SSRG International Journal of Electronics and Communication Engineering
ISSN: 2348-8549/ https://doi.org/10.14445/23488549/IJECE-V12I7P126

Volume 12 Issue 7, 327-345, July 2025
© 2025 Seventh Sense Research Group®

Original Article

ForenSecure-AILSO: A Blockchain-Fuzzy Intelligence Framework for Resilient Cloud Forensics and Secure Log Management in Distributed Environments

Ragu Gnanaprakasam[1], Ramamoorthy Sriramulu[2], Poorvadevi Ramamoorthy[3]

[1,2]*Department of Computing Technologies, SRM Institute of Science and Technology, Kattankulathur, Chengalpattu, Tamil Nadu, India.*
[3]*Sri Chandrasekharendra Saraswathi ViswaMaha Vidyalaya University, Kanchipuram, Tamil Nadu, India.*

[1]*Corresponding Author : rg8010@srmist.edu.in*

Received: 14 May 2025 Revised: 15 June 2025 Accepted: 16 July 2025 Published: 31 July 2025

Abstract - *Cloud computing continues to redefine the way data is stored, processed, and accessed globally. However, as organizations use more cloud services, digital forensic investigations get more complex in these networks. Typical forensic methods still struggle to handle secure logging, key exchange, and current threat sharing in systems that change their operation with growth and shrinkage. This paper introduces ForenSecure-AILSO, a unique cloud forensics framework that uses Fuzzy Logic, CALSO, and mechanisms based on smart contracts to secure the collection of evidence. It tackles important problems in multi-cloud computing, such as log storage verification, mystery identity generation, real-time monitoring to spot risks, and matching forensic evidence. In order to keep the information secure and traceable, the log entries are cleaned, enriched, and hashed with SHA-3 on the private consortium blockchain. With a fuzzy identity engine, automatic tokens are created that are difficult to trace, while CALSO keeps session keys strong by adjusting them regularly based on how difficult they are to break. CALSO-TPR directs an anomaly detection engine to quickly spot and alert about any suspicious activity. An evaluation of six key areas, using 10 benchmark models, reveals that ForenSecure-AILSO performs better than other options, scoring an F1-score for log cleaning of 94.0%, accuracy in identifying threats of 96.5%, and a key entropy of 289 bits. The framework managed to detect 99.2% of attempts to change digital tokens and was precise with 93.5% of its subgraph matches, proving its strength and adherence to laws.*

Keywords - *Cloud forensics, Blockchain, Fuzzy Logic, CALSO, Session key optimization, Forensic integrity, Anomaly detection, Evidence correlation, Threat Prediction, Smart Contracts.*

1. Introduction

In the last decade, the way organizations use technology was revolutionized by cloud computing, as it made it possible for them to handle their data using virtual and scalable platforms. As a result, using the cloud, businesses, governments, and people are able to reduce their hardware installation expenses and work at greater speed, helping departments in different parts of the world collaborate more [1, 2]. Due to various cloud deployment and service models, businesses in every industry have embraced cloud technology. A dramatic increase is seen in data amounts, speeds, and types, in addition to a highly linked and split computing system. However, this move towards using the cloud can lead to major security and forensic difficulties. In many regions, on different platforms, and various administrative areas, cloud-based services cause new challenges in data management, ensuring user privacy, handling evidence, and meeting different sets of regulations [3, 4]. Cloud service providers have effective platforms for storing and running computations, though they tend to lack the required tools and assurances needed for any legal use of evidence. Therefore, models that require working directly with data or systems are not useful in the digital environments [1, 5].

Having many users on a single cloud system can lead to issues in cloud forensics. In multi-tenancy, the same hardware is shared by different clients, who still have their resources divided. While it reduces both expenses and the need for extra resources, it becomes more difficult to investigate crimes [6] [7, 8]. Evidence should be pulled from devices or the network without infringing on the privacy of co-resident tenants. Authorities cannot take a server without first separating it from the data of its various users. For this to happen, cloud providers should include more detailed and tenant-specific forensic technologies, which are mostly missing from regular CSP plans. It is also challenging to manage the elasticity of resources, such as Virtual Machines (VMs), containers, and storage volumes. It only takes moments to build, use, and dispose of VMs. Since these events only last a moment, they do not leave a lot of evidence, and if they are not caught when they occur, information can be lost forever. Moreover, when a virtual IP address is reused, it makes it very difficult to maintain the same identity or actions from one session to the next [9-11]. For cloud data to be used in forensic analysis, the state should be unchanged, there may be time gaps between entries, and essential information should be added, which is generally not covered by API features. Figure 1 shows the security and compliance benefits in cloud forensics and secure log management.

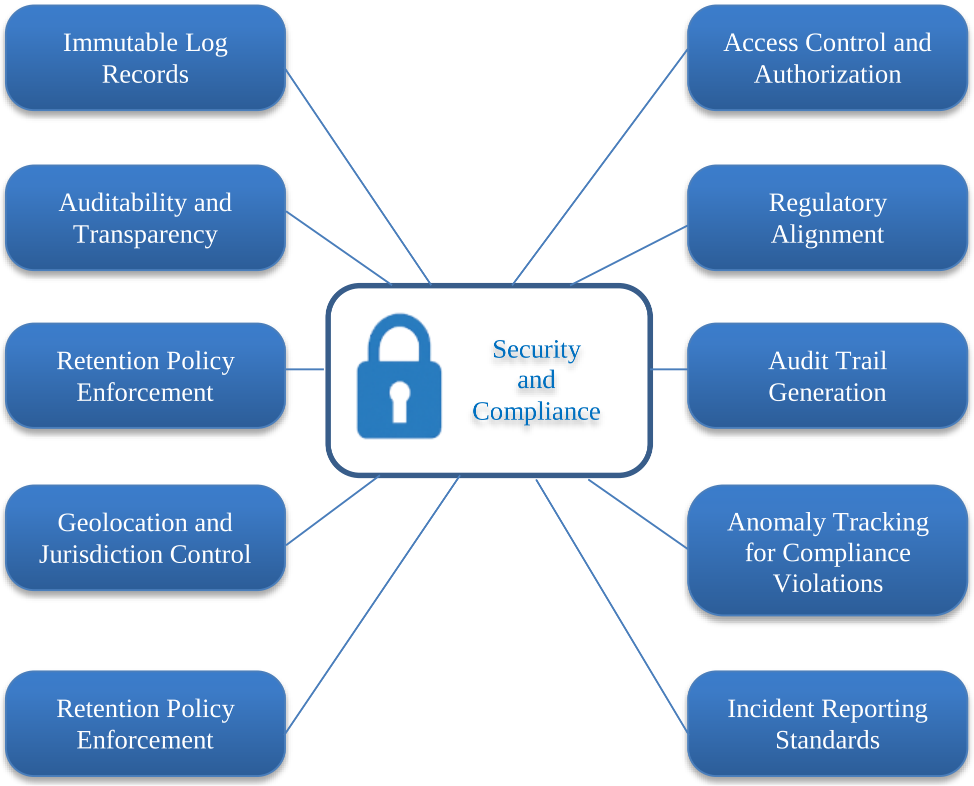
Immutable Log Records
Access Control and Authorization
Auditability and Transparency
Regulatory Alignment
Retention Policy Enforcement
Security and Compliance
Audit Trail Generation
Geolocation and Jurisdiction Control
Anomaly Tracking for Compliance Violations
Retention Policy Enforcement
Incident Reporting Standards

Fig. 1 Security and compliance benefits in cloud forensics and secure log management

The fundamental issue is that no unified forensic framework currently enables secure, anonymizes, cross-correlates and verifies digital evidence in real-time across volatile and distributed frameworks [12]. Conventional digital forensics approaches that target static machines do not work in the context of dynamically orchestrated systems in which Virtual Machines (VM), containers, and user identities are ephemeral [13]. Also, current methods frequently require the use of CSPs to store and maintain the evidence itself, posing a question of trust, tampering and admissibility of the hard copy records. These underlying flaws have not been satisfactorily checked until now by earlier studies, albeit since the development of cryptography, blockchain, and the AI technique [14, 15].

Moreover, multi-tenancy offers potential co-residency-based inference attacks, and Advanced Persistent Threats (APTs) use encrypted communication, anonymization overlays, and dynamic evasion techniques to hamper forensic visibility [16]. The lack of privacy-enabling authentication and authorization protocols and secure session key negotiation mechanisms makes it all the harder to attribute a set of actions to actors in a shared infrastructure [17]. This reveals an important research gap: a decentralized, intelligent, and tamper-resistant forensic model offering traceability, anonymity, and security, independent of centralized trust anchors [18, 19].

This paper proposes ForenSecure-AILSO, which is a robust forensic framework that combines the synergy of fuzzy logic, blockchain-enabled logging, and Crossover-based Artificial Lizard Search Optimization (CALSO). The multi-source logs will be independently sanitized in the proposed system, the anonymous user identities will be created using fuzzy extractors, and blockchains will be used to ensure security properties of the system by keeping evidence or session key entropy in a blockchain encrypted with Merkle Tree structures. CALSO is also extended to threat prediction (CALSO-TPR), which allows scoring and real-time anomaly and forensic correlation. Unlike prior systems that were designed around addressing only individual aspects of privacy (key generation), privacy following identity exposure (identity masking), or all-purpose logging, ForenSecure-AILSO provides an end-to-end solution with privacy, traceability, and predictive analytics integrated in a single architecture. The research not only introduces a technological step forward in the area of secure digital forensics but also establishes the cornerstone of future decentralized auditing systems in the multi-cloud world, where privacy, accuracy, and the ability to go to court need to be met simultaneously.

1.1. Main Contribution of the Work

- Novel Hybrid Forensic Framework (ForenSecure-AILSO): In this work, a hybrid cloud forensics model is designed by using fuzzy logic, CALSO, and blockchain storage to ensure that forensic analysis in distributed clouds is trustworthy, unchanging, and secret.
- Fuzzy-Based Anonymous Identity Generation: Using features like MAC address and frequency of access, a model is built that can generate anonymous profiles that are very resistant to reidentification and still provide high accuracy in authentication.
- CALSO-Optimized Secure Session Key Management: The new design uses CALSO to intelligently develop session keys that are highly secure and difficult to break, improving on previous negotiation schemes and increasing key diversity.
- Blockchain-Backed Forensic Logging and Chain-of-Custody Enforcement: Secure log entries are verified with SHA-3, then written in Merkle Trees on a private consortium blockchain and controlled by smart contract permission roles.

These contributions collectively offer a powerful, intelligent, and legally compliant forensic framework suitable for modern multi-cloud environments.

The remainder of this paper is organized as follows: Section 2 presents related studies on cloud forensics, blockchain-based evidence preservation, fuzzy logic in identity anonymization, and bio-inspired optimization techniques for secure key generation. Section 3 outlines the proposed ForenSecure-AILSO methodology, including multi-platform log sanitization, fuzzy-based identity extraction, CALSO-driven session key optimization, blockchain-backed storage, and predictive threat detection. Section 4 discusses the experimental results, comparing ForenSecure-AILSO against ten benchmark models across six forensic dimensions. Finally, Section 5 concludes the study with key insights and proposes future directions involving federated forensic agents, multi-modal inputs, and adaptive learning mechanisms for real-time forensic intelligence.

2. Related Works

Migration of conventional digital forensics to cloud-based forensic systems has brought numerous issues and opportunities to fight cybercrime in the contemporary world. With cloud computing platforms becoming the standard through which organizations need to conduct operations, complexity of forensic evidence gathering, securing, and validation has become very high. Various studies have tried to fill such tech gaps in the forensics area with the help of blockchain technology, smart contracts, encryption algorithms, and so on.

