IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511189

AD Monitor: A Privacy-Aware Endpoint Monitoring and User Behavior Analytics System for Insider-Threat Investigation

H Pramodh[1], Deepak Kumar G[2], Devaraj M[3], Lakshmi Narayana Reddy[4], Dr. Jagadish R M[5]

Department of Computer Science and Engineering, Ballari Institute of Technology & Management, Ballari, Karnataka, India[1-5]

Abstract: Insider threats and misuse of legitimate access is spread across operating-system audit logs, file-share activity, endpoint processes and browser events, making them hard to investigate. This paper introduces AD Monitor, a simple monitoring and investigation session that can be used in authorized Windows Active Directory laboratory environments. It integrates Windows Event Forwarding, endpoint process snapshots and life cycles events, privacy-preserving browser telemetry, deterministic detection rules and a CPU-based Isolation Forest model for user behavior analytics (UBA). A FastAPI backend normalizes and stores telemetry in SQLite and a React-based interface offers endpoint health, alerts and cases, process lineage, browser activity, an evidence-correlation timeline, and an educational attack library. The collection performed by the browser is intentionally limited to the navigation and download metadata of the domain only (not full URLs nor page content). The correlation engine collects matched and missing fields, competing candidates, and statements of what the evidence proves and does not prove. The project has automated tests for ingestion, rules, UBA, database behavior, privacy guards, process lifecycle handling and timeline correlation. The goal of AD Monitor is to serve as an explanatory teaching and investigating platform, not a production endpoint detection and response system. The small lab scope, the dependence of the telemetry and the requirement of cold-start of the UBA are discussed.

Keywords: Insider threat detection, user behavior analytics, endpoint monitoring, Windows Event Forwarding, process lineage, browser telemetry, Isolation Forest, digital investigation, privacy-aware security.

I. INTRODUCTION

Organizations rely on identity, endpoint, and file-access telemetry to understand suspicious activity performed through legitimate accounts. An insider-related incident may involve unusual access to sensitive files, repeated authentication failures, staging data for transfer, execution of a downloaded script, or activity outside expected working hours. In practice, the evidence for these actions often resides in different sensors and must be interpreted together. A single event rarely proves malicious intent.AD Monitor addresses this investigation problem in a controlled Windows Active Directory lab. It receives forwarded Windows security events and endpoint telemetry, applies deterministic rules and baseline-based anomaly detection, and presents related observations as reviewable evidence. In addition to conventional alerts, the project provides read-only process monitoring, domain-only browser activity, and a timeline that links browser downloads to subsequent process activity when identity and timing evidence support the association.The system emphasizes evidence quality and privacy. It avoids treating a process identifier alone as proof of process identity, distinguishes observed absence from a verified process exit, redacts sensitive command-line values, and reports uncertainty when attribution cannot be established. The contribution is an integrated, educational workflow that combines multiple telemetry sources with explainable correlation and analyst review.

II. RELATED WORK

Security information and event management approaches aggregate events from multiple sources to support detection and incident investigation. NIST guidance on log management highlights the importance of establishing logging infrastructure, retaining useful records, and using logs during operational response

[1]. Windows security auditing provides event records for activities such as logons, object access, and process creation; Windows Event Forwarding can centralize selected events from managed endpoints

[2]. These capabilities provide the foundation for the Windows-focused collection pipeline in AD Monitor. Rule-based detection is useful when a behavior can be described with a clear condition and threshold. Examples include repeated access to sensitive files, access to a decoy document, suspicious download-to-execution

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511189

sequences, and large volumes of file modifications. Such rules are interpretable, but their coverage depends on the available telemetry and configured policy. Anomaly detection can complement rules by identifying observations that differ from a learned baseline. Isolation Forest, introduced by Liu, Ting, and Zhou, isolates anomalous samples using random partitioning and is designed for anomaly detection

[3]. A model does not independently establish malicious intent, so AD Monitor retains deterministic detections and explicit baseline readiness states alongside UBA. Incident-response and adversary-behavior knowledge bases, including MITRE ATT&CK, provide structured descriptions of tactics and techniques that can help analysts interpret observations

[4]. AD Monitor includes an educational attack library and investigation interface; these components are intended to support learning and triage, not to claim that every technique is detectable from the available events. The distinguishing emphasis of this project is the integration of Windows event ingestion, endpoint and browser telemetry, privacy controls, explainable evidence correlation, and a classroom-oriented investigation workflow in a resource-constrained lab.

