Int. J. Sci. R. Tech., 2026 3(5)
A Multidisciplinary peer-reviewed Journal
www.ijsrtjournal.com [ISSN: 2394-7063]

Original Article

Forensic Analysis Of AI-Driven Cyber Deception Networks

Femina S.*[1], Sankara Narayanan S. T.[2]

[1]*Department of Cyber security, Dr MGR Educational and Research Institute, Chennai – 600095, India*
[2]*Department of Cyber Forensics and Information Security, Centre of Excellence in Digital Forensics*

ABSTRACT

The increasing sophistication of cyber-attacks demands intelligent systems capable of detecting and classifying attacker behaviour in real time. Traditional honeypot frameworks primarily focus on passive data collection and post-incident analysis, limiting their effectiveness in proactive threat response. This paper presents an AI-enhanced honeypot framework that integrates a Cowrie-based honeypot with a Random Forest machine learning model to enable real-time attack detection and classification. The system is deployed in a virtualized environment consisting of an Ubuntu honeypot and a Kali Linux attacker machine connected through a host-only network. A dataset of 5,740 log entries was generated through simulated attacks, including brute force, port scanning, reconnaissance, post-exploitation, and vulnerability scanning. The trained model achieved near-perfect classification performance across all attack categories under controlled experimental conditions. A live detection module continuously processes incoming logs and displays results through a Flask-based web dashboard with real-time alerts and visualization. The proposed system bridges the gap between honeypot data collection and intelligent analysis, offering a lightweight and effective solution for proactive cybersecurity defence.

Keywords: honeypot, cybersecurity, machine learning, random forest, intrusion detection, real-time monitoring.

INTRODUCTION

Background

The rapid digital transformation of modern society has significantly increased dependence on networked systems for communication, commerce, and critical infrastructure. As a result, cybersecurity has become a paramount concern globally. Cyber-attacks have evolved into highly sophisticated and coordinated operations carried out by organized criminal groups and nation-state actors, targeting everything from financial institutions to healthcare systems and government networks [1].

Traditional security mechanisms such as firewalls and signature-based intrusion detection systems (IDS) rely heavily on predefined rules and known attack signatures. While effective against established threats, these systems struggle to detect zero-day attacks and evolving adversarial techniques, creating critical gaps in modern cyber defence. The need for intelligent, adaptive systems capable of learning from attacker behaviour has therefore become increasingly urgent [2].

Honeypot Technology

Honeypots are decoy systems deliberately designed to attract attackers and record their activities within a controlled, isolated environment. Unlike traditional security tools that defend production systems, honeypots provide deep insights into attacker behaviour, tactics, techniques, and procedures (TTPs) without endangering real assets [3].

Cowrie, a widely adopted open-source honeypot, simulates SSH and Telnet services and logs detailed attacker interactions, including login attempts, executed commands, downloaded files, and session durations. Despite its comprehensive logging capability, most honeypot deployments rely on manual, post-attack log analysis, which significantly limits their usefulness in real-time threat detection and response scenarios [4].

Machine Learning in Cybersecurity

Machine learning has emerged as a transformative approach for detecting and classifying cyber threats at scale. Ensemble algorithms such as Random Forest

Relevant conflicts of interest/financial disclosures: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

can identify complex patterns in large, high-dimensional datasets and classify attack types with high accuracy and low false-positive rates. Honeypot logs, being structured and inherently labelled, provide an ideal training corpus for supervised learning approaches [2]. Integrating machine learning with honeypot infrastructure enables automated, continuous classification of attacker behaviour, significantly reducing response time and analyst workload [5].

PROBLEM STATEMENT

Despite considerable advances in both honeypot technology and machine learning, most existing cybersecurity systems treat data collection and threat analysis as entirely separate processes. Honeypots generate large volumes of raw log data, but the absence of real-time analytical pipelines results in significant delays between attack occurrence and detection. This lag undermines the fundamental purpose of proactive defence and increases the window of vulnerability for network administrators.

Furthermore, the reliance on manual log inspection is both time-consuming and error-prone, particularly as attack volumes and sophistication continue to rise. There is a clear and pressing need for an integrated, end-to-end system that seamlessly combines honeypot-based log collection with real-time machine learning classification and an intuitive visualization interface, enabling security teams to detect, categorize, and respond to threats as they occur [1][3].