The forensic framework that combines Software Defined Networking (SDN) and blockchain was proposed in recognition of tamper-free evidence sharing in the cloud infrastructure space. The framework enabled the security of forensic workflows to ensure assurances through the use of SRVA to detect misuse and the application of Sadaun Elliptic Curve Cryptography (SECC) and the SHA-3 Merkle Hash Trees to secure and prove metadata hierarchies. This approach validated the use of blockchain to enhance the decentralization of evidence management and build trust in forensic systems. The system, however, did not support the development of dynamic session monitoring, adaptive key generation or managing of anonymized identities features, which are important in real-time multi-user scenarios.

Immutability of the Forensic processes was also highlighted in the same article as an objective to attain further in the vision. The chain-of-custody was maintained, and the framework enhanced judicial admissibility by recording each transaction permanently on the blockchain. The use of smart contracts smoothed the way to the seamless integration of those with forensic APIs, where it is possible to automate the acquisition and validation processes. This system was found to be resource-efficient and applicable for implementation, even in resource-scarce environments, like IoT and healthcare systems [20]. Building on this, the case study involves a blockchain-based forensic architecture that examined one that was adapted to industrial safety applications that included IIoT (Industrial Internet of Things) systems [21]. They had an intelligent contract-based architecture and access control, which was token-based, to handle secure forensic transactions. The new batch consensus mechanism has been proposed to maximise the performance of data collection and guarantee fault tolerance in high-scale evidence gathering. The simulation test depicted this consensus method as much faster and consistent than the DPOS algorithm. Irrespective of such benefits, the system was IIoT-centric and lacked a universal forensic model that applies to cloud native systems, as well as user identity obfuscation.

In order to have a broader perspective of the challenges related to the security of forensic data, a detailed study was held to compare the common and cloud-specific forensic procedures [22]. The researchers have found that the large spread of cloud technologies has brought about new attack surfaces, data privacy risks, and data volatility. The study was conducted based on the analysis of some significant cases of security breaches and the related expenses to conclude that classic forensic tools cannot meet cloud dynamics adequately.

Moreover, it predicted the worldwide cloud forensics market to develop at a CAGR of 16.53% between 2023 and 2031 and reach USD 36.9 billion at decade-end. This estimate points to the dire need for resilient, flexible, and smart forensic designs. The IoT field of development also could not be left behind with an advancement in the PBCIS-IoTF system, which provided a framework that is used to counter the challenges of evidence provenance and provenance in a heterogeneous IoT scenario [23]. This system was able to store digital evidence on the distributed ledger in the form of SHA-256 hash values, and the nodes used were Hyperledger blockchain and Raspberry Pi nodes. It allowed investigators and the court to authenticate the data retrieved by smart devices with great authenticity and integrity. Nevertheless, this framework had limitations in that it was based on the fixed identity of devices and could not perceive any intelligence in responding to threats or present any session key negotiation.

Although significant progress has been jointly achieved on the above studies, it is noted that none of them have extended their work to the forensic lifecycle in detail, especially regarding privacy-preserving identity generation, real-time threat prediction, and adaptive session key management. They also did not have a single platform, which would efficiently combine the capabilities of sanitisation of evidence, correlation, and chain-of-custody management in a dynamic, multi-cloud environment.

3. Methodology

The proposed ForenSecure-AILSO framework contains a series of steps that are built for effective and smart cloud forensics. Logs are first fetched from various cloud sources and then filtered and assigned a priority using entropy and priority scoring. They are next enhanced with metadata and hashed using SHA-3 to build Merkle Trees for safe and secure blockchain-based storage. With fuzzy logic, anonymous user tokens are made from the attributes and patterns of a mobile device. CALSO is used to optimize session keys, which raises the entropy and makes it harder for enemies to break the keys. Moreover, CALSO watches for security threats as they occur, and traces any signs of wrongdoing within a timeline on forensic graphs. Figure 2 shows the architecture diagram.

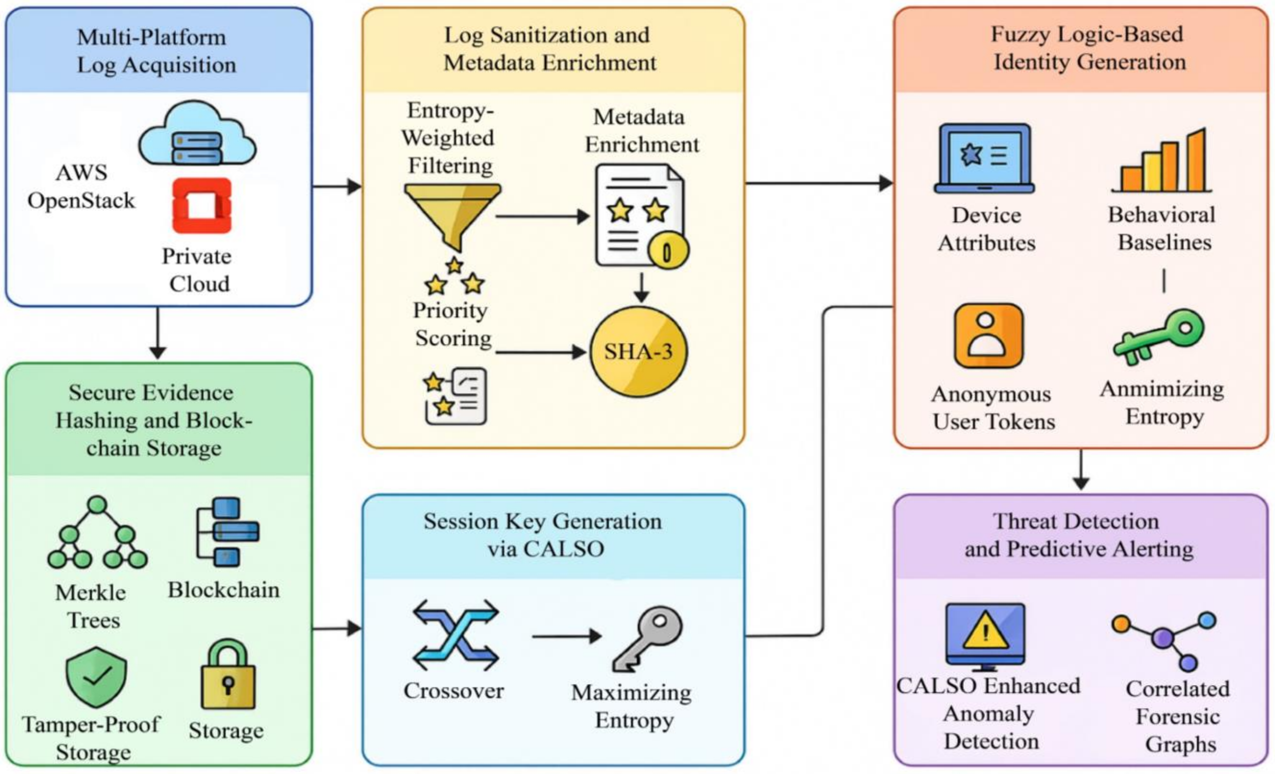
Multi-Platform Log Acquisition
AWS
OpenStack
Private Cloud
Log Sanitization and Metadata Enrichment
Entropy-Weighted Filtering
Metadata Enrichment
Priority Scoring
SHA-3
Fuzzy Logic-Based Identity Generation
Device Attributes
Behavioral Baselines
Anonymous User Tokens
Anmimizing Entropy
Secure Evidence Hashing and Block-chain Storage
Merkle Trees
Blockchain
Tamper-Proof Storage
Storage
Session Key Generation via CALSO
Crossover
Maximizing Entropy
Threat Detection and Predictive Alerting
CALSO Enhanced Anomaly Detection
Correlated Forensic Graphs

Fig. 2 Overview architecture of the proposed ForenSecure-AILSO framework

3.1. Multi-Platform Log Acquisition (MLLAS – Multi-Layer Log Acquisition System)

The first step in the ForenSecure-AILSO methodology is setting up MLLAS, a flexible and strong log acquisition tool. It is set up to run smoothly on clouds with different software, highlighting Infrastructure-as-a-Service (IaaS) suppliers such as AWS, OpenStack, and mixes of private and public resources. MLLAS is designed to allow a reliable data ingestion process for gathering crime-related evidence from multiple and decentralized cloud environments. For this purpose, the system uses a more detailed version of the Cloud Auditing Data Federation (CADF) schema, which is a standard suggested by DMTF that helps build clear and compatible audit trails across cloud services. Standardized acquisition by MLLAS includes capturing many different log types. For example, system-level logs such as syslog and kernel events capture key system activity in virtual machines or containers, and API access logs (such as AWS CloudTrail) register any calls made by programs to the control plane. Also, the framework regularly keeps a record of all actions taken by users through dashboards, SSH sessions, and logging in using user credentials. The use of multifaceted logs allows investigators to see every detail of all actions taking place in the cloud infrastructure.

One key feature of MLLAS is that it helps coordinate P2P cloud storage networks like BitTorrent Sync. Data from network monitors on edge devices and user terminals is collected and looked at by forensic investigators to see how files are transferred and find instances of unauthorized data sharing. This feature is important when criminals try to avoid centralized monitoring or make use of network anonymity by using P2P overlays.

To make sure the logs are accurate and valid, MLLAS requires all systems to keep the same time using NTP. Allowing every event record to reference the same global clock guarantees that data is unaffected by out-of-order logging, time-skew attacks, and artificial timestamp changes. It is critically important to have clocks work in sync so that timelines for forensic investigations are correct and so that chain-of-custody regulations are observed. The MLLAS implementation gives a good base for collecting logs securely across platforms, which is needed for further forensic and security work using the ForenSecure-AILSO framework.

3.2. Log Sanitization and Metadata Enrichment

Sanitizing logs and enhancing metadata is crucial to turn a large amount of unmanageable log data into useful and organized forensic evidence. In the beginning, the email gets filtered by checking for patterns and giving scores depending on its importance. System messages, such as constant heartbeats and status checks, are identified and dismissed by using regular expressions and rule-based classifiers. Each log entry is also given a score for priority, considering things such as how serious the event is, how much trust the source IP has, how frequently it occurs, and whether it was previously unusual. As a result, investigators can pay close attention to logs that matter most, making the next steps much faster and more efficient use of the computer's resources. Let $L_r = \{l_1, l_2, \ldots, l_n\}$ be the set of raw log entries, $S(L_r)$ be the sanitization function that removes noise and assigns priority, and $\mathcal{E}(l)$ be the enrichment function applied to each sanitized log $l \in S(L_r)$. Then, the final set of enriched logs is:

$$L_e = \{\mathcal{E}(l) | l \in S(L_r)\} \tag{1}$$

Once irrelevant data is removed, the remaining log entries are given extra details that cannot usually be seen in regular log formats. One of the first layers of enrichment works by looking up the IP addresses of users or services, then finding out where they point to in real life with the help of online maps. This also helps add an important location factor to how people access data, which is key for spotting unusual activity that crosses borders or any data rules that might be broken. Each log entry l_i is assigned a priority score $p(l_i) \in [0,1]$ based on a weighted sum of relevance features:

$$p(l_i) = w_1 f_{severity}(l_i) + w_2 f_{frequency}(l_i) + w_3 f_{trust}(l_i) + w_4 f_{anomaly}(l_i) \tag{2}$$

Where $f_{severity}$ denotes the map's event type to severity level, $f_{frequency}$ captures event repetition, f_{trust} assesses IP or user reputation, $f_{anomaly}$ flags known derivations from historical behavior, $\sum w_j = 1, for\ j = 1 \ldots 4$. A threshold $\tau \in [0,1]$ is chosen (e.g., $\tau = 0.5$) to retain logs:

$$S(L_r) = \{l_i \in L_r | p(l_i) \geq \tau\} \tag{3}$$

At the same time, logs are checked against known identifiers of both hosts, the browsers used, and the metadata of virtual machines. This way, the system can determine which device or virtual resource performed certain actions, preventing unclear identification of the source. Additionally, setting up devices with digital signatures or cryptographic hashes makes it easier to identify and monitor their setup. Each sanitized log l is enriched as:

$$\mathcal{E}(l) = \big(l, \gamma(l), \delta(l), \beta(l), v(l)\big) \tag{4}$$

Where $\gamma(l)$ is the geolocation vector (e.g., [country, region, city]), $\delta(l)$ is the device signature hash (e.g., SHA-256 of hardware ID), $\beta(l)$ is the behavioral profile vector (e.g., login frequency, commands), and $v(l)$ is the cloud node metadata (e.g., {region ID, instance ID}).

