	4070	841586 (LDAP) + 202919(NetBIOS)	1048575
Portmap (DS2-2)	4734	186960 (Portmap)	191694
Syn (DS2-3)	30660	1017912 (Syn)	1048572
UDP (DS2-4)	2013	1022170(UDP) + 24392 (MSSQL)	1048575

Alphanumeric-to-Numeric Conversion: Categorical attributes are converted into numerical values to ensure compatibility with machine learning algorithms. Label encoding is used, where each unique category is assigned a distinct integer value. This transformation applies to fields such as protocol types and attack labels.

- Data Normalization: To ensure uniform feature scaling, Min-Max normalization can be applied. This method rescales all numerical values to the range [0,1], preventing features with different units or magnitudes from disproportionately influencing the model. The normalization formula is:

$$X_{\text{normalized}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{4}$$

- Feature Selection: To reduce dimensionality and enhance model efficiency, only the most relevant features are retained. The selection process follows the methodology described in [46], where features are ranked based on their contribution to classification performance.

To evaluate the proposed ML-based IDS, we divide each dataset into two sets: training data (70%) and test data (30%).

5.1.2 Metrics

In machine learning evaluation, certain metrics are computed using a confusion matrix. The Confusion Matrix (CM) is a two-dimensional matrix that defines the actual and predicted categories.

- True Positive (TP): Data instances correctly identified as "Attacks" by the applied ML technique.
- False Negative (FN): Data instances incorrectly classified as 'Normal'.
- False Positive (FP): Data instances incorrectly classified as 'Attacks'.
- True Negative (TN): Data instances correctly categorized as 'Normal'.

The following evaluation metrics are commonly used:

- Accuracy: The proportion of correctly classified instances. It is reliable only when the dataset is balanced. Accuracy is computed as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{5}$$

- Precision: The probability that a predicted 'Attack' is indeed an attack. It is computed as:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{6}$$

- Recall (Sensitivity): The proportion of actual attacks correctly classified by the model. It is given by:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{7}$$

- Specificity: The proportion of actual normal instances correctly identified. It is computed as:

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{8}$$

- F1-score: The harmonic mean of precision and recall, providing a balance between the two. It is given by:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{9}$$

- AUC-ROC (Area Under the Receiver Operating Characteristic Curve): AUC-ROC measures the classifier's ability to distinguish between positive and negative classes. The ROC curve plots the true positive rate against the false positive rate at various threshold settings. The AUC represents the probability that a randomly chosen positive instance is ranked higher than a randomly chosen negative instance.
- Execution Time (ET): The total time, in seconds, required to train and test the applied ML technique.

5.1.3 Results

Since our goal with an ML-based IDS is to examine user traffic to identify intrusions, a binary classification has been performed and assessed.

A classification model is built using five classifiers: Decision Tree (DT), Random Forest (RF), K-Nearest Neighbors (KNN), XGBoost, and LightGBM. These classifiers are among the most widely used algorithms in the field of security and intrusion detection due to their strong performance in classifying network traffic and identifying anomalies [27, 47–50]. Their effectiveness in handling high-dimensional data, adaptability to various attack patterns, and robustness against overfitting make them reliable choices for cybersecurity applications. Moreover, these algorithms are frequently considered as baseline models for comparison with newly proposed approaches, helping to evaluate improvements in detection accuracy and efficiency.

- Random Forest (RF): Random Forest is an ensemble learning method that combines multiple decision trees to improve prediction accuracy [51].
- Decision Tree (DT): A fundamental classification algorithm that builds a tree structure based on feature splits to make predictions [52]. It is widely used due to its interpretability and efficiency in handling both categorical and numerical data.
- K-Nearest Neighbors (KNN): This method [53] is flexible and can be used for both regression and classification. It classifies data based on the majority vote of the k nearest neighbors.
- XGBoost (eXtreme Gradient Boosting): An optimized boosting algorithm that enhances decision tree ensembles by reducing bias and variance [54, 55]. It is designed for memory efficiency and speed on large datasets, making it suitable for high-accuracy and scalable classification tasks [50].
- LightGBM: A gradient boosting variant that accelerates training through optimized sampling and efficient data structuring [56]. Its approach enables the processing of large datasets while maintaining high accuracy and low computational complexity [57].

The obtained results, presented in tables 7 to 16, indicate that all the applied ML techniques achieve consistently high and comparable performance in terms of accuracy, recall, precision, F1-score, specificity, and AUC-ROC.

