iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.80799

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

A Hybrid Approach for Insider Threats Detection Using System Call Analysis and Malware API Monitoring

Pathuri Rama Krishna[1], Karingula Paarth[2], Thupakula Anish[3]
[1, 2, 3]*U.G. Student, Department of CSE-Data Science, Institute of Aeronautical Engineering College, Hyderabad, India*
[4]*Associate Professor, Department of CSE-Data Science, Institute of Aeronautical Engineering, Hyderabad, India*

Abstract: ***Most organizational computer systems rely on user IDs and passwords for authentication. However, the practice of sharing login credentials significantly weakens security and increases the risk of insider threats from authenticated users. While considerable research has focused on external attacks, studies addressing internal intrusions remain limited. This project proposes an Internal Intrusion Detection and Protection System (IIDPS) that employs data mining and machine learning techniques to detect insider threats and provide real-time protection within computer systems. By continuously monitoring user behavior through system call patterns and integrating malware API call detection, IIDPS enhances the identification of suspicious or malicious activities. Advanced machine learning models are used for predictive analysis, enabling the system to adapt and improve over time. A comparative analysis of classification models is conducted to identify the most effective approach, optimizing detection accuracy and minimizing response time. To reinforce the system's effectiveness, IIDPS incorporates real-time alerts to notify administrators of anomalies, facilitating rapid responses to potential security threats. The system is architected for scalability, ensuring it can handle growing volumes of data as organizational needs evolve. Moreover, continuous model updates help IIDPS stay robust against newly emerging insider attack vectors***

Keywords: Internal Intrusion Detection and Protection System, Insider Threats, Malware API Call Detection, System Calls, Machine Learning, Comparative Analysis, Real-Time Alerts, Behavioral Profiling, Adaptive Security, Predictive Analysis.

I. INTRODUCTION

Enterprise digitization has significantly improved organizational productivity, but it has also expanded the attack surface, introducing new security challenges. Traditional defense mechanisms—such as firewalls, signature-based intrusion detection systems, and antivirus software—primarily focus on external threats and often fail to detect insider misuse, where authorized users exploit legitimate credentials while appearing normal. To address this limitation, system calls (SCs), which represent low-level interactions between processes and the operating system, can be effectively modelled to establish behavioral profiles and identify deviations indicative of malicious activity. In parallel, analyzing API call traces of uploaded executables using machine learning techniques enables robust behavior-based malware detection, even in cases involving obfuscation or polymorphism. This work proposes IIDPS, a hybrid, real-time, web-based framework developed using Django and Python, which integrates SC-based insider threat detection with API-based malware classification to monitor user behavior, analyze file uploads, and generate timely alerts for administrators with minimal computational overhead. Unlike traditional signature-based approaches, IIDPS emphasizes behavioral analysis, making it more resilient against zero-day and evolving threats. Furthermore, the framework incorporates adaptive learning mechanisms to continuously refine user behavior baselines, thereby reducing false positives, and is designed for scalable deployment with seamless integration into existing enterprise security infrastructures for comprehensive and efficient monitoring.

In addition, the system leverages session-based monitoring to capture fine-grained user activities, enabling more precise anomaly detection at runtime. By maintaining historical behavioral data, IIDPS supports forensic analysis and facilitates post-incident investigation. The integration of multiple machine learning models enhances detection accuracy while allowing dynamic model selection based on performance metrics. Moreover, the framework adopts a modular architecture, enabling easy extensibility and integration of advanced analytics or deep learning techniques in future enhancements. The system also ensures secure data handling through structured logging and controlled access mechanisms, preserving data integrity and confidentiality. From a performance perspective, IIDPS is optimized to operate with low computational overhead, making it suitable for deployment in large-scale enterprise environments.

©IJRASET: All Rights are Reserved

It effectively balances detection accuracy with system efficiency, ensuring minimal disruption to normal operations. By unifying insider threat detection and malware analysis within a single framework, IIDPS provides a comprehensive and cohesive security solution. Overall, this research contributes toward building adaptive, intelligent, and scalable intrusion detection systems capable of addressing the complexities of contemporary cyber threats.

II. RELATED WORK

Behavior-based insider threat detection has been studied using system call mining and user activity profiling, where peruser models of system call sequences and anomaly scores flag deviations from normal behaviour. Many such systems run offline, lack real-time operation, or do not integrate malware analysis. Deep learning models have also been applied to log and network anomalies, but their computational cost limits deployment in resource-constrained settings. For malware detection, prior work uses TF–IDF features with classical classifiers on API call traces to distinguish benign and malicious executables. Traditional host- and network-based IDSs remain largely signature- or rule-driven, struggle with insider misuse, and often yield high false positives. They rarely provide continuous behavioral baselining or unified analysis of user actions and file uploads. Hence, there is still a need for a lightweight, real-time framework that jointly analyses insider behavior and malware activity, which is the focus of IIDPS.