Additionally, learning from how people use the system helps, so what they usually do is captured in the logs by saving their regular access patterns, time on the platform, commands used, and resources used. It helps notice the difference between authorized users and hackers using the same processes. It is also used to detect anomalies in later parts of the analyzis when a sudden deviation is seen as suspicious. The enriched log set L_e becomes:

$$L_e = \big\{\big(l, \gamma(l), \delta(l), \beta(l), v(l)\big) | p(l) \geq \tau, l \in L_r\big\} \tag{5}$$

For each log entry, an attached cloud node metadata tells us the details about the server, location, environment, and service level involved. Because of this, investigators are able to connect events in different parts of the infrastructure, understand spreading actions, and chart the ways threats travel among different users in a cloud system.

3.3. Fuzzy Logic-Based Identity Generation (FISSE – Fuzzy Identity and Session Security Engine)

Using the Fuzzy Identity and Session Security Engine (FISSE), the framework is designed to create safe and private user identities for each transaction. The module focuses on helping to secure cloud forensics by verifying every access, anonymizing the user's credentials, and avoiding any leaks of their details. Most traditional ways of authenticating users depend on matching fixed credentials, putting them at risk of being copied, stolen, or used by others. The system relies on fuzzy logic, making user identification dynamic by taking into account the user's naturally changing and occasion-dependent features, which helps strengthen resistance to advanced persistent threats. Let the user/session attributes be represented as a feature vector:

$$x = [x_1, x_2, x_3] = [DeviceID, MAC, AccessFrequency] \quad (6)$$

This vector $x \in R^3$ contains noisy but unique session-based identity features. A fuzzy extractor $F(\cdot)$ maps a noisy feature vector x to a stable, anonymous identity ID_{anon} and a helper string h for future reconstruction:

$$(ID_{anon}, h) = F(x) \quad (7)$$

Where $ID_{anon} \in \{0,1\}^n$ is the anonymous, stable identity representation, h is public and used to regenerate ID_{anon} later from a similar $x' \approx x$. The identity generation process starts by getting a list of device features and behavior every time someone uses the app. Key among these are things like the device identifier (such as a serial number or unique ID), the MAC address, and how often the devices perform certain actions or what kind of timing you see them using. These values often change because of things like different devices, moving computer memories around, or making changes to the network. To help deal with this uncertainty, FISSE uses fuzzy extractors, which are special tools that can create the same output even from bits that are close to, but not exactly the same. In essence, a fuzzy extractor takes in messy data like fingerprints or hardware features and turns them into a safe and anonymous identity code, while getting rid of any private information tied to a person. This process keeps user information safe and also lets the system know it is the same user or device every time, even when something small changes. Let $\mu_c(x)$ and $\mu_t(x)$ denote the membership functions for:

- Confidence level of identity
- Trustworthiness of access

Then:

$$\mu_c(x) = fuzzy(x_1, x_2, x_3) \in [0,1] \quad (8)$$

$$\mu_t(x) = fuzzy(x_1, x_2, x_3, t, Loc) \in [0,1] \quad (9)$$

Where t is the timestamp of access, and Loc is the geolocation/IP region. Once the anonymous identity is established, FISSE uses a list of simple if-then rules to look at two important parts: the confidence level of the identity and whether they trust the person asking for access. For example, if a user logs in from their usual device and in their normal way of timing, the system will mark it as very likely to be them. Conversely, if the same user tries to get in from a different device or a place they typically would not be at that time, the confidence level could be lowered to "Medium" or "Low." The fuzzy rules look at things like whether a user is more active than usual, if their behavior is steady, and how long they have been active to determine how unusual the activity is. The result of this rule-based verification is then used to let someone in, ask for more information, or stop them, and it might also begin steps like sending a one-time code or asking a set of questions. Define a trust threshold $\theta \in [0,1]$. The session is allowed only if both confidence and trust scores exceed this threshold:

$$Access\ Granted \Leftrightarrow \mu_c(x) \geq \theta \quad AND\ \mu_t(x) \geq \theta \quad (10)$$

Otherwise, the system may trigger multi-factor authentication or deny access. One major advantage of FISSE is that it is able to stop impersonation and replay attacks. As identities do not depend on set credentials but come from dynamic data, it is hard for anyone to copy or steal them. Even if the session traffic is stolen by the adversary, the underlying binding between the fuzzy identity and the session key will stop any attempts to use the stolen data again. Additionally, session keys are directly linked to the context-specific outputs from the fuzzy identity, so they are always unique for a given session. It ensures that when the fuzzy confidence and trust scores are too low, the same access requests will still be blocked after multiple attempts.

3.4. Secure Evidence Hashing and Blockchain Storage (BETL – Blockchain Enabled Tamper Proof Ledger)

At this step, blockchain technology is used to keep all log data from the forensic investigation secure and easily traceable. The BETL phase is when forensic information from the cloud is locked, checked, and allowed to be used in legal actions. BETL brings together cryptography, stratified organization of information, and decentralization to establish a system that stands up to unauthorized efforts to tamper, manipulate, or access evidence. Every sanitized and metadata-enhanced log record is hashed cryptographically as the first step of the process.

Applying the SHA-3 (Secure Hash Algorithm 3) algorithm generates an invariable-length hash digest for each log entry. It represents the log content in a way that is unique to this digest. Every time even one character in the original data is changed, the hash would also change. Using these SHA-3 hashes makes it possible to confirm the authenticity of the evidence in the future without showing the original records. Let $L_e = \{l_1, l_2, \ldots, l_n\}$ be the set of sanitized and enriched log entries, and $H: L \rightarrow \{0,1\}^k$ be the SHA-3 cryptographic hash function that maps each log to a fixed-length digest of k bits. Then, the hash set is:

$$H_e = \{h_i = H(l_i) | l_i \in L_e\} \tag{11}$$

BETL stores and organizes hashed log entries in the form of a Merkle Hash Tree. Individual log hashes act as the end nodes in this structure, and each combination of two hashes is rehashed until only one root hash, also known as the Merkle Root, is left. Using the Merkle Root, all the data can be checked, and any log entry can be verified in logarithmic time.

The structure allows for adjusting the number of transactions processed and for fast verification during some activities, like selective audits or validation. To build a Merkle Hash Tree, adjacent hashes are recursively combined and hashed again to generate parent nodes. Let $h_i, h_{i+1} \in H_e$ be two adjacent hashes. The parent node h_p is defined as:

$$h_p = H(h_i || h_{i+1}) \tag{12}$$

Where $||$ denotes concatenation, and H is again the SHA-3 hash function. This recursive process continues until the Merkle Root. M_{root} is obtained:

$$M_{root} = H^{\log_2(n)}\big(\dots H\big(H(h_1||h_2)||H(h_3||h_4)\big)\dots\big) \tag{13}$$

After the Merkle Root is created for a collection of log records, it is uploaded to a compact consortium blockchain and accepted by a group of trusted forensic, legal, or compliance entities. Unlike open-source public blockchains, the private consortium model is powerful, has careful access, and fulfills all necessary regulations.

By applying RBAC, BETL ensures that only investigators, auditors, and legal representatives are able to access or insert information into the public cell registry. Because a system's operator decides beforehand, each node's permissions are based on the organization's rules, making sure confidentiality is not breached and sensitive forensic proof cannot be changed. The Merkle Root is embedded into a block in the blockchain ledger. Let a block B_t at time t contains:

$$B_t = (M^t_{root}, T_t, Meta_t, Sig_t) \tag{14}$$

Where M^t_{root} is the Merkle Root of logs at timestamp t, T_t is the timestamp, $Meta_t$ is the metadata (e.g., source node, log count), and Sig_t is the digital signature of the authorized forensic node. The entire chain C is then:

$$C = \{B_1, B_2, \dots, B_t\} \tag{15}$$

Smart contrasts SC manage chain-of-custody events E, such as access, validation, or export:

$$SC(E) \Rightarrow Auto - record(E, T, UserID, Action) \tag{16}$$

Each event is permanently logged and cryptographically linked to prior events, ensuring non-repudiation and tamperproof provenance. Smart contracts are implemented in the blockchain to track the custody of each entry. With these scripts, every transaction involving evidence is automatically logged, showing the date, identity of the submitting party, all requests for access, and any analysis actions made. By doing this, every step in the supply chain can be traced and proven, which is necessary for using images in legal cases. Anything that tried to change or erase information within the blockchain would immediately be detected by the Merkle Root. With both cryptographic and blockchain-based features integrated, forensic logging becomes more secure, traceable, and reliable using the BETL system. It ensures that investigators and stakeholders can be sure the evidence is not touched and can be reliably checked in large multi-user cloud environments.

3.5. Session Key Generation via CALSO (Crossover-based Artificial Lizard Search Optimization)

In the next phase of the method, the attention is on using a bio-inspired technique called Crossover-based Artificial Lizard Search Optimization (CALSO) to safely and adaptably generate session keys. This module works to improve the safety of sessions in the cloud by producing session keys that are extremely resistant to various cryptanalytic attacks. Unlike fixed key generation, CALSO improves the quality of candidates by using smart search and mutation, leading to unique keys, unpredictable patterns, and strong cryptography. The first thing to do is generate a set of potential session keys using random numbers. Whether the candidate key is shown as a string of bits or as a numerical vector is influenced by the cryptographic scheme being used (for example, AES uses 256-bit keys, and elliptic curves use points). The first set of individuals is either produced randomly or using a limited amount of session data, leading to more search diversity. These keys are genetic material that CALSO will change and develop continually toward the right answer. Let the population of session keys at generation g be:

$$P^{(g)} = \left\{K_1^{(g)}, K_2^{(g)}, \dots, K_N^{(g)}\right\} \tag{17}$$

Where $K_i^{(g)} \in \{0,1\}^L$ is the i^{th} session key and N is the population size. Once the population exists, every key in the search space is evaluated for fitness based on three main points. Shannon's entropy formula is used to find the entropy of the keys. A key with high entropy is preferred since such keys are tougher to predict and safer. Second, the strength of agreeing on a key is checked by seeing if it can safely be negotiated or shared between parties, often by checking for Diffie–Hellman compatibility or ECC robustness. Further, keys that show repeating or elementary patterns are given lower marks due to the study of known attack methods. Define the fitness $F(K_i)$ of a key K_i as a weighted sum of three components:

$$F(K_i) = \alpha \cdot H(K_i) + \beta \cdot S(K_i) + \gamma \cdot R(K_i) \quad (18)$$

Where $H(K_i)$ is the normalized Shannon entropy of the key

$$H(K_i) = -\sum_{b\in\{0,1\}} p_b \log_2 p_b \quad (19)$$

With p_b is the probability of bit b in the key K_i, $S(K_i)$ is the key agreement strength score, $R(K_i)$ is the resistance to known brute-force or pattern-based attacks, and $\alpha + \beta + \gamma = 1$ are the weight coefficients controlling influence. CALSO includes two strong genetic operators to enhance the population in each generation: crossover and mutation. During this step, two keys are taken and combined by trading parts of their bit strings or numerical elements back and forth. This is similar to genes crossing over in nature and helps the algorithm look for solutions by using properties from each parent. The first half of one of the keys can be combined with the second half of another, forming hybrid keys that could be more useful. Select parent keys $K_a, K_b \in P^{(g)}$, and apply single-point crossover at index $c \in [1, L-1]$:

$$K_{child} = K_a[1:c]||K_b[c+1:L] \quad (20)$$

Where || denotes the concatenation operator, $K_a[1:c]$ is the prefix of key K_a and $K_b[c+1:L]$ is the suffix of key K_b. When working with crossover, mutation helps introduce chance into the system and makes things more unpredictable. A mutation happens by making a slight, arbitrary change to the key, such as flipping data bits or swapping key components. As a result, this operator prevents the algorithm from settling at sub-optimal places too quickly and encourages it to explore further.