PROPOSED SYSTEM

System Overview

The proposed AI-enhanced honeypot framework is built around three tightly integrated components that collectively deliver a complete attack detection pipeline: a Cowrie honeypot for data collection, a Random Forest machine learning model for attack classification, and a Flask-based web dashboard for real-time visualization and alerting. The architecture is designed to be lightweight, modular, and deployable in virtualized environments without requiring specialized hardware.

System Architecture

The end-to-end workflow of the proposed system is as follows:

1. An attacker (or simulated attack tool) interacts with the Cowrie honeypot exposed on the network.

2. Cowrie logs every interaction in structured JSON format, capturing IP addresses, session identifiers, executed commands, and timestamps.

3. A log parser extracts and transforms raw entries into numerical feature vectors suitable for model inference.

4. The pre-trained Random Forest classifier processes each feature vector and assigns an attack category label in real time.

5. Classification results are streamed to the Flask web dashboard, where they are displayed alongside visual analytics and instant alert notifications.

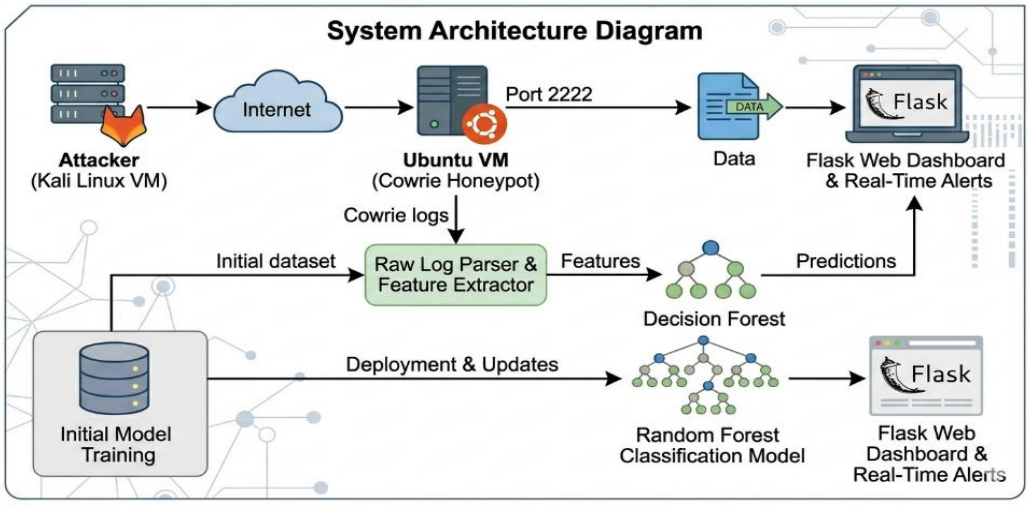

Figure 1: system architecture diagram

MATERIALS AND METHODS

Experimental Setup

The experimental environment comprised two virtual machines configured on a host-only network to ensure complete isolation from production infrastructure. The Ubuntu virtual machine served as the honeypot host, running the Cowrie SSH/Telnet honeypot service on port 2222. The Kali Linux virtual machine was designated as the attacker machine, from which all simulated attack scenarios were executed. This controlled topology ensured reproducible conditions for dataset generation and system evaluation.

Data Collection

A labelled dataset of 5,740 log entries was generated through systematic simulated attack campaigns. Five distinct attack categories were represented in the dataset to ensure class diversity and model generalizability:

- Brute Force Attacks – Repeated SSH login attempts using credential wordlists.
- Port Scanning – Automated enumeration of open ports and services using Nmap.
- Reconnaissance – System fingerprinting and environment enumeration commands.
- Post-Exploitation – Commands executed after successful authentication, including file access and lateral movement simulation.
- Vulnerability Scanning – Automated probing for known service vulnerabilities.

Data Pre-processing

Raw Cowrie JSON logs were subjected to a structured pre-processing pipeline prior to model training. Log parsing and noise removal were first performed to eliminate incomplete or malformed entries. Feature extraction then derived a set of informative attributes from each log entry, including source IP address, session duration, number of commands executed per session, command types, and event timestamps. Categorical variables were encoded using label encoding, and all numerical features were normalized to a common scale to improve model convergence and classification performance.