Based on the performance metrics across different datasets (DS1-1 to DS1-5), Random Forest (RF) and XGBoost consistently deliver the highest accuracy, precision, recall, F1-score, and AUC-ROC values. These classifiers exhibit near-perfect performance, especially on DS1-1 and DS1-2, where their AUC-ROC reaches 1.0, indicating excellent classification capability. LightGBM also performs competitively, showing

Table 7 Performance measures of ML classifiers on DS1-1

Classifier & Metrics	Accuracy	Recall	Precision	F1-score	Specificity	AUC-ROC	ET (s)
RF (DS1-1)	99.96	99.99	99.93	99.96	99.99	99.99	141.7778
DT (DS1-1)	99.91	99.89	99.92	99.91	99.89	99.91	103.2179
KNN (DS1-1)	99.67	99.64	99.70	99.67	99.64	99.91	93.3240
XGBoost (DS1-1)	99.96	99.98	99.93	99.96	99.98	100	100.8895
LightGBM (DS1-1)	99.95	99.98	99.93	99.95	99.98	100	98.1135

Table 8 Performance measures of ML classifiers on DS1-2

Classifier & Metrics	Accuracy	Recall	Precision	F1-score	Specificity	AUC-ROC	ET (s)
RF (DS1-2)	99.99	99.99	99.99	99.99	99.99	100	164.5452
DT (DS1-2)	99.99	99.98	99.99	99.99	99.98	99.99	135.7885
KNN (DS1-2)	99.74	99.70	99.78	99.74	99.70	99.96	131.0990
XGBoost (DS1-2)	99.99	99.99	99.99	99.99	99.99	99.99	131.8735
LightGBM (DS1-2)	99.95	99.93	99.97	99.95	99.93	99.92	132.2069

Table 9 Performance measures of ML classifiers on DS1-3

Classifier & Metrics	Accuracy	Recall	Precision	F1-score	Specificity	AUC-ROC	ET (s)
RF (DS1-3)	98.60	97.55	99.69	98.61	97.51	97.43	19.3400
KNN (DS1-3)	98.24	97.24	99.29	98.25	97.19	99.25	7.6303
XGBoost (DS1-3)	98.30	97.04	99.63	98.32	96.97	99.63	12.8367
LightGBM (DS1-3)	98.34	97.07	99.68	98.36	97.00	99.64	10.2859

slightly lower but still outstanding results. Decision Trees (DT) demonstrate strong

Table 10 Performance measures of ML classifiers on DS1-4

Classifier & Metrics	Accuracy	Recall	Precision	F1-score	Specificity	AUC-ROC	ET (s)
RF (DS1-4)	99.20	98.85	99.56	99.20	98.85	99.97	75.4401
DT (DS1-4)	98.91	99.01	98.81	98.91	99.01	98.91	12.2467
KNN (DS1-4)	98.76	98.541	98.98	98.76	98.54	99.87	6.3839
XGBoost (DS1-4)	98.94	98.35	99.55	98.95	98.34	99.95	11.1445
LightGBM (DS1-4)	98.89	98.06	99.76	98.90	98.03	99.95	9.9482

Table 11 Performance measures of ML classifiers on DS1-5

Classifier & Metrics	Accuracy	Recall	Precision	F1-score	Specificity	AUC-ROC	ET (s)
RF (DS1-5)	96.49	94.92	98.24	96.55	94.74	99.54	191.3277
DT (DS1-5)	95.32	95.46	95.18	95.32	95.46	95.32	40.0258
KNN (DS1-5)	96.06	94.92	97.34	96.12	94.78	99.27	21.8064
XGBoost (DS1-5)	96.59	94.70	98.72	96.67	94.47	99.58	42.7382
LightGBM (DS1-5)	96.61	94.67	98.78	96.68	94.43	99.59	29.3645

Table 12 Performance measures of ML classifiers on DS1-6

Classifier & Metrics	Accuracy	Recall	Precision	F1-score	Specificity	AUC-ROC	ET (s)
RF (DS1-6)	90.78	93.81	87.33	90.45	94.23	97.26	738.9104
DT (DS1-6)	86.29	86.61	85.86	86.24	86.71	86.29	212.0339
KNN (DS1-6)	92.49	92.49	75.23	82.97	93.88	92.26	72.1180
XGBoost (DS1-6)	85.98	91.41	79.44	85.01	92.52	94.36	78.2441
LightGBM (DS1-6)	85.26	91.34	77.93	84.10	92.61	94.04	75.6771

Table 13 Performance measures of ML classifiers on DS2-1

Classifier & Metrics	Accuracy	Recall	Precision	F1-score	Specificity	AUC-ROC	ET (s)
RF (DS2-1)	100	100	99.99	100	100	100	459.3689
DT (DS2-1)	99.99	100	99.99	99.99	100	99.77	143.1180
KNN (DS2-2)	99.98	100	99.95	99.98	100	99.99	60.8065
XGBoost (DS2-1)	100	100	99.99	99.99	100	100	53.3871
LightGBM (DS2-1)	99.97	99.97	99.97	99.97	99.97	99.95	49.1065

Table 14 Performance measures of ML classifiers on DS2-2

Classifier & Metrics	Accuracy	Recall	Precision	F1-score	Specificity	AUC-ROC	ET (s)
RF (DS2-2)	99.99	100	99.99	99.99	100	100	73.1566
DT (DS2-2)	99.99	99.99	99.98	99.99	99.99	99.99	21.6255
KNN (DS2-2)	99.86	99.93	99.79	99.86	99.93	99.98	8.1718
XGBoost (DS2-2)	99.99	99.99	99.98	99.99	99.99	100	12.2943
LightGBM (DS2-2)	99.99	100	99.99	99.99	100	100	11.9126