III. PROPOSED SYSTEM

AD Monitor is a full-stack monitoring and investigation application for authorized lab endpoints joined to a Windows Active Directory domain. It has five functional areas: (i) event and telemetry collection, (ii) normalization and durable ingestion, (iii) detection and UBA, (iv) evidence correlation and case review, and (v) dashboards and educational presentation. The backend is implemented with FastAPI and uses SQLite in write-ahead logging (WAL) mode with database migrations. The frontend is a React application built with Vite and TypeScript.

The system receives Windows Security events through a Windows Event Forwarding bridge, including selected logon, object-access, share-access, process-creation, and Defender events. A PowerShell endpoint agent contributes health metrics, process snapshots, and lifecycle information. A Manifest V3 browser extension and registered Windows Native Messaging host contribute domain-level navigation and download lifecycle metadata. The backend stages incoming batches, validates collector identity and telemetry kind, normalizes records, evaluates detection rules, and stores reviewable findings.

AD Monitor is designed for monitoring and investigation, not active endpoint control. The process monitor is read-only and does not expose remote process termination controls. The application supports analyst review states and notes while preserving the system-generated correlation confidence. Cases can be exported as JSON or CSV for further examination.

IV. SYSTEM ARCHITECTURE

The AD Monitor system follows a layered architecture that supports centralized security monitoring and threat detection in Active Directory environments. Each layer performs a specific function, making the system easier to understand, maintain, and extend. The architecture integrates Windows systems, event collection, threat detection, risk classification, database management, and a web-based dashboard. Security events generated by the Active Directory server and connected Windows systems are collected and processed to identify potentially suspicious activities. The detected events and their associated risk levels are then presented to administrators through a centralized interface.

A. Data Sources (Windows Environment)

The Data Sources component collects information from three main sources: Active Directory security events, Windows endpoint systems, and browser activity. Active Directory provides login and logout records, user account activities, group policy changes, and privilege escalation events. Endpoint systems provide process creation and termination details, file and registry activities, and system events. Browser activity monitoring collects relevant information such as visited domains, browsing patterns, and download activities while applying privacy controls. These data sources provide the basic information required to identify potential insider threats.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511189