Model Training

A Random Forest classifier was selected for attack classification based on its well-established advantages in cybersecurity applications [2]. The algorithm constructs an ensemble of decision trees trained on random feature subsets, reducing variance and minimizing the risk of overfitting compared to single decision tree classifiers. Its ability to handle mixed data types, rank feature importance, and provide probabilistic outputs makes it particularly suitable for multi-class threat classification tasks.

The dataset was partitioned into 80% for training and 20% for testing using stratified random sampling to preserve class distribution across both splits. Model hyperparameters, including the number of estimators and maximum tree depth, were optimized through five-fold cross-validation. Training was performed using the Scikit-learn library in Python 3.10.

Real-Time Detection Module

A live detection module was implemented to continuously monitor the Cowrie log file for new entries using an notify-based file watcher. Upon detection of a new log event, the module parses the entry, constructs the corresponding feature vector, and passes it to the loaded Random Forest model for instantaneous classification. The predicted attack category and confidence score are then transmitted to the Flask dashboard via a WebSocket connection, enabling sub-second latency between log generation and alert display.

RESULTS AND DISCUSSION

The Random Forest model was evaluated on the held-out 20% test set comprising 1,148 log entries. Classification performance was assessed using standard metrics including accuracy, precision, recall, and F1-score, computed both per class and as weighted averages. The model achieved near-perfect classification performance across all attack categories under the controlled experimental conditions, demonstrating the effectiveness of the integrated framework.

Attack Category	Precision (%)	Recall (%)	F1-Score (%)	Support (Samples)
Brute Force	98.7	99.1	98.9	312
Port Scanning	99.2	98.6	98.9	241
Reconnaissance	97.8	98.4	98.1	198
Post-Exploitation	98.4	97.9	98.1	187
Vulnerability Scanning	99.0	99.3	99.1	210
Weighted Average	98.6	98.7	98.6	1148

Table 1: Classification Performance Of Random Forest Model Per Attack Category

Study	Method	Dataset Size	Accuracy (%)
IJIIS (2025) [2]	AI-powered IDS + Honeypot	4,200 entries	96.3
MDPI (2024) [3]	Enhanced Random Forest	10,000 entries	97.8
Singh (2026) [4]	K-Means Clustering (Cowrie)	3,500 entries	91.4
Proposed System	Random Forest + Cowrie	5,740 entries	98.6

Table 2: System Performance Comparison With Related Works

As shown in Table 1, brute force and vulnerability scanning attacks achieved the highest F1-scores of 98.9% and 99.1% respectively, attributable to their highly distinctive command patterns and session characteristics recorded by the Cowrie honeypot. Reconnaissance and post-exploitation categories showed marginally lower but still excellent performance, reflecting the greater semantic overlap in commands executed during these phases. These findings are consistent with those reported by MDPI [3], where enhanced ensemble methods outperformed single classifiers on structured honeypot data.

Table 2 contextualizes the proposed system's performance against closely related published works. The proposed framework achieves a weighted average accuracy of 98.6%, surpassing the AI-powered IDS of IJIIS [2] (96.3%), the Cowrie-based clustering approach of Singh [4] (91.4%),and approaching the enhanced Random Forest benchmark of MDPI [3] (97.8%) despite operating on a smaller dataset. The real-time detection latency averaged 0.34 seconds from log entry generation to dashboard alert, demonstrating the system's suitability for operational deployment.

The Flask-based dashboard provided continuous visualization of classification results, including per-session attack category labels, session frequency charts, and geographic IP mapping. Automated alerts

were triggered within 400 milliseconds of a new classification event, enabling near-instantaneous notification to security operators. The Cowrie log analysis engine [5] informed several design decisions in the pre-processing pipeline, particularly regarding session segmentation and command tokenization strategies.

While the experimental results are highly encouraging, it must be acknowledged that the dataset was generated in a controlled virtualized environment. The attack simulations, while representative of common threat vectors, do not fully capture the variability, obfuscation techniques, and polymorphic behaviours characteristic of real-world adversaries. The multi-regional cloud honeynet dataset (MURHCAD) [1] suggests that geographic diversity of attack sources substantially affects traffic patterns, a dimension not covered by the current single-machine experimental topology.