Table 15 Performance measures of ML classifiers on DS2-3

Classifier & Metrics	Accuracy	Recall	Precision	F1-score	Specificity	AUC-ROC	ET (s)
RF (DS2-3)	98.16	97.95	98.39	98.17	97.94	99.72	728.0295
DT (DS2-3)	97.56	97.61	97.51	97.56	97.62	97.56	189.0081
KNN (DS2-3)	98.08	98.06	98.09	98.08	98.06	99.50	86.4101
XGBoost (DS2-3)	98.17	98.02	98.32	98.7	98.01	99.76	94.5674
LightGBM (DS2-3)	98.17	98.84	97.49	98.16	98.86	99.77	92.8878

Table 16 Performance measures of ML classifiers on DS2-4

Classifier & Metrics	Accuracy	Recall	Precision	F1-score	Specificity	AUC-ROC	ET (s)
RF (DS2-4)	100	100	100	100	100	100	336.1381
DT (DS2-4)	100	100	99.99	100	100	100	85.2248
KNN (DS2-4)	99.98	100	99.97	99.98	100	99.99	45.5181
XGBoost (DS2-4)	100	100	100	100	100	100	72.9209
LightGBM (DS2-4)	100	100	100	100	100	100	64.1251

performance, particularly on DS1-2, but remain slightly less effective than RF and XGBoost in terms of overall classification metrics.

Execution time (ET) varies significantly. While XGBoost and LightGBM provide high accuracy, they are generally more computationally efficient than RF, which often requires longer training times. K-Nearest Neighbors (KNN) consistently exhibits the shortest execution time among the evaluated classifiers. This is due to the simplicity of KNN, which does not involve an explicit training phase, as it simply stores the training data and performs distance calculations only during the prediction phase. In contrast, models such as Random Forest (RF) require a more complex training process, involving the construction of multiple decision trees, which explains their higher execution time.

It is important to note that the obtained results can also be influenced by the specific parameters of each applied ML technique. Hyperparameter settings can significantly impact both performance and computational efficiency. Furthermore, the use of the SMOTE technique has effectively addressed the issue of class imbalance, improving the balance of class distributions and leading to more reliable evaluation metrics. Additionally, leveraging parallelization through job management has contributed to reducing memory usage and optimizing execution times, particularly for models such as RF and XGBoost, which involve a high computational load due to extensive operations on large datasets.

5.2 Evaluation of the proposed model

To effectively manage user permissions in a Hadoop environment, we have deployed a three-node Hadoop cluster (version 3.6.6), consisting of a Name Node, a Secondary Name Node, and a Data Node. This cluster is configured to support multiple users while ensuring secure interactions between Hadoop components and users. A crucial aspect of this setup is the integration of Kerberos V5, which guarantees authentication for both Hadoop services and users, enhancing overall system security.

To evaluate our approach, we compared it with a generic authorization model based on RBAC (Role-Based Access Control) and ABAC (Attribute-Based Access Control).

The choice of a generic approach is driven by the need for a more equitable comparison with policy-based authorization mechanisms, ensuring that the evaluation remains independent of any specific framework. This allows for broader compatibility with various authorization systems while maintaining adaptability to future frameworks.

Both our approach and the generic model enforce RBAC and ABAC across multiple Hadoop services, including HDFS, MapReduce, YARN, Hive, HBase, Oozie, and Spark. Users are randomly assigned predefined roles, each with specific permissions governing data access and operations. Additionally, attribute-based policies enable fine-grained and adaptive authorization. All predefined rules are stored in JSON file. Furthermore, both approaches dynamically manage varying request loads per user, evaluating each request individually to ensure compliance with predefined access control policies. Users can submit different types of .jar files to perform read, write, and execute operations while leveraging Hadoop services.

5.2.1 Evaluation Criteria

The proposed model was implemented to classify participating users into three categories based on their behavior and trustworthiness:

- Trusted users: Having consistent, authorized access patterns.
- Uncertain users: Exhibiting unusual or borderline behaviors requiring further evaluation.
- Untrusted users: Detected as potential threats, triggering security measures.

Beyond user classification, the model serves as a proactive security mechanism by mitigating unauthorized access attempts. The effectiveness of the proposed approach was assessed through simulations that compared its performance considering the following metrics:

- Authorized Requests under Policy 1 (AR-P1): Number of authorized service requests obtained by applying Policy 1.
- Authorized Requests under Policy 2 (AR-P2): Number of accepted service requests obtained by applying Policy 2.
- Authorized Requests under Policy 3 (AR-P3): Number of service requests accepted following the selection of Policy 3 by the proposed method.
- Rejected Requests under Policy 4 (RR-P4): Number of service requests denied following the execution of Policy 4.
- Rejected Requests under Policy 5 (RR-P5): Number of service requests rejected due to the application of Policy 5.
- Rejected Requests under Policy 6 (RR-P6): Number of service requests denied following the application of Policy 6.
- Total Authorized Requests (AR): Total number of service requests approved using the proposed model.
- Total Rejected Requests (RR): Overall number of service requests denied when the proposed model was applied.
- Number of users classified as Trusted (TU), Uncertain (UU), or Untrusted (UT).