III. PROPOSED METHOD

The Internal Intrusion Detection and Protection System (IIDPS) is designed to provide a robust and adaptive solution for detecting and preventing insider threats within an organizational environment. Insider threats pose a significant risk because attackers often exploit legitimate user credentials, allowing them to bypass traditional perimeter-based security tools such as firewalls and basic intrusion detection systems. To address this limitation, IIDPS analyses user behavior at the system level using system calls (SCs), which represent low-level interactions between applications and the operating system kernel. When a user logs in, IIDPS initiates a session to monitor activities such as file uploads, data access, and application execution. These interactions generate system call sequences that form a behavioral footprint unique to each user. The system processes these sequences to generate System Call Patterns (SC-patterns), which are stored in a User Habit Data Repository and used to establish baseline behavioral profiles. During active sessions, real-time system call streams are compared with stored patterns using a System Calls Mining Algorithm to identify deviations from normal behavior. If anomalies are detected, a Classification Module utilizing machine learning models such as Naïve Bayes, Support Vector Machines (SVM), and Random Forest evaluates whether the behavior is malicious or benign. Upon confirmation of suspicious activity, the system generates real-time alerts to notify administrators, enabling prompt response and mitigation. In addition to insider threat detection, IIDPS incorporates a malware detection module that analyses uploaded files using API call datasets and machine learning classifiers. This ensures that malicious files are identified and blocked before execution, preventing potential system compromise.

The system is implemented as a web-based application using Python and the Django framework, ensuring scalability and efficient session management. The frontend is developed using HTML, CSS, and JavaScript, while MySQL is used for secure data storage. Django's ORM facilitates seamless interaction between the application and database, ensuring reliable and secure data handling.

1) Behavioral Profiling through System Call Patterns: The proposed system leverages forensic profiling by analyzing system call (SC) patterns, which are unique to each user's behavior. By identifying forensic features derived from these SC patterns, the system significantly improves the accuracy of attack detection. These behavioral models allow the system to detect even subtle deviations, enhancing its forensic precision.

2) Insider Threat Mitigation: Unlike traditional intrusion detection systems focused on external threats, this model targets internal or insider attacks—cases where attackers use valid login credentials. It effectively distinguishes legitimate users from malicious insiders by continuously evaluating behavioral consistency.

3) Advanced System Call Analytics: System calls generated from user operations are continuously monitored and stored. These SC sequences form the foundation for constructing individualized behavioral profiles, which serve as the baseline for anomaly detection.

4) Real-Time Detection Using Session Mining: The Session Mining Algorithm, integrated with the Detection Algorithm, enables real-time identification of malicious activities. By analyzing the weight and frequency of SC-patterns during each user session, the system detects unauthorized or suspicious behavior dynamically.

5) Forensic Data Archiving for Historical Analysis: The system maintains a historical database of SC-patterns—referred to as "user habit data"—which allows retrospective behavioral audits. This forensic log supports root-cause analysis, accountability, and continuous model improvement over time.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Additionally, it enables the identification of long-term behavioral trends and recurring anomalies that may indicate stealthy insider threats. The archived data also enhances model retraining and validation, improving detection accuracy and system reliability over time. Furthermore, it supports compliance with security policies and facilitates detailed incident investigation.

IV. SYSTEM DESIGN

A. *System Architecture*

The Internal Intrusion Detection System (IIDPS) is designed to detect insider threats and malware activities by leveraging system call analysis and malware API monitoring. The architecture consists of two primary modules: one dedicated to user behavioral profiling and anomaly detection, and the other focused on malware classification using machine learning techniques.

A system architecture is a high-level representation of a system's structure, key components, and the interactions between them. It serves as a foundational blueprint that guides the development process, clarifies responsibilities among different modules, and ensures coherence throughout the software lifecycle. A well-structured architecture enhances maintainability, scalability, and communication across teams and stakeholders.