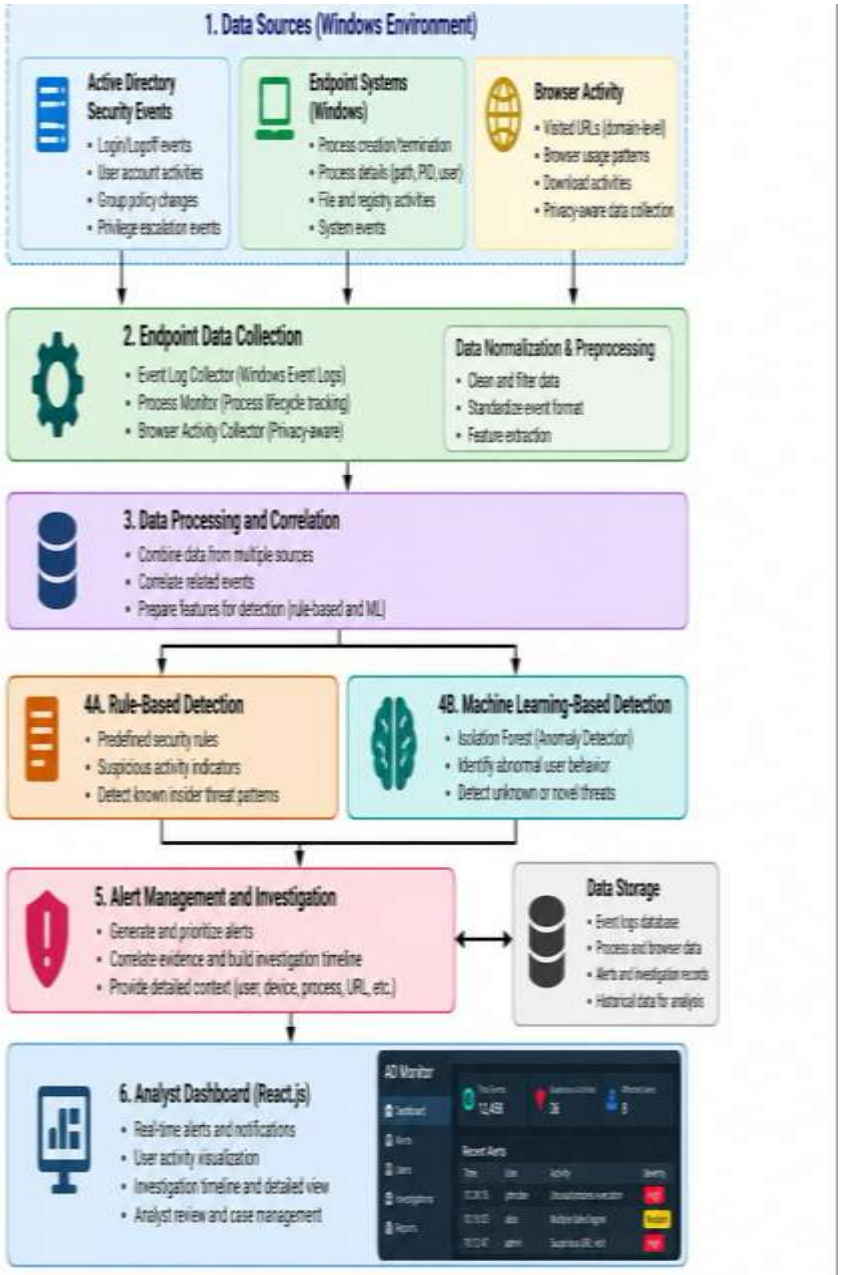


Figure 1: System Architecture of the Proposed AD Monitor System

B. Endpoint Data Collection

The Endpoint Data Collection component gathers information from the Windows environment using event log collectors, process monitors, and browser activity collectors. The event log collector gathers Windows event records, the process monitor tracks process activity, and the browser activity collector gathers relevant browsing information. The collected data then undergoes normalization and preprocessing, where irrelevant data is filtered, event formats are standardized, and useful features are extracted. This step prepares the data for further processing and analysis.

C. Data Processing and Correlation

The Data Processing and Correlation component combines the collected information from different sources and identifies relationships between related events. It processes the data, correlates activities, and prepares relevant features for both rule-based and machine-learning detection. For example, a login event followed by unusual process execution may be analyzed together to determine whether the activity requires further investigation. This component helps transform raw event records into meaningful information for threat detection.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511189

D. Detection and Analysis

The Detection and Analysis component contains two methods: Rule-Based Detection and Machine Learning-Based Detection. Rule-based detection uses predefined security rules and suspicious activity indicators to identify known insider-threat patterns. Machine-learning-based detection uses the Isolation Forest algorithm to identify unusual user behavior and anomalies that may not match predefined rules. The two methods complement each other by identifying different types of suspicious activities, and their results are forwarded to the alert management stage.

E. Alert Management and Investigation

The Alert Management and Investigation component processes the results produced by the detection methods. It generates alerts for suspicious activities, prioritizes them, and assigns severity levels. It also correlates evidence, builds investigation timelines, and provides relevant details such as user, device, process, and URL information where available. The connected data storage component maintains event logs, process and browser records, alerts, investigation records, and historical data. This information supports security analysts in investigating potential threats.

F. Analyst Dashboard (React.js)

The Analyst Dashboard provides a centralized interface for security analysts to monitor and investigate suspicious activities. Developed using React.js according to the diagram, it displays alerts and notifications, visualizes user activity, presents investigation timelines, and provides detailed views of relevant information. It also supports analyst review and case management. Through the dashboard, security analysts can examine alerts, review supporting evidence, and investigate potential insider threats more efficiently.