- CPU Consumption (CC): Assesses the computational resource utilization by measuring the percentage of CPU used during the execution of the model.
- RAM Consumption (RC): Tracks the memory usage of the model, ensuring that resource consumption remains efficient and does not degrade system performance.

The performance of the EHO-based multi-objective optimization was evaluated across iterations using the following key metrics:

- Best Fitness: Measures the quality of the solutions generated by the algorithm, indicating how close each solution is to the optimal one.
- Fitness Error Value: Tracks the difference between the current solution and the theoretical optimal, showing how much the algorithm still needs to improve. A decreasing fitness error over time demonstrates convergence toward the optimal solution.
- Execution Time (estimated in seconds): Measures the computational efficiency of the algorithm, indicating how quickly it computes a solution.

5.3 Results

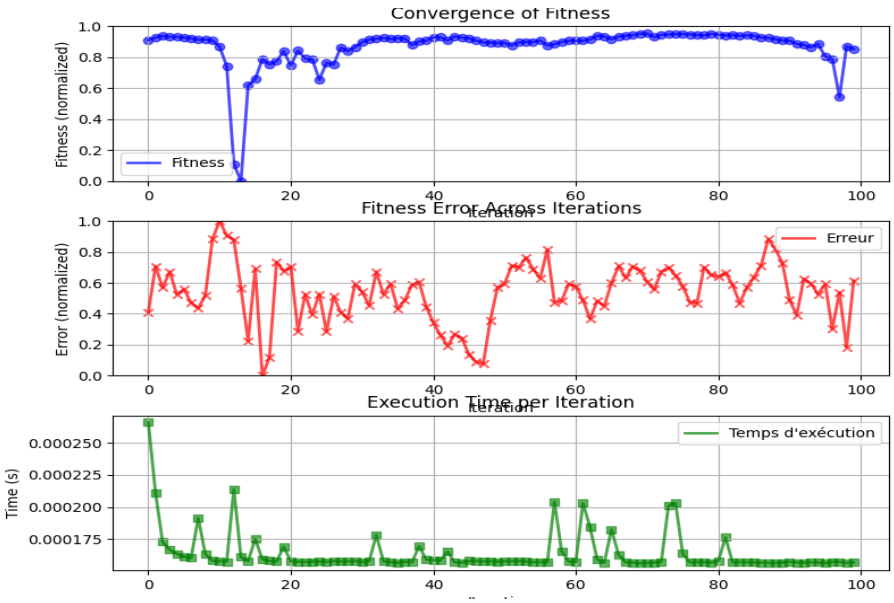

Fig. 2 Performance analysis of EHO-based multi-objective optimization

Figures 2 and 3 show the simulation performance results obtained from various executions of the proposed model while assigning these weights "0.5, 0.3, and 0.2" to security risk, resource consumption cost, and user satisfaction respectively.

The best fitness values obtained, as shown in Figure 2, typically ranged from 0.6X to 0.9X, indicating that the EHO algorithm was, in most cases, able to identify solutions relatively close to the optimal one. Although the fitness values did not reach

the theoretical maximum, the steady improvement observed throughout the iterations suggests that the algorithm effectively narrowed the gap toward the ideal solution.

However, in two cases, the fitness values ranged between 0.001 and 0.1. This outcome is a direct result of the algorithm's stochastic nature and the influence of initial conditions, which can sometimes lead to poor convergence. In these instances, the algorithm may have struggled to escape local optima or failed to explore a sufficiently diverse set of solutions during early iterations.

While the majority of runs demonstrated strong convergence, these specific cases highlight potential areas for improvement. Future work could explore refined initialization strategies or adaptive mechanisms to reduce the likelihood of such occurrences. Nevertheless, the overall trend indicates that the EHO algorithm successfully optimizes solutions in most cases, with occasional exceptions due to inherent stochastic variations.

Figure 3 presents the distribution of accepted and rejected requests after multiple execution cycles, providing valuable insights into the decision-making process of the proposed authorization approach. Beyond merely quantifying accepted and rejected requests, the figure highlights the reasons behind each decision by referencing the corresponding applied policies. The policy selection process incorporates multiple critical factors, including security risks associated with each request, the computational and storage resources required for execution, and the overall impact on user satisfaction. By integrating these parameters, the system ensures that access permissions are governed not only by static policies but also by dynamic, context-aware risk assessments. This enhances adaptability, reducing unauthorized access while optimizing resource allocation within the Hadoop ecosystem.

Additionally, the approach improves the Name Node's ability to classify users into three categories: trusted, untrusted, or uncertain.

The proposed model generally processes fewer accepted service requests in the presence of intrusions and consistently requires higher computational resources, particularly in terms of CPU and RAM usage, compared to the generic authorization model. This increased consumption stems from the integration of a machine learning-based intrusion detection system, which demands additional processing power for real-time anomaly detection, feature extraction, and classification of potentially malicious requests. Unlike traditional authorization mechanisms that rely solely on predefined rules, the proposed model continuously analyzes behavioral patterns and dynamically assesses security risks. While this enhances threat detection capabilities, it also leads to higher computational resource utilization. Table 17 illustrates a representative example of the obtained results, comparing the two approaches.