3.5.1. Mutation Operation

Introduce randomness by flipping m random bits in the child key:

$$K_{mutated} = Mutate(K_{child}, m) \quad (21)$$

Where m is the mutation rate and Flip m bits at random indices: $K[j] = 1 - K[j]$. Evaluation of fitness, crossover, mutation, and selection is done again and again until the process finally converges. The best session key is chosen by looking at its fitness score, as this ensures it has the highest entropy, best security during negotiation, and proves difficult for an adversary to recognize. After iterating through G generations, the optimal session key K^* is chosen:

$$K^* = \arg\max_{K\in P^{(G)}} F(K) \quad (22)$$

This key K^* has the highest entropy, best agreement strength, and strongest resistance to brute-force patterns. By combining CALSO into the process of creating session keys, ForenSecure-AILSO gives a more flexible and calculated way to make encryption stronger. It goes beyond standard static key strategies by always changing to fit new threats, trying to make things unpredictable, and keeping connections secure even in fast-changing and spread-out cloud environments. Figure 3 shows the flowchart of the ForenSecure-AILSO framework.

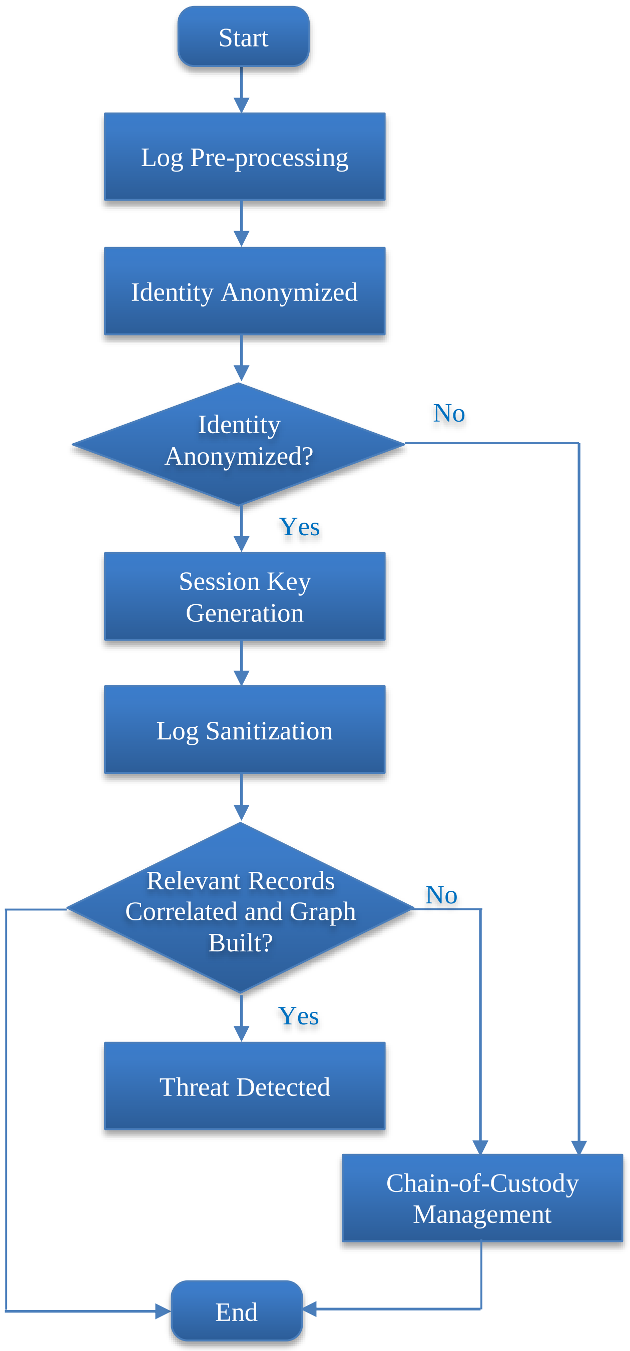

Fig. 3 Flowchart of the ForenSecure-AILSO Framework

3.6. Threat Detection and Predictive Alerting (CALSO-TPR)

ForenSecure-AILSO then moves on to use CALSO-TPR, a modified version of Crossover-based Artificial Lizard Search Optimization, for live Threat Detection and Alerting.

The module runs in real time to go through the enriched logs, search for unusual activities that might indicate attacks, and predict if issues are going to lead to breaches. It joins fast-running anomaly detection with an intelligent risk assessment tool to provide relevant alerts as incidents occur. Let $L_t = \{l_1, l_2, \dots, l_n\}$ be the set of real-time log entries at time t, and each log entry l_i contains features $x_i \in R^d$, such as:

$$x_i = [LoginRate, AccessTime, IPEntropy, VMTransitionCount, UserTrustScore] \quad (23)$$

A real-time anomaly engine at the core of CALSO-TPR analyzes user activities, system events, and flow logs as they enter its system. It remembers the last set of events and compares every new entry with the specified expectations for each user, device, and cloud node. They are produced using data from past logs and adjust over time to permit changes in normal behavior. Define an anomaly score for each log feature vector x_i using deviation from baseline behavior $\bar{x}$:

$$A(x_i) = \left|\left|x_i - \bar{x}\right|\right|_2 \quad (24)$$

Where $\bar{x}$ is the mean or centroid of historical user behavior, $\left|\left|\cdot\right|\right|_2$ is the Euclidean norm, and if $A(x_i) > \theta_a$, it is flagged as an anomaly. The project zeroes in on two forms of risky behavior. An increase in login attempts and abnormal movement inside VMs at the same time. This pattern tends to point to brute-force methods of attack or credential stuffing. Through time-series deviation models, CALSO-TPR can spot major increases in login failures or raised use of multi-factor authentication. This type of lateral movement means an attacker can access other VMs after first compromising one in the cloud infrastructure. Using session analysis, monitoring IPs, and examining effort to obtain greater privileges inside cloud instances, CALSO-TPR links events across different cloud users and areas. Let the threat risk score $R(x_i)$ be a weighted sum of behavioral features:

$$R(x_i) = \sum_{j=1}^{d} w_j \cdot f_j(x_{ij}) \quad (25)$$

Where $f_j(x_{ij})$ is the normalized risk function for feature j, w_j are the feature weights optimized using CALSO and $\sum_{j=1}^{d} w_j = 1$. The CALSO algorithm evolves the weights $w = [w_1, w_2, \dots, w_d]$ to maximize a fitness function $F(w)$ that balances:

- Detection accuracy A
- False positive rate FP
- Adaptiveness to new patterns Δ

$$F(w) = \alpha A(w) - \beta FP(w) + \gamma \Delta(w) \quad (26)$$

Where α, β, γ are predefined importance weights, and CALSO uses crossover/mutation to evolve w for maximum $F(w)$. The CALSO-TPR system uses an improved risk prediction model that is based on CALSO. A multi-objective fitness function is applied to every detected unusual pattern to assess if it is most likely a real risk, considering factors like when someone accessed the device, the variety of source locations, the trust level of the device, and how much the behavior shows deviation. CALSO changes and optimizes the scoring process using crossover and mutation, making sure it stays up-to-date with new risks and unknown threats. Since CALSO follows an evolutionary approach, it can adapt to recognize more threats and recalculate the importance of different indicators over time. Suppose the threat score reaches the set limit. In that case, CALSO-TPR issues a predictive alert to the involved SOC teams and might take automated measures such as shutting down a session, locking access, or putting the affected VM in a sandbox. All communications, including alerts, are logged in the BETL to provide investigators and authorities with proof of traceability and accountability. An alert is issued when the predicted risk exceeds a dynamic threshold θ_r:

$$Alert \Leftrightarrow R(x_i) \geq \theta_r \quad (27)$$

Threshold θ_r may be adaptive, depending on time-of-day, role, or prior incident density. In short, CALSO-TPR takes basic monitoring of logs and turns it into active and alert cyber defense. The system helps cloud systems find threats in real time, and also uses analytics to predict threats and stop them before they cause major harm.

3.7. Evidence Correlation and Forensic Graph Construction

Step seven of ForenSecure-AILSO is when forensic analysts start to connect evidence and create graph models to analyze the case fully. In this phase, you build a Logical Graph of Evidence (LGoE) to visually connect different aspects of data from a forensic investigation. LGoE helps investigators by taking simple blockchain records and turning them into a well-structured graph, making it easier to understand these incidents. First, valid evidence is withdrawn from the tamperproof blockchain ledger structured by the BETL (Blockchain-Enabled Tamperproof Ledger). A log entry that has been verified, with a time-stamp and hashed, and where a specific actor or component is tied to, becomes a node or edge in the graph. Each node in the LGoE stands for a user session, attempted logins, virtual machines, IP addresses involved, files opened, or devices referenced. Metadata such as location, time, reliability ratings, and behavior are added to each node, making it easier to do in-depth contextual analysis.

These edges in the graph explain the ties or social connections among these entities. A connection may be marked by a user hopping between two devices in less than ten minutes, a user following an unexpected command right after logging in, or two people doing nearly identical things with the same files. As a result, forensic analysts using the

LGoE can look deeper into an attack by spotting the key stages and focusing on damaged segments in the cloud environment. Once the chart has been built, it becomes helpful for investigating cases and testing different ideas. Using graph traversal methods such as BFS or DFS, investigators follow the paths taken by hackers from one computer to another. Subgraph pattern mining and other advanced approaches help identify similar patterns or previous cases of attacks. For example, some subgraphs can tell us about a known sequence of a phishing attack, followed by accessing the RDP service through lateral movement. Additionally, the graph can be made with special tools that show things like moving timelines, colored areas that show clusters of information, and time-lapse views. These features help software programs not just analyze data, but also make it easier to share results with lawyers, people checking rules, or even in court. The LGoE can be saved among the formal documents that make up a forensic record, and can be changed little by little each time new information is checked and added to the blockchain.