V. SYSTEM DESIGN AND METHODOLOGY

5.1 Telemetry collection and ingestion. Data is collected only from the desired data types; Collector types are registered and scoped so that endpoint and browser collectors only submit the data types they are supposed to submit. Versioned ingestion batches can be acknowledged, have a transport identifier, and are checkpointed and duplicate handled. Records are retained in the queue in case of temporary connections failures with the help of local spooling and retry behaviour. Windows Event Forwarding records are read using a persistent bookmark by the collector bridge; endpoint and browser components submit their own telemetry through the same ingestion boundary.

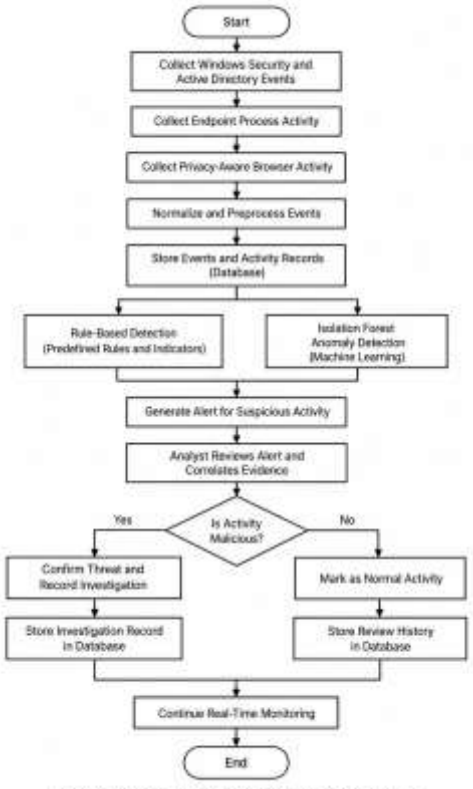


Figure 2. Workflow of the Proposed AD Monitor System

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511189

5.2 Process monitoring and lifecycle reconstruction. The endpoint agent also samples process metadata periodically, such as process identifier, process creation time, process owner, resource utilization, process command line strings to be redacted, process signature status, and cached hash (when available). CPU utilization is computed over a time period, not from the first observation. Process instances are associated based on order of creation time or process GUID evidence provided if available but a PID is not enough to establish identity. If a process is not observed if it is not seen to have terminated between the two snapshots, the interface displays the process as being “no longer observed” rather than an exact time of exit.

5.3 rivacy-aware browser telemetry. The browser extension captures top-level HTTP/HTTPS navigation, and shrinks URLs to domain-level information before persistence. Stored navigation telemetry excludes paths, query strings, fragments, credentials, titles, form inputs and cookies. The following are not included: Incognito/private browsing and pre-enrollment history import. Download observations are also shown as updates to a single logical download record, which ensure that multiple state changes do not count as multiple downloads. Collection gaps, such as pause, disconnection, and queue overflow, are represented explicitly.

5.4 The project introduces configurable deterministic rules for behaviors like restricted-path access denial, bulk-sensitive-file reads, canary-file accesses, after-hours-sensitive behavior, data staging, removable storage transfer evidence, downloaded script execution, downloaded volume, rapid file modification, non-primary-device access, and Windows Defender detections, among others. The project README provides a description of other process/browser/correlation rules. Rule execution will be based on the event types that are required and have been enabled and collected. The UBA component is based on Isolation Forest using CPU and temporal feature windows. When the system has a cold start, it reports “Collecting baseline”, and doesn't score users with less than 24 observation windows, deterministic rules are still in place.

5.5 Evidence correlation/analyst review. When other fields available, such as device SID, user SID, timing, are in agreement, and when domain visits, browser downloads, process creations, are in agreement, the investigation timeline is linked. Every association has a confidence label, and matched/missing fields, with the ability to reveal other candidates. The interface helps to differentiate between what is supported, and what it fails to prove. Activity can be identified by analysts as unreviewed, for followup, benign/expected, confirmed lab test, or escalated to an alert, and the analysts can enter review notes that do not replace the confidence machine-generated confidence.