6 Discussion

This research represents a step forward in enhancing authorization mechanisms and reinforcing security resilience against unauthorized access attempts. By dynamically evaluating user trust and integrating machine learning for anomaly detection,

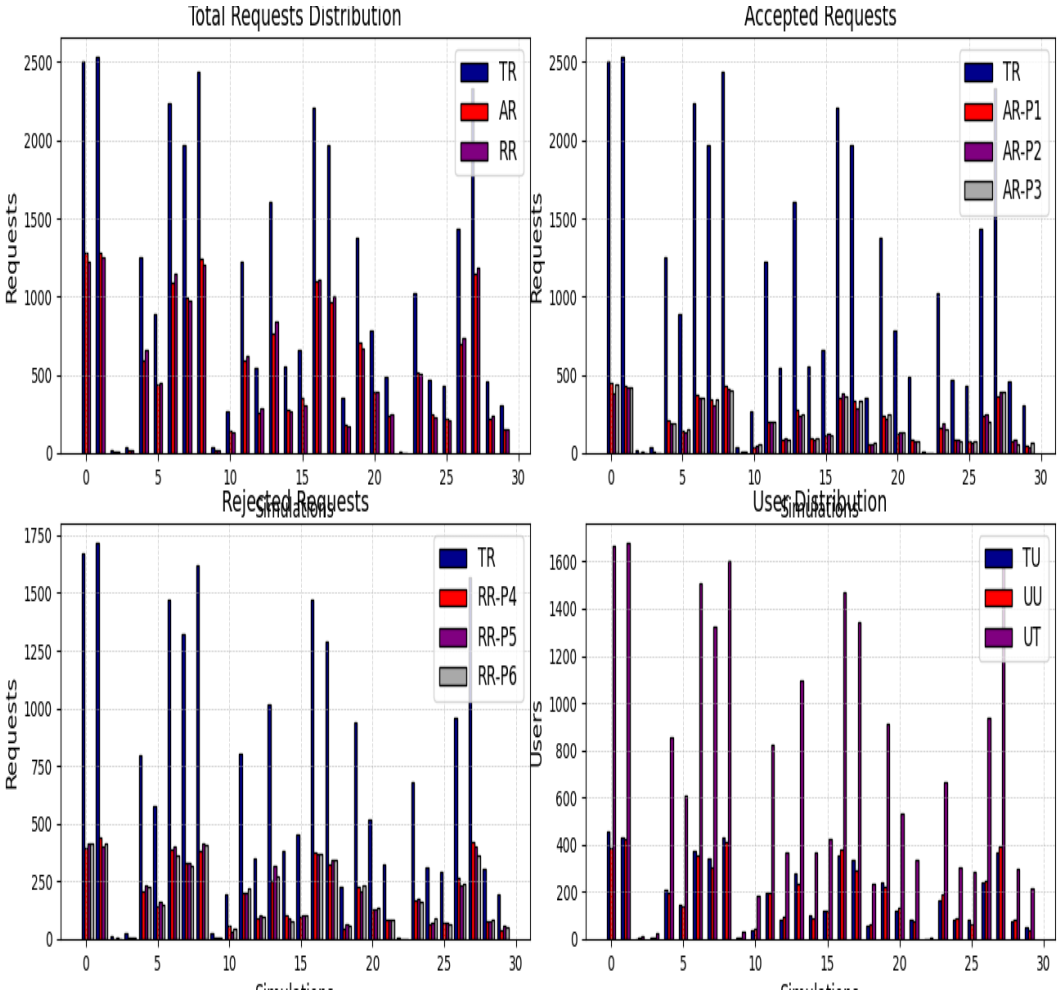

Fig. 3 Performance of the proposed model

our model establishes a more intelligent and context-aware access control system, proactively mitigating potential security threats. Unlike conventional rule-based authorization systems, which rely on static policies, our approach continuously assesses the legitimacy of incoming requests, ensuring that suspicious activities are identified and blocked before they can compromise the integrity of the system. This preventive mechanism significantly reduces the risk of unauthorized data access, privilege escalation attacks, and the exploitation of system vulnerabilities, thereby strengthening the overall security posture.

Our model adopts a binary classification framework that focuses exclusively on the detection of malicious activities, without differentiating attack severity. While this design choice offers a straightforward and computationally efficient intrusion detection

Table 17 Performance measures of compared approaches

Approach & Metrics	**Number of users**	**Number of requests**	**Avg(AR)**	**AVG(RR)**	**Avg(CC)**	**Avg(RC)**	**Avg(ET)**
ITM-AH	177	497	354	143	73.16	65.47	5598
Generic approach	177	497	199	298	42.96	48.28	83.45

mechanism, it lacks the granularity required for more refined security assessments. A more sophisticated approach would involve transitioning to a multi-class classification model capable of assigning confidence scores to detected anomalies. Such an enhancement would prevent minor deviations from disproportionately impacting the overall trust assessment while enabling a more nuanced evaluation of user behavior.