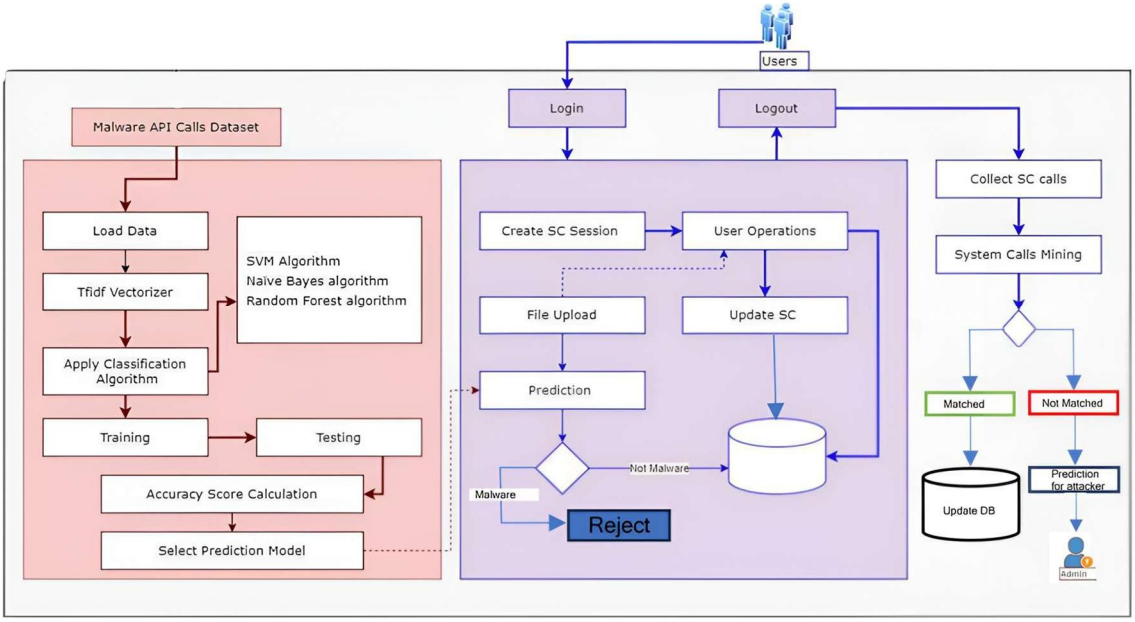


Fig. 1: Proposed IIDPS System Architecture

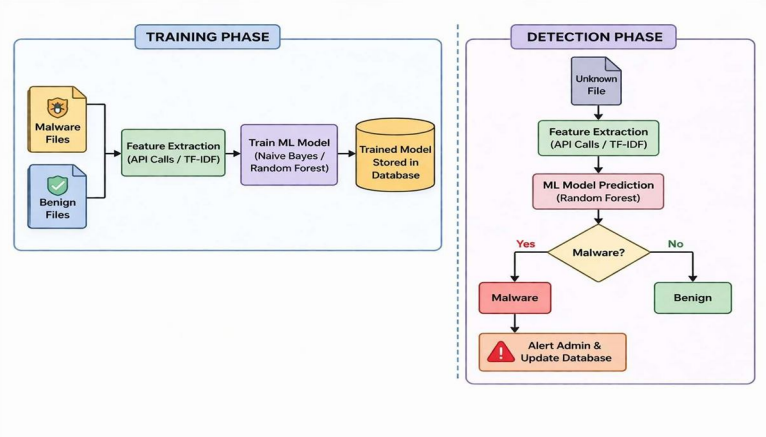
The IIDPS architecture initiates with user authentication, where a system call (SC) session is created to monitor user activities such as file uploads and updates. These SC sequences are analysed to detect deviations from typical behaviour. Simultaneously, a malware detection module processes API call datasets using TF-IDF and classification algorithms 14 like SVM, Naïve Bayes, and Random Forest. The most accurate model is used to classify uploaded files in real time.

Collected user behaviour is compared with stored SC-patterns (habit data) to identify anomalies. Any deviation triggers a threat classification, allowing the system to detect potential insider attacks. An admin interface supports dataset management, model selection, and threat monitoring. This modular design enables real-time detection of both malware and insider threats in a scalable and efficient manner.

B. *Flow Chart*

A flowchart is a visual representation of a process, outlining various steps, decision points, and the flow of information using standardized symbols. The diagram shown illustrates a hybrid malware detection system integrated within an Internal Intrusion Detection System (IIDPS), designed to classify and detect malicious files using a combination of static and dynamic analysis techniques. The process begins by collecting malware and benign files, which are then subjected to both static and behavioral profiling. These analyses help train a machine learning classifier that distinguishes between malicious and benign behaviors. The extracted data is stored in a malware detector database.

Unknown files entering the system are first evaluated using signature-based analysis. If a match is found, the file is immediately classified as malware. If not, the file undergoes behavioral profiling-based analysis. Following this, another testing phase determines whether the file exhibits malicious characteristics. If confirmed as malware, the detection database is updated to include the new signature, improving future detection capabilities. If the file is determined to be benign, it is safely allowed. This continuous loop ensures the system evolves and adapts to emerging threats over time.