VI. IMPLEMENTATION

The technologies that are being implemented are summarized in Table 1. The backend is served with API routes using FastAPI and validates ingestion contracts with Pydantic. Authentication features Argon2id password hashing, sessions, CSRF protections, and login-attempt auditing. Users, Sessions, Collectors, Devices, Employees, Events, Ingestion Inbox Records, Rule Definitions and Versions, Alerts, Evidence, Features, and Model Checkpoints are stored in the SQLite database. The frontend offers overview, entities, cases, administration, process monitoring, browser activity, investigation timeline, attack library and teacher presentation mode views.

The deployment is targeting a small Windows lab consisting of a host (backend and web interface), a domain-controller virtual machine (forwarded security events and Active Directory), and an enrolled employee workstation (endpoint collector and browser collector). The project file contains an example topology and setup scripts. The implementation deliberately omits to use a heavyweight distributed infrastructure, to maintain the app's feasibility on a small lab computer.

TABLE I TECHNOLOGIES IDENTIFIED IN THE SUPPLIED PROJECT REPOSITORY.

Component	Technology / approach
Backend API	Python, FastAPI, Uvicorn, Pydantic v2
Frontend	React, TypeScript, Vite
Persistence	SQLite with WAL mode and SQL migrations
Authentication	Argon2id password hashing, sessions, CSRF controls, login auditing
UBA	NumPy-based feature processing and Isolation Forest model
Endpoint collection	PowerShell agent; Windows Security and optional Sysmon lifecycle events
Browser collection	Manifest V3 extension and Python Native Messaging host
Testing	pytest, API/database/rules/UBA/integration tests

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511189

VII. TESTING AND OBSERVATIONS

The repository includes an automated pytest suite focusing on API behavior, database and migration behavior, normalization, deterministic rules, UBA, process/browser/timeline integration and some security/privacy guards. The implementation-status notes include information about the following tests: reversible migration, process snapshot CPU deltas, PID reuse, out-of-order lifecycle events, command-line redaction, URL sanitization, browser download revisions and collection gaps, native-host framing, allow-list checks, and investigation timeline correlation (including negative cases and analyst review).

These tests are implemented level tests which evidence that selected workflows and edge-cases are exercised. They aren't able to set production detection rates, false-positive rates, scalability for enterprise load, or effectiveness against real insider incidents. This is also mentioned in the documentation provided with the UBA, as in a lab setting with one or two employee VMs, the baseline is sparse and comprehensive individual baselines would need 14-30 days of active telemetry. It is therefore best judged as a working prototype for a laboratory system; any quantitative claims of performance would need to be made in a separately designed and reported experiment.

VIII. LIMITATIONS AND FUTURE WORK

Correctly setup Windows auditing, Event Forwarding, endpoint enrollment and browser-host registration are essential to the system. If a sensor is missing, a rule may not be able to fire, and if either of the identity and workstation fields are incomplete, the result may be a lack of clarity in who the client is. Inserting a USB device is not sufficient to be considered as a file transfer event, and an alert must be based on correlated evidence of a write to a destination.Even if a USB is inserted the event is not considered to be a file transfer and an alert must be based on correlated evidence of a write to a destination. Standards-based socket counters don't examine encrypted HTTPS data. SQLite WAL is appropriate for the intended workload at the lab level, but may need to be reconsidered for a multi-node enterprise deployment. The project is not a certified endpoint detection and response product and is not meant to be used as a sole solution for protection of production systems.

Possible future improvements include larger (ethically collected) evaluation dataset, precision, recall, alert volume, ingestion latency and resource usage measurement, longer term UBA baseline studies, and usability testing with students or analysts. A production oriented version might include a PostgreSQL deployment option, hardened transport and deployment guidance, more formal retention controls, and other validated telemetry integrations. These extensions should maintain existing focus on privacy, explainability and clear distinction between the facts observed and the relationships inferred.

IX. CONCLUSION

This paper introduced AD Monitor, a privacy protected endpoint monitoring and investigation workspace in Windows Active Directory lab environments. It combines the ability to process forwarded security events, endpoint process telemetry, domain-level browser activity, deterministic detection rules, an evidence-first investigation timeline, and an Isolation Forest UBA (user behavior analytics). Process identity protects, explicit collection gaps, privacy filtering and analyst review are methods to increase the interpretability of the presented evidence and to not overstate what a sensor observation proves. Automated tests of critical ingestion, detection, process, browser and correlation workflows are available in the repository. More quantitative assessment and extended baseline data collection are needed before claims can be made with respect to the effectiveness of detection in the field or enterprise readiness.