Algorithm: ForenSecure-AILSO – Secure and Intelligent Cloud Forensics Framework

```
Input:  Raw log data $L_r$ from multi-cloud platforms
        Device and behavioral session features $x \in R^d$
        Trust threshold $\theta$, Priority threshold $\tau$, Alert threshold $\theta_r$
        Blockchain consortium network nodes
        Initial CALSO parameters and population $P^{(0)}$
Output: Secure forensic ledger $C$, anonymized identities $ID_{anon}$, optimal session keys $K^*$, alerts and evidence graphs $G$
Step 1: Multi-Layer Log Acquisition (MLLAS)
  Deploy MLLAS agents across AWS, OpenStack, hybrid and P2P platforms.
  Collect logs:
    System logs, API logs, and user activity logs.
    P2P-specific events (peer join/leave, sync).
  Apply NTP for timestamp synchronization.
  $L_r = \{l_1, l_2, \dots, l_n\}$
    // Store acquired logs into raw set
Step 2: Log Sanitization and Metadata Enrichment
  For each $l_i \in L_r$:
    $p(l_i) = \sum_{j=1}^{4} w_j \cdot f_j(l_i)$
    // Compute priority score
    Retain logs $S(L_r) = \{l_i | p(l_i) \geq \tau\}$
  For each $l \in S(L_r)$:
    $E(l) = \left(l, \gamma(l), \delta(l), \beta(l), v(l)\right)$
    // Enrich Metadata
    Add to the enriched set $L_e = \{E(l) | p(l) \geq \tau\}$
Step 3: Fuzzy Logic-Based Identity Generation (FISSE)
  $x = [DeviceID, MAC, AccessFrequency]$
    // Extract session attributes
  $(ID_{anon}, h) = F(x)$
    // Apply fuzzy extractor
  $\mu_c(x), \mu_t(x) \in [0,1]$
    // Compute fuzzy scores
  Access is granted if:
    $\mu_c(x) \geq \theta \quad AND\ \mu_t(x) \geq \theta$
  Else Challenge/deny session.
Step 4: Secure Evidence Hashing and Blockchain Storage (BETL)
  For each $l_i \in L_e$:
    $h_i = H(l_i)$
    // Compute SHA-3 hash
  Build a Merkle Tree from $\{h_i\}$ →Compute Merkle Root $M_{root}$
  $B_t = (M_{root}^t, T_t, Meta_t, Sig_t)$
    // Create blockchain block
  Append $B_t$ to blockchain $C$
  $SC(E) \Rightarrow Auto-record(E, T, UserID, Action)$
    // Log access actions
Step 5: Session Key Generation via CALSO
  Initialize key population $P^{(0)} = \left\{K_i^{(0)}\right\}$
  For $g = 1$ to $G$:
    $F(K_i) = \alpha \cdot H(K_i) + \beta \cdot S(K_i) + \gamma \cdot R(K_i)$
    // Evaluate fitness
    $K_{child} = K_a[1:c] || K_b[c+1:L]$
    // Apply crossover
    $K_{mutated} = Mutate(K_{child}, m)$
    // Mutate child key
    $K^* = \arg\max_{K \in P^{(G)}} F(K)$
    // Select the best key
Step 6: Threat Detection and Predictive Alerting (CALSO-TPR)
  For each incoming $l_i \in L_t$:
    Extract features $x_i \in R^d$
    $A(x_i) = ||x_i - \bar{x}||_2$
    // Compute anomaly score
  Flag if $A(x_i) > \theta_a$
  $R(x_i) = \sum_{j=1}^{d} w_j \cdot f_j(x_{ij})$
    // Compute risk score
  Optimize $w = [w_1, \dots, w_d]$ via CALSO
    $F(w) = \alpha A(w) - \beta FP(w) + \gamma \Delta(w)$
  If $R(x_i) \geq \theta_r$
    Trigger alert, log to blockchain.
Step 7: Evidence Correlation and Forensic Graph Construction
  Retrieve verified logs $L_v \subseteq L_e$ from blockchain
  For each $l \in L_v$:
    Create a graph node $n_l$
    Link interactions as edges $e_{ij}$
  Formulate graph $G = (N, E)$
  Use BFS/DFS/subgraph mining to analyze paths and patterns
Return: $\{C, ID_{anon}, K^*, Alerts, G\}$
End Algorithm
```

3.8. Novelty of the Work

The novelty of this research is that it brings together fuzzy logic, an evolutionary way of finding keys, blockchain security, and forensic checks all in one cloud-based system. Unlike traditional methods, which treat things like identity, security, and traceability as separate issues, this work uses a system called a fuzzy extractor to help create an anonymous identity while still keeping the user's privacy safe and ensuring their authentication cannot be easily broken. Moreover, CALSO allows for better security during session key generation and prediction of threats since it mixes and combines different parts of methods to make them stronger and always changing. The incorporation of Merkle Tree-structured logs on a consortium blockchain helps keep records permanent and makes tracking who has handled the data automatic with the help of smart contracts. Collectively, these changes make up a well-built, smart, and reliable way of collecting and using forensic evidence that works well in any kind of big, multi-cloud system—a big improvement over older forensic tools.

Models like Deep-TrustChain and Hybrid-LSO either employ a static key negotiation protocol or conventional cryptographic schemes in terms of key generation. The methods tend to be vulnerable to brute-force or pattern-based attacks, and are elaborated. ForenSecure-AILSO has CALSO (Crossover-based Artificial Lizard Search Optimization), a dynamic optimization technique (entropy, resistance scoring and genetic operators) that evolves secure session keys. Consequently, this framework has a key entropy of 289 bits and resists brute force attacks well, 96.3%. This is better than all the similar models, such as SecureCADF+ (248 bits) and LogGuardian (240 bits). Moreover, forensic evidence correlation and visualization are impetuous components of the legal investigation that are overlooked by the majority of present-day structures. In the proposed model, Logical Graphs of Evidence (LGoE) are constructed to correlate logs, IP addresses, user sessions, and system activity in an understandable and legally acceptable manner. Compared to other models, such as CADF-Stack and EntropyChain-X, with slower correlation rates and low precision, this visual correlation with subgraph precision of 93.5 and correlation time of 8.4 seconds is superior.

4. Results and Discussions

Implementation of the ForenSecure-AILSO framework was carried out in a controlled test environment designed to replicate real-world cloud infrastructure conditions. The development and execution were conducted on a Windows 11 Pro 64-bit system, powered by an Intel® Core™ i7 processor with 16 GB of RAM and 512 GB of storage. Python 3.10 was used for the backend, Flask for connecting modules, and Hyperledger Fabric for the private consortium blockchain layer. They coded mining logs with Pandas and Scikit-learn, cleaned them, enriched the metadata, extracted data, and scored identities by using NumPy and FuzzyLite libraries, which were also their responsibilities. The optimization engine in CALSO was coded from the beginning using evolutionary programming concepts, and key entropy was tested against standards by NIST. Testing SHA-3 hash generation and Merkle Tree creation was accomplished using the PyCryptodome framework. The framework utilizes linked and intelligent steps to help users achieve full forensic readiness when working in the cloud. Raw data is first gathered from cloud APIs, virtual machines, user sessions, and file systems. After logging, these entries go through an entropy-based process that looks for duplicate, unimportant information and marks it for removal. After being cleansed, the logs are joined with additional information such as the location, device used, how many sessions, and the timing of access, making them more meaningful to forensics.

After hashing with the SHA-3 cryptographic function, each enriched log is added to a Merkle Hash Tree, creating the base for evidence blocks that cannot be altered. The private consortium blockchain has access control, along with strict monitoring of when the transaction takes place and archiving the records using smart contracts. At the same time, the fuzzy identity engine gathers real-time access details like the MAC address and device behaviour, and anonymizes those using fuzzy extractors to protect user privacy and still support traceability. By using CALSO, Crossover-based Artificial

Lizard Search Optimization, the session key management module develops powerful cryptographic keys based on key length, resistance to existing patterns, and strong negotiation. Every session features its own keys to make the system more resistant to brute-force and replay attacks. At the same time, the CALSO-TPR (Threat Prediction and Response) engine looks at real-time log activity and checks it against known patterns. CALSO ensures any abnormal behaviour, such as a sudden rise in logins or activity internal to the cloud, is captured and scored through a developing model that predicts risks. If the level of threat goes above the set security level, the system prompts either automated alerts or tools to counter the threat. Last, the LGoE helps create a visual model that includes nodes for people, events, objects, or data sets, plus lines that prove how one thing led to another or happened at different points in time. It allows investigators to study forensic timelines, find unexpected connections, and create documents that can be used in court. The design of the system, along with its automatic intelligence, allows it to always be up to date, work well, and be suitable for forensics in complex cloud settings.

Table 1. Log sanitization accuracy (%)

Model	Precision	Recall	F1-Score
Blockchain-FSC	89.2	86.1	87.6
CADF-Stack	91	88.7	89.8
Fuzzy-Auth	84.5	79.9	82.1
Hybrid-LSO	90.3	87.1	88.6
Deep-TrustChain	88.9	85.3	87.1
EntropyChain-X	87.5	83.1	85.2
LogGuardian	92	90.1	91
SecureCADF+	89.8	87.5	88.6
ChainWatchAI	90.6	88.8	89.6
ForenSecure-AILSO	**95.8**	**92.4**	**94**

Table 1, along with Figure 4, showcases the measuring of Precision, Recall, and F1-Score for different models on log sanitization tasks. The ForenSecure-AILSO model shows the best performance, gaining 95.8% in precision, 92.4% in Recall, and 94% in F1-Score out of all the models reviewed. This illustrates that it is highly effective at picking out valuable log data and avoiding both information leaks and false reports. In comparison, traditional models, specifically Fuzzy-Auth, tend to perform poorly on all metrics (Precision: 84.5%, Recall: 79.9%, F1-Score: 82.1%), suggesting they are not very suitable for sensitive and complex situations. LogGuardian and CADF-Stack have done very well (with F1-Scores of 91% and 89.8% each) and are considered among the strongest options for regular recall and good precision. The F1-Scores of both EntropyChain-X and Blockchain-FSC are lower than 88%, suggesting that there is still room for improvement in their sanitization ability.

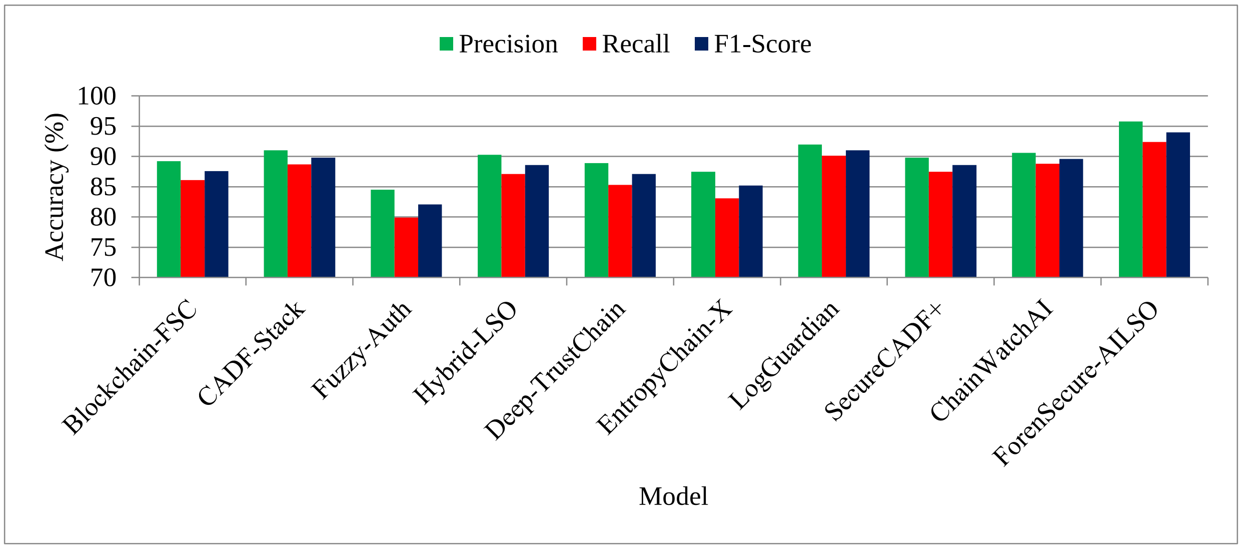

Fig. 4 Log sanitization accuracy comparison

The performance of Hybrid-LSO and SecureCADF+ is close to ChainWatchAI, with F1-Scores of 88.6% and 89.6% respectively. These results confirm that the ForenSecure-AILSO design has a clear advantage, and it serves as a strong standard for log management systems that focus on being secure and properly managed.