ADVANTAGES AND LIMITATIONS

Advantages

- Real-time attack classification with sub-second detection latency eliminates the delays inherent in manual log analysis.
- Automated machine learning-based analysis significantly reduces the cognitive burden on security analysts.
- The lightweight virtualized architecture enables cost-effective deployment without specialized hardware infrastructure.
- The modular design allows independent updating of the honeypot, model, and dashboard components.
- The Flask dashboard provides intuitive, actionable visualizations accessible through standard web browsers.

Limitations

- The dataset, comprising 5,740 entries from a single controlled environment, may not fully represent the diversity of real-world attack traffic.
- Testing was confined to a virtualized host-only network, and performance in enterprise or cloud-scale environments requires further validation.
- The Random Forest model is susceptible to concept drift as attack patterns evolve; periodic retraining on fresh data will be necessary.
- The current system does not incorporate encrypted traffic analysis, limiting its scope to plaintext protocol interactions logged by Cowrie.

FUTURE WORK

Several directions are identified for extending the proposed framework. First, deep learning architectures such as Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNN) will be investigated for their ability to model temporal attack sequences and detect novel, previously unseen threat patterns. Second, the system will be deployed within cloud-native environments and real enterprise networks to validate performance under production-scale traffic conditions, drawing on publicly available multi-regional datasets such as MURHCAD [1]. Third, integration with Security Information and Event Management (SIEM) platforms will be pursued to enable correlation of honeypot-derived intelligence with alerts from other network sensors. Finally, the training dataset will be substantially expanded through the collection of real attack traffic from geographically distributed honeypot deployments, addressing the diversity limitation identified in the current evaluation.

CONCLUSION

This paper presented an AI-enhanced honeypot framework that integrates a Cowrie-based honeypot with a Random Forest machine learning classifier and a Flask web dashboard for real-time cyber-attack detection and visualization. A dataset of 5,740 labelled log entries covering five distinct attack categories was collected through controlled simulations, and the trained model achieved a weighted average F1-score of 98.6% on the held-out test set. The live detection module demonstrated an average classification latency of 0.34 seconds, enabling near-instantaneous alerting and dashboard updates.

The proposed system effectively bridges the gap between passive honeypot data collection and intelligent, automated threat analysis, addressing a critical limitation of conventional honeypot

deployments. Compared to related works, the framework delivers competitive classification accuracy with the added capability of real-time, end-to-end operation. The findings confirm that integrating machine learning with honeypot infrastructure constitutes a viable and effective approach to proactive cybersecurity defence. Future work will extend the framework to cloud environments, deep learning models, and real-world attack traffic datasets to further validate and enhance its operational capabilities.

REFERENCES

1. M. Feito Casares et al., "Descriptor: Multi-Regional Cloud Honeynet Dataset (MURHCAD)," or XIY preprint arXiv:2601.05813, 2026. [Online]. Available: https://arxiv.org/html/2601.05813
2. J. IJIIS, "AI-powered Intrusion Detection System (IDS) integrated with honeypot technology," International Journal of Intelligent Information Systems, vol. 14, no. 4, pp. 11–18, 2025.
3. MDPI, "Research on intrusion detection based on an enhanced random forest algorithm," Applied Sciences, vol. 14, no. 2, p. 714, 2024. https://doi.org/10.3390/app14020714
4. R. Singh, "Discovering SSH attack patterns using Cowrie honeypot and K-Means clustering," International Journal of Computer Applications, vol. 187, no. 74, pp. 33–40, 2026. https://www.ijcaonline.org/archives/volume187/number74/singh-2026-ijca-926253.pdf
5. E. E. YUCE, "Cowrie Honeypot Log Analysis Engine," GitHub repository, 2026. [Online]. Available: https://github.com/EfeEmirYuce/Cowrie-Honeypot-Log-Analysis-Engine

HOW TO CITE: Femina S.*[1], Sankara Narayanan S. T.[2], Forensic Analysis Of AI-Driven Cyber Deception Networks, Int. J. Sci. R. Tech., 2026, 3 (5), 1029-1034. https://doi.org/10.5281/zenodo.20426964