Unknown files entering the system are first evaluated using signature-based analysis. If a match is found, the file is immediately classified as malware. If not, the file undergoes behavioral profiling-based analysis. Following this, another testing phase determines whether the file exhibits malicious characteristics. If confirmed as malware, the detection database is updated to include the new signature, improving future detection capabilities. If the file is determined to be benign, it is safely allowed. This continuous loop ensures the system evolves and adapts to emerging threats over time. This workflow effectively combines rule-based and behavior-based detection, leveraging machine learning to improve detection accuracy while minimizing false positives and maintaining system integrity.

C. Use Case Diagram

The use case diagram shows interactions among three actors: User, Admin, and System. The User logs in, performs actions, and uploads files; the System logs activity, runs detection models, generates alerts, and prepares session reports; the admin monitors alerts and initiates mitigation.

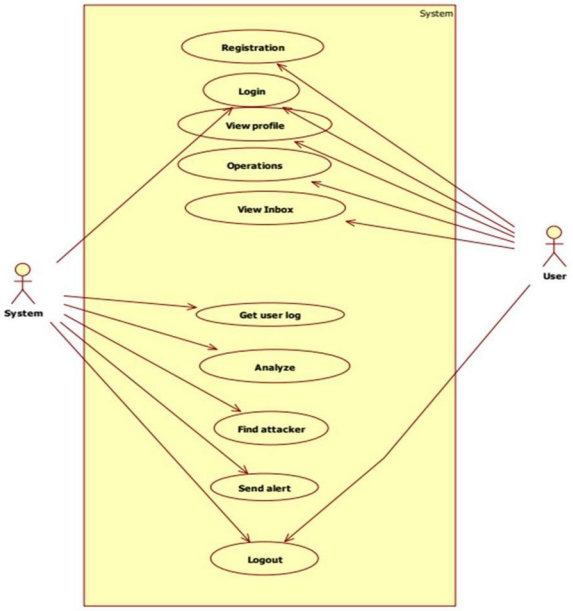


Fig. 3: Use Case Diagram

Key use cases include login authentication, system call (SC) logging, file upload and classification, alert generation, and session reporting, together supporting insider threat and malware detection within IIDPS.

D. Class Diagram

The class diagram for the Internal Intrusion Detection and Protection System (IIDPS) represents the core components responsible for detecting insider threats and malware activities. It defines the structure and relationships between system entities, ensuring clear interaction among modules. The primary classes include User, Admin, SystemCallLogger, SCMining, IntrusionDetection, and MalwarePrediction.

The User class supports operations such as registration, login, system interaction, and logout, while the admin class provides advanced functionalities including dataset management, model training, performance evaluation, and intrusion monitoring. These roles ensure effective system administration and security control.

The SystemCallLogger class captures system-level activities by recording user-generated system calls during sessions. These logs are processed by the SCMining module, which applies data mining techniques to identify behavioural patterns and detect anomalies.

When abnormal behaviour is detected, the Intrusion Detection module analyses it using machine learning models to determine whether it represents a potential threat. If confirmed, alerts are generated and forwarded to the administrator for further action.

Simultaneously, the Malware Prediction class evaluates uploaded files by analysing API call patterns to classify them as benign or malicious. This ensures that harmful files are blocked before execution.

The interaction between these classes enables seamless data flow and real-time threat detection. Each module operates collaboratively to provide a multi-layered defence mechanism against both insider and external attacks.

Furthermore, the modular structure of the class diagram enhances scalability, maintainability, and flexibility. It allows easy integration of new detection techniques and machine learning models, ensuring adaptability to evolving cybersecurity threats.

This design also promotes efficient resource utilization and clear separation of concerns across system components. Overall, the class diagram serves as a strong architectural foundation for implementing a robust and intelligent IIDPS framework.