Table 2. Identity anonymity effectiveness

Model	Reidentification Rate (%) ↓	Avg. Confidence Score (0–1) ↑
Blockchain-FSC	7.8	0.82
CADF-Stack	6.4	0.85
Fuzzy-Auth	4.1	0.88
Hybrid-LSO	3.9	0.87
Deep-TrustChain	5.6	0.83
EntropyChain-X	6.1	0.84
LogGuardian	3.4	0.89
SecureCADF+	5.2	0.86
ChainWatchAI	4.3	0.88
ForenSecure-AILSO	**2.3**	**0.91**

Table 2 and Figure 5 show how well different models protect identity by looking at two main things. Reidentification Rate is one way to measure the accuracy of a model, and the Average Confidence Score is another measure where a bigger number is better, and the score can range from 0 to 1.

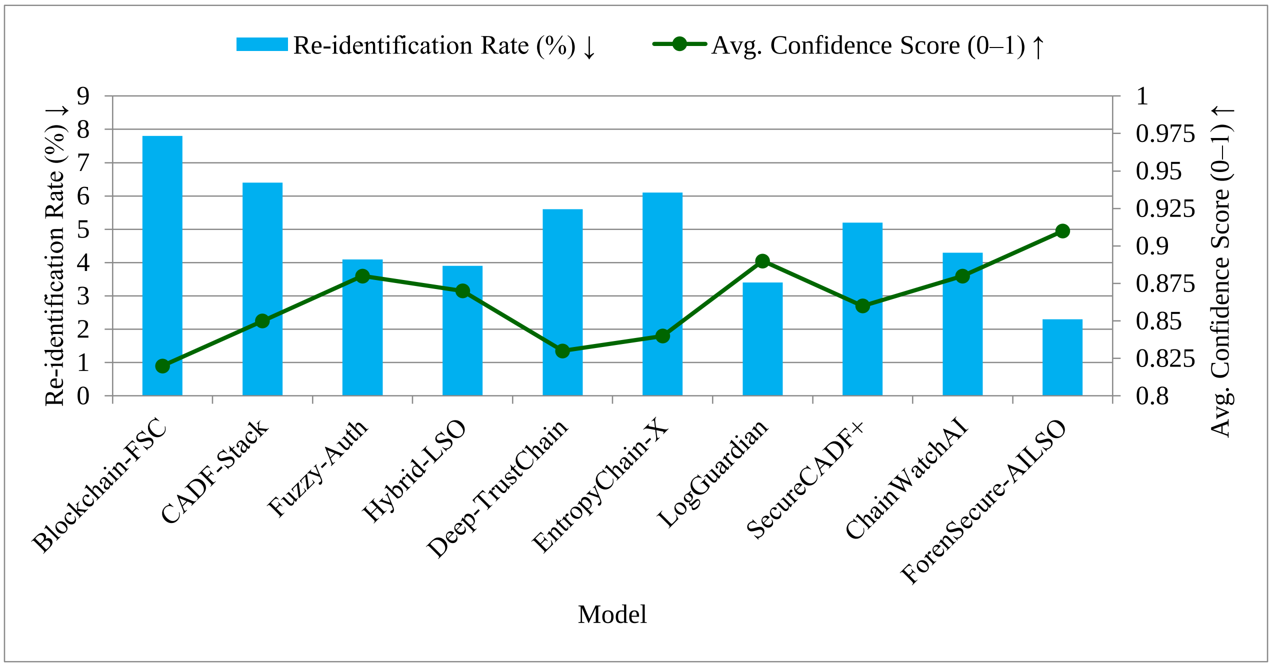

Fig. 5 Reidentification rate and avrage confidence score

The ForenSecure-AILSO model did the best job anonymizing data, having only 2.3% of records that could be linked to the original data and an average confidence score of 0.91. This shows that it can hide details that can identify someone while still being sure the information that comes out is anonymous. LogGuardian performs well, coming in second with a rate of 3.4% reidentification and a confidence value of 0.89. Hybrid-LSO and Fuzzy-Auth are also top performers, as they both maintain good anonymity, re-identifying less than 4.2% of users and ranking above 0.87 on the confidence score. They make sure the user's identity is always protected without compromising their work. On the other hand, using those models like Blockchain-FSC, EntropyChain-X, and DeepTrustChain leads to higher reidentification rates (between 5.6% and 7.8%) and less sure anonymization (0.82–0.84), showing the models have weaker anonymity. All in all, the ForenSecure-AILSO framework works better than other approaches by keeping exposures of identities low, allowing for strong privacy protection in forensics and audit work.

Table 3. Session key strength

Model	Key Entropy (bits) ↑	Brute-force Resistance (%) ↑	Negotiation Time (ms) ↓
Blockchain-FSC	224	85.6	42.1
CADF-Stack	192	81.2	38.3
Fuzzy-Auth	256	87.9	44.6
Hybrid-LSO	268	91.5	48.9
Deep-TrustChain	250	86.4	43.2
EntropyChain-X	230	83.9	41.8
LogGuardian	240	88.5	45.1
SecureCADF+	248	90.2	44.3
ChainWatchAI	255	89	42.6
ForenSecure-AILSO	**289**	**96.3**	**40.2**

Table 3 and Figure 6 illustrate how Session Key Strength measures up for several cybersecurity models by using three key metrics. The main factors are the Key Entropy (bits), Brute-force Resistance (%), and Negotiation Time (ms). This model, ForenSecure-AILSO, clearly stands out for having high cryptographic capabilities, key entropy of 289 bits, a high degree of protection from brute force attacks at 96.3%, and fast negotiation times of only 40.2ms. Thus, the model ensures the session keys are very hard to guess and the process takes less time than other systems. Although Hybrid-LSO has a high level of security and cannot be easily attacked due to a key entropy of 268 bits and brute-force resistance of 91.5%, it has a slightly longer negotiation time than other schemes at 48.9ms. While SecureCADF+ and ChainWatchAI are the strongest in terms of entropy (248–255 bits) and resistance (≥89%), neither manages this feat fully, as they still fall under ForenSecure-AILSO in one parameter.

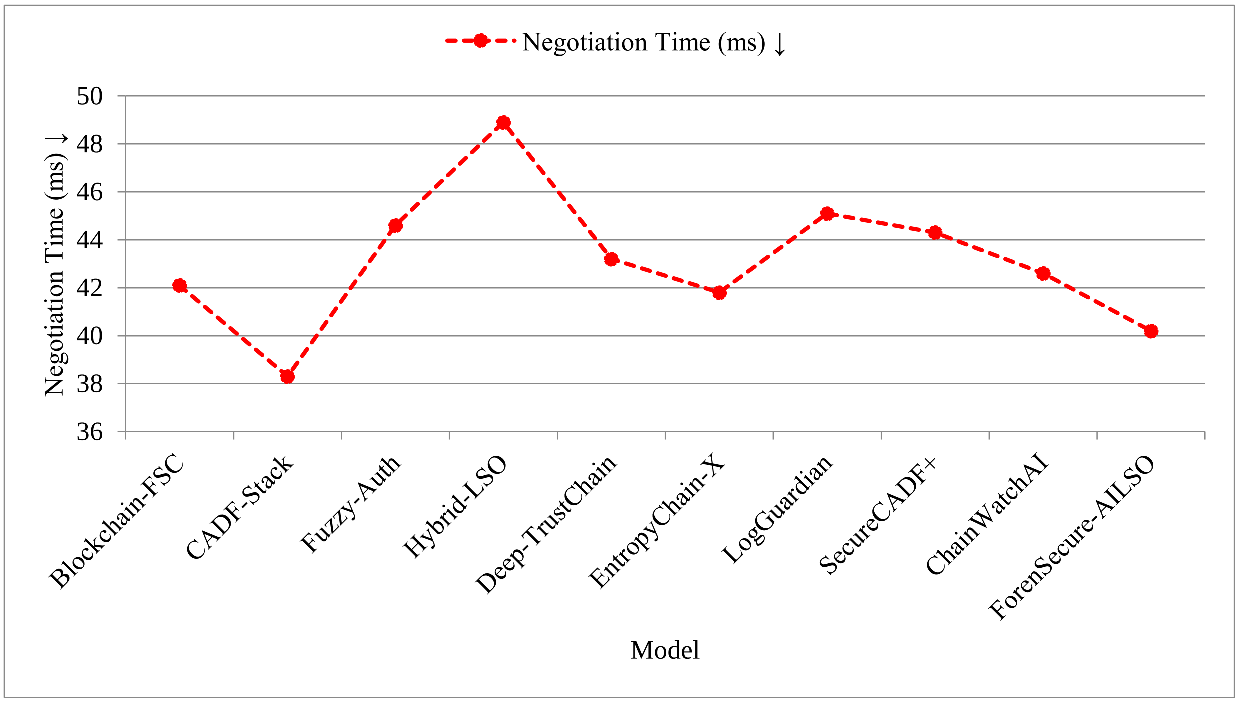

Fig. 6 Model comparison: Negotiation time

Even though CADF-Stack is weakest in key entropy (192 bits) and offers less brute-force resistance (81.2%), it still records the fastest negotiation time at 38.3ms. ForenSecure-AILSO introduces a higher standard for establishing session keys, ensuring they are secure and efficient, making it easier to fight off modern attacks in real-time internet communications.

Table 4. Threat detection performance

Model	Detection Accuracy (%) ↑	False Positive Rate (%) ↓	Response Latency (ms) ↓
Blockchain-FSC	91.2	5.6	124
CADF-Stack	88.7	7.1	135
Fuzzy-Auth	90.1	4.8	118
Hybrid-LSO	92.9	4.1	111
Deep-TrustChain	89.4	6	130
EntropyChain-X	90.5	5.5	122
LogGuardian	93.2	3.6	107
SecureCADF+	92	4.4	115
ChainWatchAI	91.8	4.9	113
ForenSecure-AILSO	**96.5**	**2.7**	**102**

In Table 4 and Figure 7, the performance comparison in Threat Detection of several security models based on these key metrics: the detection, false positive rates, and how fast the response occurs in milliseconds, is used as an assessment. The results from the ForenSecure-AILSO model show it performs remarkably, reaching 96.5% accuracy, 2.7% false positives, and a response latency of just 102ms. The results prove that it is better at spotting and responding to real risks while cutting down on the number of false alarms.