ACKNOWLEDGMENTS

Add the names of the project supervisor, department, institution, and other contributors, who should be acknowledged, here. Avoid keeping acknowledgements that are not true, or omitted when they are not applicable.

REFERENCES

[1]. Hsieh et al., “IAD2: Anomaly Detection on Active Directory Log Data for Insider Threat Monitoring,” in Proceedings of the International Carnahan Conference on Security Technology (ICCST), IEEE, 2015.
[2]. F. Liu, Y. Liu, O. De Vel, J. Zhang, and Y. Xiang, “Anomaly-Based Insider Threat Detection Using Deep Autoencoders,” in Proceedings of the IEEE International Conference on Data Mining Workshops (ICDMW), 2018.



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511189

[3]. M. Singh, B. M. Mehtre, and S. Sangeetha, "User Behavior Profiling Using Ensemble Approach for Insider Threat Detection," in Proceedings of the IEEE International Conference on Identity, Security, and Behavior Analysis (ISBA), 2019.
[4]. W. Huang, H. Zhu, C. Li, Q. Lv, Y. Wang, and H. Yang, "ITDBERT: Temporal-Semantic Representation for Insider Threat Detection," in Proceedings of the IEEE Symposium on Computers and Communications (ISCC), 2021.
[5]. R. Nasir, M. Afzal, R. Latif, and W. Iqbal, "Behavioral Based Insider Threat Detection Using Deep Learning," IEEE Access, 2021.
[6]. "User Behaviour Based Insider Threat Detection in Critical Infrastructures," in Proceedings of the 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC), IEEE, 2021.
[7]. T. Al-Shehari, M. S. Al-Razgan, T. Alfahik, R. A. Alsowail, and S. Pandiaraj, "Insider Threat Detection Model Using Anomaly-Based Isolation Forest Algorithm," IEEE Access, 2023.
[8]. U. Rauf, Z. Wei, and F. Mohsen, "Employee Watcher: A Machine Learning-Based Hybrid Insider Threat Detection Framework," in Proceedings of the 7th Cyber Security in Networking Conference (CSNet), IEEE, 2023.
[9]. Z. Q. Wang and A. El Saddik, "DTITD: An Intelligent Insider Threat Detection Framework Based on Digital Twin and Self-Attention Based Deep Learning Models," IEEE Access, 2024.
[10]. F. R. Alzaabi and A. Mehmood, "A Review of Recent Advances, Challenges, and Opportunities in Malicious Insider Threat Detection Using Machine Learning Methods," ACM Transactions on Information and System Security, publication details require verification.
[11]. O. Gonzales, K. Yang, and S. Huang, "Towards More Effective Insider Threat Countermeasures: A Survey of Approaches for Addressing Challenges and Limitation," in Proceedings of the IEEE International Systems Conference (SysCon), 2024.
[12]. Deltrux Research Team, "Deltrux: Insider Threat Detection of End-User Behavior Analysis Using Machine Learning," in Proceedings of the 3rd International Conference for Advancement in Technology (ICONAT), IEEE, 2024.
[13]. N. Dixit, R. Gupta, and P. Yadav, "User Behavior Analysis to Detect Insider Threat by Using Machine Learning Algorithms," in Proceedings of the 4th International Conference on Innovative Practices in Technology and Management (ICIPTM), IEEE, 2024.
[14]. M. A., M. Puteh, and S. R., "Insider Threat Detection Using Machine Learning Models for User Behavior Analysis," in Proceedings of the 5th International Conference on Expert Clouds and Applications (ICOECA), IEEE, 2025.
[15]. S. A. Laczi and V. Pósér, "Behavioural Analysis in Human-Machine Interaction for Insider Threats," in Proceedings of the IEEE 29th International Conference on Intelligent Engineering Systems (INES), 2025.

 This work is licensed under a Creative Commons Attribution 4.0 International License