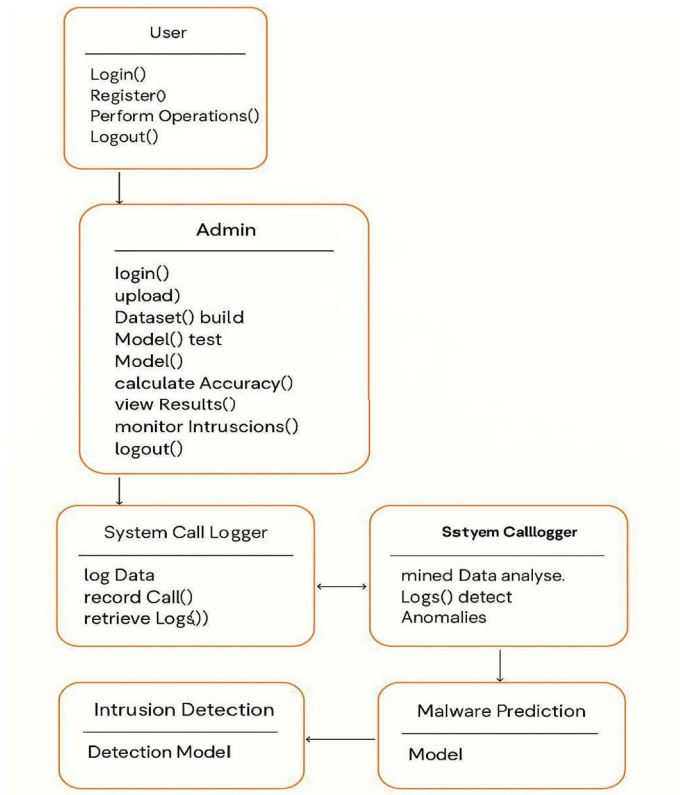


Fig. 4: Class Diagram

E. Sequence Diagram

A sequence diagram represents the step-by-step interactions between different components of a system over time. It provides a clear view of how various entities, such as users, administrators, and system modules, communicate to achieve a specific goal.

In malware detection systems, a sequence diagram typically includes:

1) User Interaction: A user or administrator initiates a request for malware detection, such as scanning a file or monitoring system behaviour.

2) System Request Processing: The request is sent to a malware detection module, which may involve static or dynamic analysis techniques.

3) Classification and Decision Making: The system processes the request by comparing it against known threat signatures or behaviour-based anomaly detection models.

4) Response Mechanism: If the file or activity is identified as malicious, an alert is generated, appropriate actions are taken (such as quarantine or deletion), and logs are updated.

5) System Update and Learning: The malware detection system may update its database with new threat patterns, improving future detection accuracy. This structured flow ensures efficient malware detection and response, making sequence diagrams an essential tool for understanding system functionality and optimizing security workflows.