One inherent limitation in our current honesty metric computation lies in its assumption that setting H=0 when all detection models flag an attack necessarily implies malicious intent. This rigid interpretation may lead to false positives, particularly in cases where anomalies are transient or contextually justified. A more adaptive methodology would incorporate additional contextual parameters such as anomaly persistence, historical user behavior, and the operational context of the request to refine classification accuracy and minimize misinterpretations.

Similarly, our mechanism for updating the maximum number of detected attacks (Max) presents certain biases. Since updates occur only when the number of detected attacks (NDA) exceeds its previous threshold, the model may become overly reactive to temporary spikes or false positives. To counteract this, a dynamic thresholding mechanism, informed by historical detection patterns and model confidence scores, would help balance sensitivity and specificity, ensuring more reliable and consistent attack detection.

The effectiveness of our anomaly detection system is inherently tied to the quality, diversity, and representativeness of the training dataset. An imbalance in the dataset such as an under representation of certain attack patterns could lead to high false,positive rates (misclassifying legitimate users as threats) or false negatives (failing to detect actual intrusions). Addressing this challenge requires robust dataset augmentation techniques, adversarial training, and continuous learning mechanisms, ensuring that the model remains adaptive to evolving threat landscapes.

To validate the scalability and practical applicability of our approach, real world deployment in large scale enterprise Hadoop environments is essential. Such deployment would enable a more comprehensive evaluation of security performance, computational overhead, and end-user experience. By monitoring key operational metrics such as authorization latency, system throughput, and resource utilization, the approach could be further refined to achieve an optimal balance between security effectiveness and computational efficiency. While the integration of machine learning inevitably increases resource consumption, this challenge can be mitigated by deploying the system on high performance computing infrastructures, ensuring seamless scalability without compromising security or responsiveness.

7 Conclusion

Ensuring a secure and efficient authorization process is essential for protecting the Hadoop environment. Traditional access control mechanisms, which rely on predefined policies, enforce authorization rules but struggle to detect and mitigate security threats posed by sophisticated attacks and unauthorized behaviors. Their static nature limits adaptability in dynamic and evolving security landscapes.

This paper introduces an advanced trust-based authorization model that enhances Hadoop's security framework by integrating machine learning-driven Intrusion Detection Systems (IDS), conventional security tools such as monitoring systems, and a multi-objective decision-making approach based on Elephant Herding Optimization (EHO). This hybrid strategy strengthens access control by dynamically assessing user trust and responding to potential threats in real time.

The proposed model combines Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC) to establish a flexible and adaptive authorization framework across various Hadoop services. Its performance is evaluated against a generic policy-driven approach using identical access control policies. The machine learning-based IDS mechanism is assessed based on multiple criteria, including accuracy, precision, recall, F1-score, specificity, AUC-ROC, and execution time, while the effectiveness of the EHO-based decision-making mechanism is also analyzed.

It is important to highlight that the obtained results can be significantly influenced by multiple factors, such as dataset characteristics, hyperparameter selection, and the specific properties of each applied ML technique. Additionally, computational power plays a crucial role in system performance, particularly in terms of execution time and resource consumption. High performance hardware with greater processing capabilities, increased RAM, and efficient parallelization mechanisms can considerably reduce execution time and improve scalability. Conversely, on resource-constrained systems, complex models may suffer from prolonged execution times and excessive memory usage, potentially leading to operational bottlenecks or failures. Therefore, any performance comparison must account for these hardware-related dependencies to ensure a fair and comprehensive evaluation.

To further refine the model, the enhancements discussed in Section 6 should be addressed. Additionally, a comparative analysis with other advanced authorization frameworks, such as [33], could provide deeper insights into the strengths and limitations of different security paradigms. Future research will explore the integration of deep learning techniques, including Recurrent Neural Networks (RNNs) for analyzing sequential attack patterns, Autoencoders and Generative Adversarial Networks (GANs) for detecting zero-day attacks, and Reinforcement Learning (RL) for dynamically adjusting authorization policies in response to real-time threats.

Declarations

- Funding: This research received no external funding.
- Conflict of interest/Competing interests : The authors declare that they have no competing interests.
- Ethics approval and consent to participate: Not applicable.

- Consent for publication: All authors have read and agreed to the published version of the manuscript.
- Data availability : The datasets analyzed during this study were downloaded from the Kaggle website.
- Code availability : The authors can provide the code upon request.
- Author contribution: Methodology, and manuscript revision: N.B. and A.M.; software and validation: N.B.