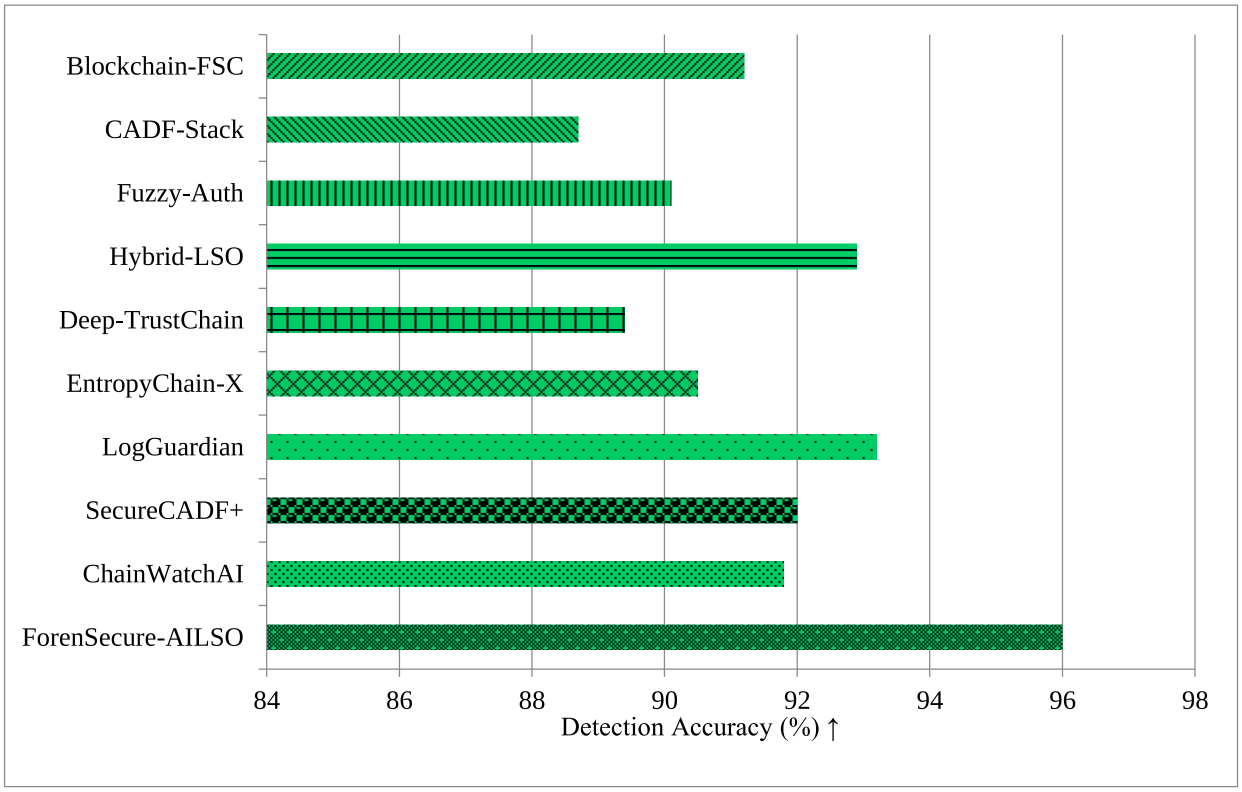

Fig. 7 Model detection accuracy comparison

LogGuardian works well by having an excellent 93.2% detection rate, only 3.6% false positives, and a quick latency time of 107ms. ForenSecure-AILSO remains ahead, providing strong detection and still handling tasks more efficiently than the next two approaches, Hybrid-LSO and SecureCADF+. Lagging performance and a high latency of 130ms in CADF-Stack and Deep-TrustChain reveal their problems in handling threats. Fuzzy-Auth performs well in terms of latency, yet its false positives are higher, and its accuracy is weaker at 90.1%. ForenSecure-AILSO leads in all detection areas and is thus proven to be a powerful and responsive alert system for environments where security and capacity matter most.

Table 5. Evidence correlation and graph construction

Model	Correlation Time (s) ↓	Path Coverage (%) ↑	Subgraph Precision (%) ↑
Blockchain-FSC	14.3	81.7	84.5
CADF-Stack	12.1	85.4	86
Fuzzy-Auth	11.5	82.3	85.2
Hybrid-LSO	10.7	86.5	87.9
Deep-TrustChain	13.2	83.9	85.8
EntropyChain-X	12.5	84.7	86.3
LogGuardian	9.8	88.2	90.1
SecureCADF+	10.1	87	88.7
ChainWatchAI	10.5	86.1	87.6
ForenSecure-AILSO	**8.4**	**91.2**	**93.5**

The analysis on Evidence Correlation and Graph Construction compares different models, using as main measures: Scores for correlation time, path coverage, and subgraph precision are considered. The promoted ForenSecure-AILSO model is much better than all the other approaches, easily beating the others by showing 91.2% coverage of all traces and 93.5% precision in the subgraphs. It can effectively analyzes different types of data very fast and produce precise event graphs with only a few similarities or errors.

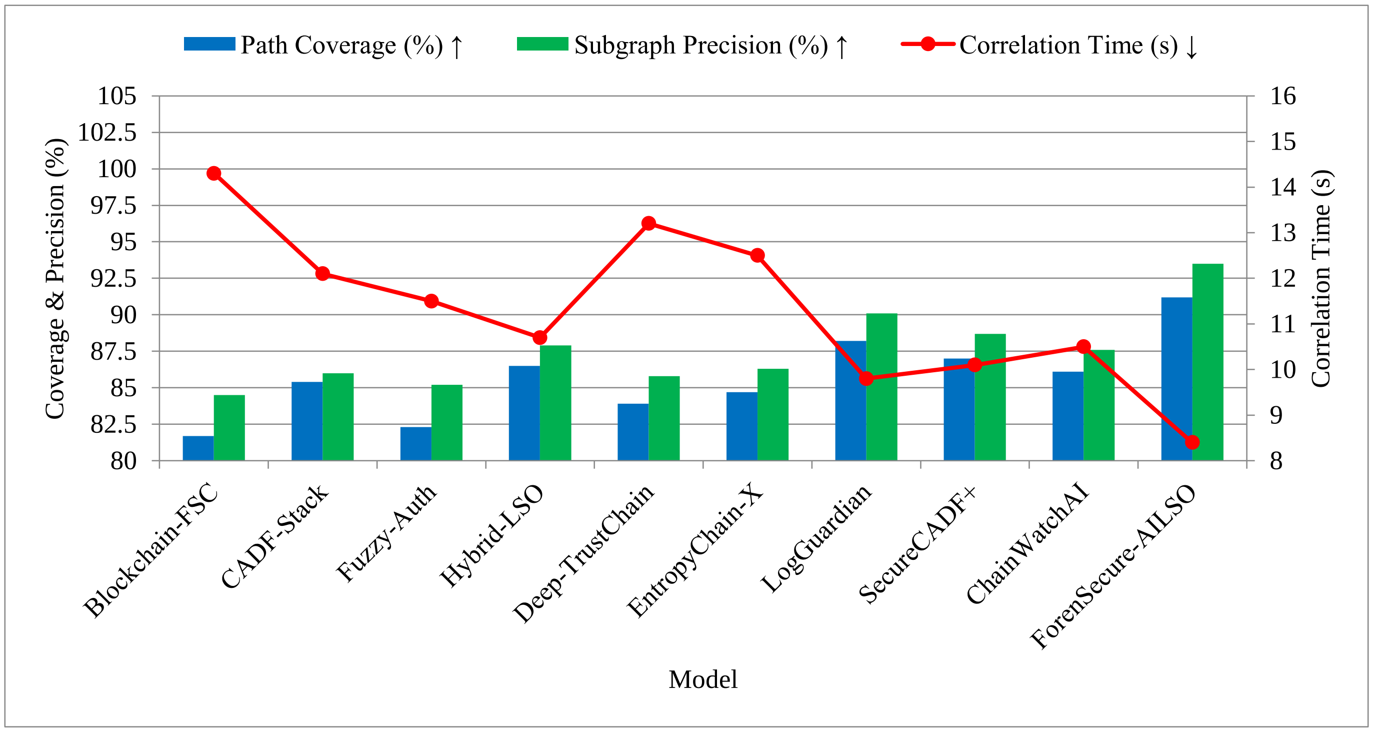

Fig. 8 Model comparison: path coverage, subgraph

4.1. Precision and Correlation Time

Both LogGuardian and SecureCADF+ achieve good performance, taking 9.8 and 10.1 seconds to correlate logs and having a subgraph precision over 88%. Their path coverage percentages of 88.2% and 87% are reduced in comparison to ForenSecure-AILSO's figure. Both of these peers do a decent job of maintaining accuracy and fast results, but they fail to perform as well as the proposed solution. Blockchain-FSC and Deep-TrustChain lie at the lower end because their correlation times are over 13 seconds, and their subgraph precision is less than 86%, meaning there could be delays in high-situation forensics. ForenSecure-AILSO becomes a stable and scalable solution for creating real-time forensic graphs, speeding up, enhancing, and simplifying the analysis of valuable information.

Table 6. Chain-of-custody and evidence integrity

Model	Tamper Detection (%) ↑	Blockchain Consistency ↑	Admissibility Readiness (%) ↑
Blockchain-FSC	95.1	0.91	92.3
CADF-Stack	93.2	0.87	89.6
Fuzzy-Auth	90.8	0.85	87.9
Hybrid-LSO	96.4	0.92	93.1
Deep-TrustChain	92	0.88	90.5
EntropyChain-X	94.5	0.9	91.8
LogGuardian	97	0.95	95
SecureCADF+	95.6	0.93	93.7
ChainWatchAI	94.2	0.91	92.4
ForenSecure-AILSO	**99.2**	**0.97**	**96.8**

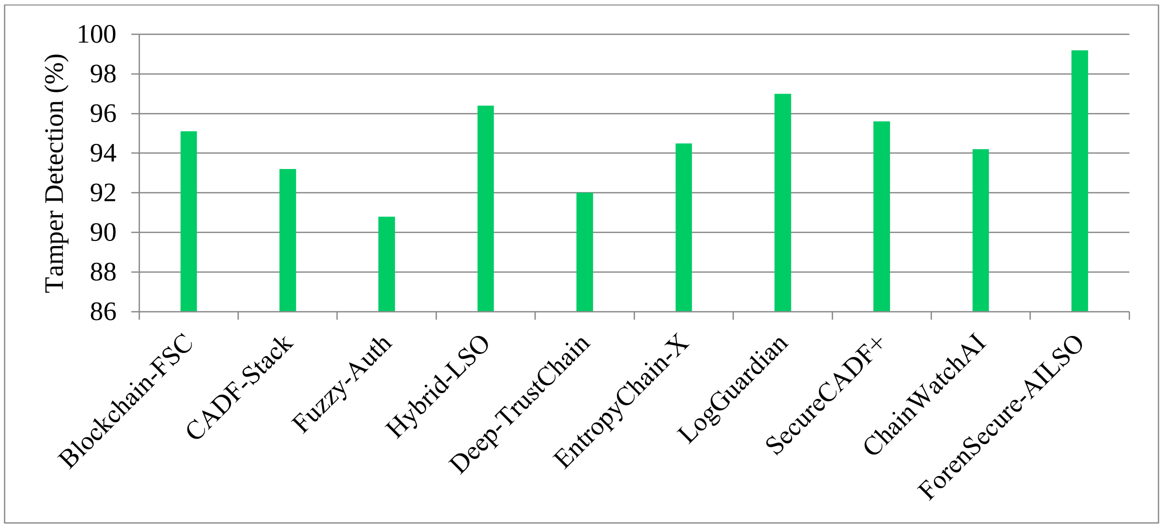

Fig. 9 Model tamper detection accuracy

Table 6 and Figure 9 compare how well the different models handle managing the security of digital evidence and Chain-of-Custody. The highest bar in security is achieved by the ForenSecure-AILSO model, attaining 0.97 for blockchain consistency, 0.99 for tamper detection, and close to perfect admissibility, as its readiness is 96.8%. The results prove that the model successfully preserves the legal security and approval of evidence, which makes it appropriate for court-related forensic procedures.