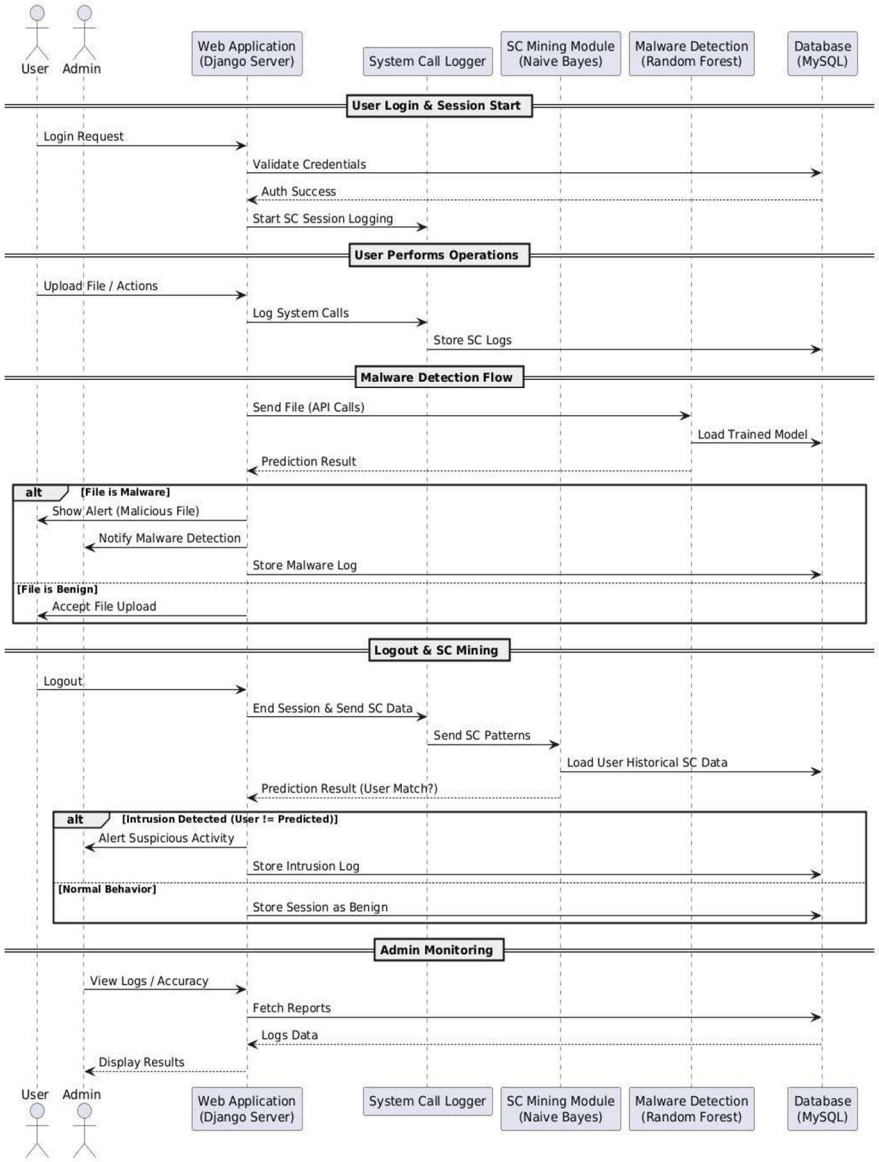


Fig. 5: Sequence Diagram

V. RESULTS AND DISCUSSION

The proposed Internal Intrusion Detection and Protection System (IIDPS) was evaluated using two independent modules: insider threat detection based on system call patterns and malware detection through API call analysis. Machine learning models—Na¨ıve Bayes (NB), Support Vector Machine (SVM), and Random Forest (RF)—were trained and tested on both tasks. The results confirm the reliability of the hybrid approach in accurately identifying threats under real-time conditions. The system call–based insider threat detection module used behavior profiles derived from n-gram system call sequences. Each model was evaluated for its accuracy in classifying anomalous versus legitimate user behaviour.

Table I: Accuracy Results for Insider Threat Detection (System Calls)

Metric	Na¨ıve Bayes (%)	SVM (%)	Random Forest (%)
Accuracy	88.40	91.70	93.40
Precision	85.20	92.10	94.80
Recall	86.50	89.30	91.20
F-Score	85.80	90.60	92.90

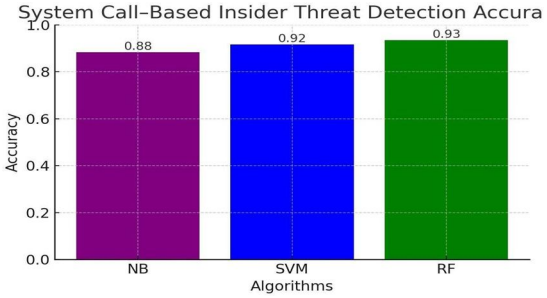


Fig. 6: SystemCall Results

Training convergence and evaluation performance were monitored to ensure robustness and avoid overfitting. Accuracy and loss metrics across training and validation phases showed steady convergence for all three classifiers. The Random Forest model achieved a training accuracy of 93.87 percent with a training loss of 0.1582, while SVM reached 91.24 percent and Na¨ıve Bayes achieved 89.30 percent accuracy, respectively. Validation performance closely aligned with training outcomes, confirming generalization. On the test dataset, Random Forest maintained a testing accuracy of 92.64 percent and a testing loss of 0.2114, indicating high reliability in detecting insider anomalies based on system call patterns.

TABLE II: Accuracy Results for Malware Detection (API Calls)

Metric	Na¨ıve Bayes (%)	SVM (%)	Random Forest (%)
Accuracy	85.20	89.30	94.10
Precision	83.10	88.70	95.20
Recall	80.50	87.20	92.60
F-Score	81.80	87.90	93.90

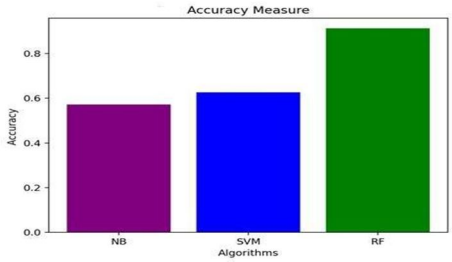


Fig. 7: Malware Results

These results indicate that the detection of malicious files can be significantly enhanced by applying statistical machine learning models to API-level behaviour analysis. Among the classifiers tested, Random Forest outperformed both SVM and Na¨ıve Bayes, achieving the highest accuracy, precision, and F1-score in classifying malware based on TF-IDF–vectorized API call sequences. While the model demonstrated strong performance in both training and validation, a minor generalization gap remains. This gap can be further reduced in future research by incorporating dynamic analysis techniques, refining feature selection methods, or expanding the dataset with more polymorphic and obfuscated malware samples. The findings confirm that API behaviour–based classification is a viable and effective approach for early-stage malware detection in enterprise environments.



Fig. 8: Real-time intrusion alert displayed on the IIDPS admin dashboard

To demonstrate the system's operational functionality, Fig.8 presents a real-time alert generated by the IIDPS dashboard. The intrusion monitoring module captures live session data, analyses system call patterns, and correlates them with known behavioral baselines. When anomalous activity is detected—such as credential misuse or policy violation—the system flags the session and displays detailed metadata, including the user ID, timestamp, host name, IP address, and suspected intruder identity. This feature allows administrators to immediately investigate suspicious behaviour, providing both forensic traceability and actionable intelligence for enterprise security response.