References

[1] Bhathal, G.S., Singh, A.: Hadoop framework vulnerabilities, security issues and attacks. Array **100002**(2), 1–8 (2019)

[2] Koo, J., Kang, G., Kim, Y.: Security and privacy in big data life cycle: A survey and open challenges. Sustainability **10571**, 1–32 (2020)

[3] Perwej, Y.: The hadoop security in big data: A technological viewpoint and analysis. International Journal of Scientific Research in Computer Science and Engineering **7**(3), 1–14 (2019)

[4] Hena, M., Jeyanthi, N.: Authentication framework for kerberos enabled hadoop clusters. International Journal of Engineering and Advanced Technology (IJEAT) **9**(1), 2249–8958 (2019)

[5] Begum, G., Huq, S.Z.H., Kumar, A.P.S.: Sandbox security model of hadoop system. Journal of Big Data, 1–10 (2020)

[6] HKhalid, M., Yousaf, M.M.: A comparative analysis of big data frameworks: An adoption perspective. Appl. Sci. **11**(11033), 1–25 (2021)

[7] Sirisha, N., Kiran, K.V.D.: Authorization of data in hadoop using apache sentry. International Journal of Engineering & Technology **7**(36), 234–236 (2018)

[8] Sutar, P.: Challenges in big data security and mitigating threats by leveraging apache hadoop. International Journal of Scientific & Engineering Research **11**(5), 643–659 (2020)

[9] Neuman, B.C., Ts'o, T.: Kerberos: An authentication service for computer networks. IEEE Communications Magazine **32**, 33–38 (1994)

[10] Tabassum, R., Tyagi, N.: Hadoop identity authentication using public private key concept. International Journal of Engineering Trends and Technology (IJETT) **45**(9), 436–442 (2017)

[11] Rahul, P.K., Kumar, T.G.: A novel authentication framework for hadoop. Advances in Intelligent Systems and Computing **324**, 333–340 (2015)

[12] Khalil, I., Dou, Z., Khreishah, A.: Tpm-based authentication mechanism for apache hadoop. In 10^{th} International Conference on Security and Privacy in Communication Networks (SecureComm) (2014)

[13] Hu, D., Chen, D., Zhang, Y., Pei, S.: Tpm-based authentication mechanism for apache hadoop. International Journal of Security and Its Applications **9**(11), 429–438 (2015)

[14] Jeong, Y., Shin, S., Han, K.: High-dimentional data authentication protocol based on hash chain for hadoop systems. Cluster Comput **19**, 47–84 (2016)

[15] JChattaraj, D., Sarma, M., Das, A., Kumar, N., Rodrigues, J., Park, Y.: Heap: An efficient and fault-tolerant authentication and key exchange protocol for hadoop-assisted big data platform. IEEE Access (2019)

[16] Shen, P., Ding, X., Ren, W.: Research on kerberos technology based on hadoop cluster security. In 2^{nd} International Conference on Advances in Energy, Environment and Chemical Science (AEECS), 429–438 (2018)

[17] Kankal, M., Patil, P.: An adaptive authentication based on blockchain for big data hadoop framework. International Journal of Engineering and Techniques **5**(1), 120–138 (2019)

[18] Balaraju, J., Rao, P.P.: Investigation and finding a dna cryptography layer for securing data in hadoop cluster. Int. J. Advance Soft Compu. Appl **12**(3) (2020)

[19] Hena, M., Jeyanthi, N.: A three-tier authentication scheme for kerberized hadoop environment. Cybernetics and information technologies **21**(4), 119–136 (2021)

[20] Wang, C.A.: Project rhino goal: at-rest encryption for apache hadoop. Available: https://blog.cloudera.com/blog/2014/06/project-rhino-goal-at-rest-encryption/

[21] Haggag, M., antawy, M.M., El-Soudani, T.M.M.S.: Token-based authentication for hadoop platform. Ain Shams Engineering Journal **14**(101921), 1–15 (2023)

[22] Gupta, M.S., Patwa, F., Benson, J., Sandhu, R.: Multi-layer authorization framework for a representative hadoop ecosystem deployment. In Proceedings of the 22^{nd} ACM Symposium on Access Control Models and Technologies (SACMAT), 183–190 (2017)

[23] Wen, Y., Wang, C.: Data privacy mechanisms development and performance evaluation for personal and ubiquitous blockchain-based storage. The journal of Supercomputing **17**(79), 1–35 (2003)

[24] Cherbal, S., Zier, A., Hebal, S., Louail, L., Annane, B.: Security in internet of things: a review on approaches based on block chain, machine learning, cryptography, and quantum computing. The journal of Supercomputing, 1–79

(2003)

[25] Chikouche, N., Cayrel, P., Mboup, E.B.B.: A privacy-preserving code-based authentication protocol for internet of things. The journal of Supercomputing **75**, 8231–8261 (2019)

[26] Rajeh, W.: Hadoop distributed file system security challenges and examination of unauthorized access issue. Journal of Information Security **13**, 23–42 (2022)

[27] Paya, A., Arroni, S., GarciaDiaz, V., Gomez, A.: A survey of network-based intrusion detection data sets. Computers & Security **136**(103546), 147–167 (2024)

[28] Grover, C., Aulakh, M.K.: Big data authentication and authorization in hdp (hadoop distributed platform) using kerberos and ranger. International journal of advanced research in science and engireering **6**(6), 183–190 (2017)

[29] Idar, H.A., Belhadaoui, H., Filali, R., Malassa, O.: A role-attribute based access control model for dynamic access control in hadoop ecosystem. IAENG International Journal of Computer Science **50**(1) (2023)