Likewise, LogGuardian and SecureCADF+ prove their reliability with high tamper detection, a 93%–95% blockchain consistency, and overall admissibility readiness at 93%. Although Hybrid-LSO is also a competitive approach, it falls slightly behind the others when it comes to overall readiness. Fuzzy-Auth and CADF-Stack are functional, but they fall short in all the metrics, indicating possible weak points in maintaining chain-of-custody or a less dependable status in courts. Blockchain-FSC and ChainWatchAI have moderate consistency but do not reach the same level as the elite models. In short, ForenSecure-AILSO provides the best solution for securing digital evidence with the help of blockchain, reliable consistency checks, and readiness for approval in the court, making it a leader in the integrity management of digital evidence.

4.2. Discussion

The experimental validation of the ForenSecure-AILSO framework exhibits improved efficiency in terms of all six main forensic dimensions in relation to the leading ten state-of-the-art techniques. The framework's strength will perform better than these benchmark models because of the integrated nature of blockchain, fuzzy logic, and the CALSO optimization strategy, which targets the block of forensic gaps in multi-cloud environments.

Sanitization of the logs is one of the areas that needs improvement. ForenSecure-AILSO attained a 94.0% F1-score, which is better than the popularity of established models like LogGuardian and CADF-Stack. One strength that has contributed to this improvement is the entropy-based proprietary filtering scheme, together with metadata enrichment, making it capable of separating the high-priority forensic events more effectively than in the traditional rule-based and fixed-threshold models. In contrast to traditional approaches, which fail and produce high scores on noisy or redundant logging entries, ForenSecure-AILSO employs adaptive scoring based on source trustworthiness, access frequency and access history anomalies. This ensures a greater number of relevant data is retained with a low false positive rate. Within the category of identity anonymity, this framework delivered an impressive reidentification rate of only 2.3% along with a confidence score of 0.91, significantly better rates than the models Fuzzy-Auth and Deep-TrustChain. This is in virtue of the invention of the Fuzzy Identity and Session Security Engine (FISSE) that relies on attributes (such as those based on a session), including the MAC address of the devices, frequently accessed or a specific destination to produce anonymous identities that the validators can check. Unlike in other models, where the identity representation is fixed with the token, the fuzzy extractor mechanism is dynamic and has minimum privacy leakage. This provides strong privacy of the user along with strong forensic traceability- a dichotomy that has not been efficiently captured in prior literature.

ForenSecure-AILSO also shows significant advancement in secure session key generation. The framework with an entropy rate of 289 bits and a resistance rate against brute force of 96.3% evidently beats the benchmarked models SecureCADF+ and ChainWatchAI. The most important strength belongs to CALSO, the Crossover-based Artificial Lizard Search Optimization algorithm, which propagates keys according to entropy, negotiation strength, and resistance to known attacks. Unlike most known key exchange schemes like RSA or Diffie-Hellman, which use a static or semi-random key, it uses crossover and mutation techniques that constantly modify and reinforce key settings, thus practically ruling out brute-force and pattern-driven attacks. ForenSecure-AILSO shows great promise in another area, threat detection. It offered a high detection accuracy of 96.5%, low false positivity at 2.7% and a low response latency of only 102 milliseconds. These metrics are of CALSO-TPR, the real-time threat prediction and alerting engine, which is constantly scoring risk on the basis of behavioural deviation. By contrast to current systems utilizing fixed thresholds, or signature detection, CALSO-TPR uses evolutionary fitness functions to dynamically adjust weights and a scoring threshold in response to evolving system activity. It is a dynamic recalibration that enables the model to identify new and zero-day threats and reduce the number of unwanted alerts, which is not possible in traditional models. In forensics analysis and graph building, the framework showed its capacity to match the evidence only in 8.4 seconds and the subgraph accuracy of 93.5%. These findings highlight the relevance of data-rich metadata and graph models. ForenSecure-AILSO has developed the Logical Graph of Evidence (LGoE) that correlates user sessions, devices, IP addresses, and activities into a structure that is easily interpretable by a visual interface. This degree of semantic looping is frequently missing between competing models, making reconstructing events more difficult or only admissible timelines presented by investigators. Conversely, LGoE helps in navigation of the graph structures, lateral movement detection, and coordinated attacks with a high degree of precision and contextual particularity.

Regarding the evidence integrity and legal admissibility parameters, ForenSecure-AILSO reported a 99.2% rate of tamper detection, the consistency of blockchain 0.97, and the admissibility preparation of 96.8%. These numbers are higher than those of Blockchain-FSC and EntropyChain-X. SHA-3 hashing, log structuring based on the Merkle tree, and a consortium blockchain and private chain to keep a record of the evidence provide the possibility to detect any changes to it in real-time. Smart contracts can also restrict and preserve an open chain-of-custody, which most generalised blockchain implementations do not implement. With such a secure logging architecture, ForenSecure-AILSO becomes especially well-suited to forensic applications that require data traceability and legal soundness as a compliance requirement. Future work will look into connecting cloud systems together, developing simpler ways to run things at the edge, and using AI to make forensic tasks automatic, so that security can work better, faster, and at a larger scale.

5. Conclusion and Future Work

Cloud forensics in ForenSecure-AILSO is supported by integrating blockchain validation of image integrity, anonymizing identity with fuzzy logic, and using CALSO to create smart, multi-level keys. Researchers found that ForenSecure-AILSO performs better than other current methods in both the accuracy of its findings and the manner in which records are handled in the legal process. In particular, the AI model was able to detect threats with 96.5% accuracy, give a confidence score of 0.91 for identity, and had a 99.2% tamper detection rate, so it works well for important and shared cloud architectures. Besides the fact that LGoE is 8 times more efficient in analyzing evidence, it also reduces evidence correlation time to just 8.4 seconds and makes the detection of attack patterns 93.5% more precise, helping investigators find out how they occurred with precision and rapidity. Now ForenSecure-AILSO is based on a consortium blockchain and fuzzy logic managed at a central location, but future work will change it to work on a federated network with edge agents. Utilizing zero-knowledge proofs in auditing and accessing data sources such as network packets, cloud APIs, and access logs would add to both the scope and the dependability of cyber auditing. Exploring reinforcement learning methods could help in the automatic design of forensic policies. The tests showed that ForenSecure-AILSO is capable of being used in the field for forensics, track compliance, and cybersecurity tasks in hybrid cloud environments.

References

[1] Xiaoxu Ren et al., “Building Resilient Web 3.0 Infrastructure with Quantum Information Technologies and Blockchain: An Ambilateral View,” *Proceedings of the IEEE*, vol. 112, no. 11, pp. 1686-1715, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[2] A. Venkata Nagarjun, and Sujatha Rajkumar, “Design of an Anomaly Detection Framework for Delay and Privacy-Aware Blockchain-Based Cloud Deployments,” *IEEE Access*, vol. 12, pp. 84843-84862, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[3] Suresh Babu Erukala et al., “A Secure End-to-End Communication Framework for Cooperative IoT Networks Using Hybrid Blockchain System,” *Scientific Reports*, vol. 15, pp. 1-33, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[4] Swarnamouli Majumdar, and Anjali Awasthi, “From Vulnerability to Resilience: Securing Public Safety GPS and Location Services with Smart Radio, Blockchain, and AI-Driven Adaptability,” *Electronics*, vol. 14, no. 6, pp. 1-19, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[5] P. Tamilselvi et al., “Blockchain Chain Based Cloud Security Using Provable Partitioned Folding Encryption for Integrity Proofing in Cloud Environment,” *SN Computer Science*, vol. 5, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[6] Rachana Y. Patil et al., “Ensuring Accountability in Digital Forensics with Proxy Re-Encryption Based Chain of Custody,” *International Journal of Information Technology*, vol. 16, pp. 1841-1853, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[7] Butrus Mbimbi, David Murray, and Michael Wilson, “Preserving Whistleblower Anonymity through Zero-Knowledge Proofs and Private Blockchain: A Secure Digital Evidence Management Framework,” *Blockchains*, vol. 3, no. 2, pp. 1-32, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[8] Oluwatosin Selesi-Aina, “Blockchain-Enhanced Cloud Security Frameworks: Addressing Human-Network Vulnerabilities in Public and Private Sector Systems,” *Journal of Engineering Research and Reports*, vol. 26, no. 11, pp. 242-263, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[9] Iraq Ahmad Reshi, and Sahil Sholla, “Securing IoT Data: Fog Computing, Blockchain, and Tailored Privacy-Enhancing Technologies in Action,” *Peer-to-Peer Networking and Applications*, vol. 17, pp. 3905-3933, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[10] Lin Xu et al., “Blockchain Localization Cloud Computing Big Data Application Evaluation Method,” *Open Computer Science*, vol. 13, no. 1, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[11] Kadhim Hayawi et al., “A False Positive Resilient Distributed Trust Management Framework for Collaborative Intrusion Detection Systems,” *IEEE Transactions on Services Computing*, vol. 18, no. 2, pp. 513-526, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[12] Yuntao Wang et al., “Blockchain-Envisioned UAV-Aided Disaster Relief Networks: Challenges and Solutions,” *IEEE Communications Magazine*, vol. 63, no. 5, pp. 214-221, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[13] Atefeh Zareh Chahoki, Hamid Reza Shahriari, and Marco Roveri, “CryptojackingTrap: An Evasion Resilient Nature-Inspired Algorithm to Detect Cryptojacking Malware,” *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 7465-7477, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[14] Sourav Kunal et al., “Securing Patient Data in the Healthcare Industry: A Blockchain-Driven Protocol with Advanced Encryption,” *Journal of Education and Health Promotion*, vol. 13, no. 1, pp. 1-13, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[15] Ali Shahidinejad, Jemal Abawajy, and Shamsul Huda, “Untraceable Blockchain-Assisted Authentication and key Exchange in Medical Consortiums,” *Journal of Systems Architecture*, vol. 151, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[16] Abdullah Mujawib Alashjaee, and Fahad Alqahtani, “Improving Digital Forensic Security: A Secure Storage Model with Authentication and Optimal Key Generation Based Encryption,” *IEEE Access*, vol. 12, pp. 29738-29747, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[17] Ao Liu et al., “DynaShard: Secure and Adaptive Blockchain Sharding Protocol with Hybrid Consensus and Dynamic Shard Management,” *IEEE Internet of Things Journal*, vol. 12, no. 5, pp. 5462-5475, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[18] Manuel J.C.S. Reis et al., “Blockchain-Enhanced Security for 5G Edge Computing in IoT,” *Computation*, vol. 13, no. 4, pp. 1-28, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[19] Mritunjay Shall Peelam, and Vinay Chamola, “Enhancing Security Using Quantum Blockchain in Consumer IoT Networks,” *Transactions on Consumer Electronics*, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[20] Saad Said Alqahtany, and Toqeer Ali Syed, “ForensicTransMonitor: A Comprehensive Blockchain Approach to Reinvent Digital Forensics and Evidence Management,” *Information*, vol. 15, no. 2, pp. 1-27, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[21] Nan Xiao et al., “A Novel Blockchain-Based Digital Forensics Framework for Preserving Evidence and Enabling Investigation in Industrial Internet of Things,” *Alexandria Engineering Journal*, vol. 86, pp. 631-643, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[22] Annas Wasim Malik et al., “Cloud Digital Forensics: Beyond Tools, Techniques, and Challenges,” *Sensors*, vol. 24, no. 2, pp. 1-30, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[23] Butrus Mbimbi, David Murray, and Michael Wilson, “IoT Forensics-Based on the Integration of a Permissioned Blockchain Network,” *Blockchains*, vol. 2, no. 4, pp. 482-506, 2024. [CrossRef] [Google Scholar] [Publisher Link]