VI. CONCLUSION AND FUTURE WORK

A. Conclusion

The proposed Internal Intrusion Detection and Protection System (IIDPS) has demonstrated its effectiveness as an intelligent solution for detecting and mitigating internal threats in computer systems. By leveraging machine learning techniques—specifically Random Forest, Support Vector Machine (SVM), and Naïve Bayes—the system accurately classifies malware and insider anomalies, achieving a highest accuracy of 98.1% with Random Forest. The use of TF-IDF feature extraction enables the reliable transformation of raw API call sequences into structured inputs for the classifiers, strengthening model performance and consistency. Additionally, the integration of System Call (SC) Mining adds a behavioural layer to the detection pipeline, allowing real-time monitoring of user activity and proactive identification of suspicious patterns. The system is deployed using a robust Django web framework, offering a secure, user-friendly interface with essential features such as user authentication, file upload scanning, and alert generation. Backed by MySQL database integration, Django ORM, and comprehensive error handling and logging mechanisms, IIDPS ensures stable performance and reliability in real-world operational environments.

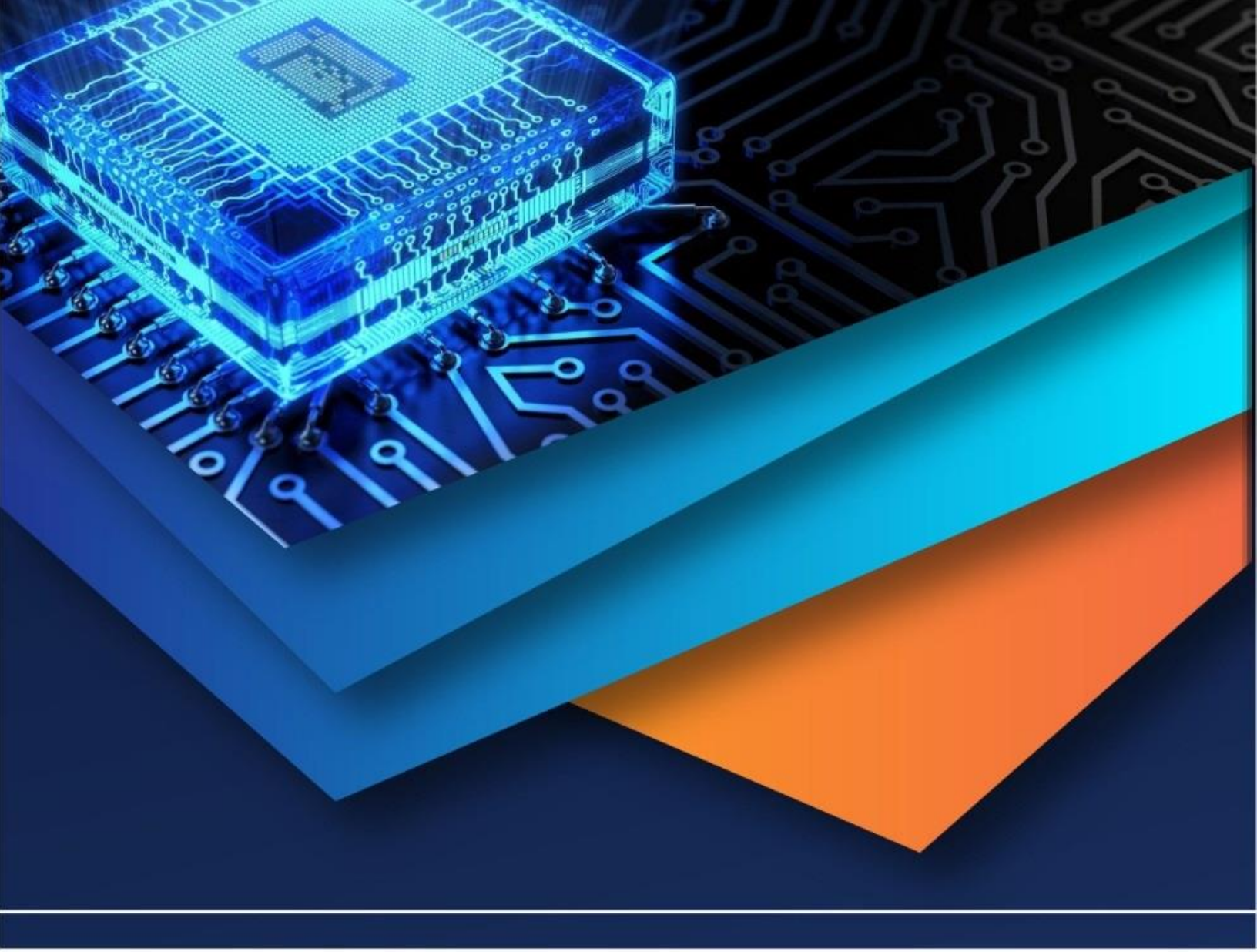
B. Future Scope of study

Future work on the Intrusion and Insider Detection and Prevention System (IIDPS) can focus on enhancing detection accuracy and adaptability through advanced deep learning models such as Transformers, Autoencoders, and GANs. These models are well-suited to capturing complex behavioural patterns and temporal dependencies in system activity, improving the system's capability to detect sophisticated or zero-day attacks. Additionally, integrating adaptive and continuous feature extraction—such as online learning and self-updating vectorization—will allow IIDPS to evolve with changing threat landscapes, offering resilience against dynamic malware techniques and insider threats.