[30] Chen, C., Elsayed, M.A., Zulkernine, M.: Unveiling machine learning strategies and considerations in intrusion detection systems: a comprehensive survey. In IEEE 6^{th} International Conference on Dependability in Sensor, Cloud and Big Data Systems and Application (DependSys) (2020)

[31] Yang, M.: Tdacs: an abac and trust-based dynamic access control scheme in hadoop. arXiv:2011.07895 (2020)

[32] Raju, Y.R., Donavalli, H.: Authorization in hadoop using kerberos authentication system and transport layer security. International Journal on Emerging Technologies **11**(1), 403–408 (2020)

[33] Tall, A.M., Zou, C.C.: A framework for attribute-based access control in processing big data with multiple sensitivities. International Journal on Emerging Technologies **13**(2), 403–408 (2023)

[34] Battat, N., Makhoul, A., H., K.: Towards an efficient monitoring in multi-hop mobile ad hoc networks. Ad Hoc Sens. Wirel. Networks **48**(2), 93–120 (2020)

[35] Azad, S., Mahmud, M., Zamli, K.Z., Kaiser, M.S., Jahan, S., Razzaque, M.A.: ibust: An intelligent behavioural trust model for securing industrial cyber-physical systems. Expert Systems With Applications **238**(121676), 1–16 (2024)

[36] Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B.I., Tygar, J.D.: Adversarial machine learning. In: Proceedings of the 4^{th} ACM Workshop on Security and Artificial Intelligence, 43–58 (2011)

[37] Wang, G., Deb, S., Coelho, L.D.S.: Elephant herding optimization. Symposium

on computational and business intelligence (ISCBI), 1–5 (2015)

[38] Wang, G.G., Deb, S., Gao, X.Z., Coelho, L.D.S.: A new metaheuristic optimisation algorithm motivated by elephant herding behaviour. International Journal of Bio-Inspired Computation **8**(6), 394–409 (2016)

[39] Li, J., Lei, H., Alavi, A.H., Wang, G.: Elephant herding optimization: Variants, hybrids, and applications. Mathematics **8**(1415), 1–25 (2020)

[40] Krishna, D., Alam, M.M., Dhanatwal, M.: Swarm-intelligent elephant herding optimized support vector machine for predicting heart disease. Multidisciplinary Science Journal **5**, 1–6 (2023)

[41] Dataset, C.-I.-.: Available online: https://www.kaggle.com/datasets/chethuhn/network-intrusion-dataset

[42] Dataset, C.-D.-.: Available online: https://www.kaggle.com/datasets/dhoogla/cicddos2019

[43] Ring, M., Wunderlich, S., Scheuring, D., Landes, D., Hotho, A.: Survey of network-based intrusion detection data sets. Comput. Secur. **86**, 147–167 (2019)

[44] Shroff, J., Walambe, R., Singh, S.K., Kotecha, K.: Enhanced security against volumetric ddos attacks using adversarial machine learning. Comput. (2022)

[45] Halim, Z., Yousaf, M.N., Waqas, M., et, a.: An effective genetic algorithm-based feature selection method for intrusion detection systems. Computers & Security (2021)

[46] González, A.P., Riego, G.V. A. S. nd Díaz, Gómez, A.: Apollon: A robust defence system against adversarial machine learning attacks in intrusion detection systems. MComputers Security **136**(103546), 1–13 (2023)

[47] Hossain, A., Islam, S.: Ensuring network security with a robust intrusion detection system using ensemble-based machine learning. Array **19**(100306), 1–14 (2023)

[48] Nakamura, K.: A practical approach for discriminating tectonic settings of basaltic rocks using machine learning. Applied Computing and Geosciences **19**(100132), 1–10 (2023)

[49] Altulaihan, E., Almaiah, M.A., Aljughaiman, A.: Anomaly detection ids for detecting dos attacks in iot networks based on machine learning algorithms. Sensors **24**(713) (2024)

[50] Nassreddine, G., Nassereddine, M., Al-Khatib, O.: Ensemble learning for network intrusion detection based on correlation and embedded feature selection techniques. Computers **14**(82), 1–23 (2025)

[51] Breiman, L.: Random forests. Mach. Learn **45**, 5–32 (2001)

[52] Quinlan, J.R.: Induction of decision trees. Machine Learning **1**(1), 81–106 (1986)

[53] Sun, J., Du, W., Shi, N.: A survey of knn algorithm. Information Engineering and Applied Computing, 1–10 (2018)

[54] Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. KDD '16: Proceedings of the 22^{nd} ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794 (2016)

[55] R., D.S.S.N.A.A.A.: Effective intrusion detection system using xgboost. Information **9**(7) (2018)

[56] Ke, G.L., Meng, Q., Finley, T., Wang, T.F., Cheng, W., Ma, W.D., et, a.: Lightgbm: A highly efficient gradient boosting decision tree. Adv. Neural Inf. Process. Syst. **30** (2017)

[57] GZhang, H., Ge, L., Zhang, G., Fan, J., Li, D., Xu, C.: A two-stage intrusion detection method based on light gradient boosting machine and autoencoder. Mathematical Biosciences and Engineering **20**, 6966–6992 (2023)