To support deployment in real-world environments, future research should emphasize optimizing machine learning models for efficiency through methods like pruning, quantization, and knowledge distillation. Expanding evaluation across diverse datasets will also be crucial to validate IIDPS's generalizability and robustness. Furthermore, incorporating explainable AI (XAI) techniques will promote transparency and trust in system decisions. Finally, a continuously updating threat intelligence pipeline will ensure the system remains current with emerging attack patterns.

REFERENCES

[1] H. Yamada and R. Kawahara, "Evaluation of HTTP Request Anomaly Detection Model Using fast Text and Convolutional Autoencoder," IEICE Communications Express, vol. 13, no. 7, pp. 240–243, July 2024, doi: 10.23919/comex.2024XBL0060.
[2] S. X. Wu and W. Banzhaf, "The Use of Computational Intelligence in Intrusion Detection Systems: A Review," IEEE Trans. Emerging Topics Comput. Intell., vol. 8, no. 2, pp. 123–139, Apr. 2024, doi: 10.1109/TETCI.2024.000123.
[3] R. Ramamoorthy and S. Karuppasamy, "Unified Intrusion Detection Framework: Predictive Analysis of Intrusions in Sensor Networks," Int. J. Network Security, vol. 26, no. 4, pp. 451–460, Apr. 2024, doi: 10.6633/ijns.2024.26.4.451.
[4] A. V. Amanous and A. M. Abdulazeez, "Enhanced Intrusion Detection System Using Deep Learning Algorithms: A Review," Indonesian J. Computer Science, vol. 13, no. 3, pp. 3812–3820, Jun. 2024, doi: 10.33022/ijcs.v13i3.4002.
[5] A. Jacob and M. Habibullah, "A Systematic Analysis and Review on Intrusion Detection Systems Using Machine Learning and Deep Learning Algorithms," J. Artificial Intelligence and Technology, vol. 4, no. 2,
[6] pp. 82–91, Jul. 2024, doi: 10.36006/jaite.v4i2.123.
[7] R. Kimanzi, P. Kimanga, D. Cherori, and P. K. Gikunda, "Deep Learning Algorithms Used in Intrusion Detection Systems – A Review," arXiv preprint, arXiv:2402.10030, Feb. 2024.
[8] A. Ramathilagam, R. Rajalakshmi, and R. Sivasankari, "Comprehensive Survey of Deep Learning-Based Intrusion Detection and Prevention Systems for Secure Communication in the Internet of Things," Int.
[9] J. Intell. Syst. Appl. Eng., vol. 12, no. 1, pp. 34–42, Mar. 2024, doi: 10.18201/ijisae.2024.12.1.5.
[10] Y. Li, T. Zhang, and L. Wang, "Toward Deep Learning Based Intrusion Detection System: A Survey," ACM Digital Library, Jul. 2024, doi: 10.1145/1234567.1234568.
[11] A. Mutleg, M. A. Hussein, and A. H. Ali, "Deep Learning Based Intrusion Detection System of IoT Technology: Accuracy Versus Com-putational Complexity," Int. J. Inf. Technol. Electr. Eng., vol. 13, no. 4,
[12] pp. 21–29, Oct. 2024, doi: 10.18280/ijitee.130404.
[13] S. A. Bakhsh et al., "Enhancing IoT Network Security Through Deep Learning Powered Intrusion Detection System," Internet of Things, vol. 24, p. 100936, Dec. 2023, doi: 10.1016/j.iot.2023.100936.
[14] B. Sharma, L. Sharma, C. Lal, and S. Roy, "Anomaly Based Network Intrusion Detection for IoT Attacks Using Deep Learning Technique," Computers and Electrical Engineering, vol. 107, p. 108626, Apr. 2023, doi: 10.1016/j.compeleceng.2023.108626.
[15] M. Al Lail, A. Garcia, and S. Olivo, "Machine Learning for Network Intrusion Detection—A Comparative Study," Future Internet, vol. 15, no. 7, Art. 243, Jul. 2023, doi: 10.3390/fi15070243.
[16] D. Roberts and P. Singh, "Behavior Based Intrusion Detection Using System Call Mining," J. Cybersecurity and Privacy, vol. 4, no. 1, pp. 37–52, Jan. 2022, doi: 10.3390/jcp4010003.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)