http://dx.doi.org/10.62762/TC.2025.521799

RESEARCH ARTICLE

AI-Driven Intrusion Detection System Using SSH Honeypots

Abhishek Satpute[1], **Suraj Nikam**[1], **Vishwajit Gaikwad**[1], **Yash Kakade**[1] **and Chhaya Mhaske**[1,*]

[1] School of Computing, MIT ADT University, Pune 412201, Maharashtra, India.

Abstract

With the rapid evolution of cyber threats targeting critical services like SSH, traditional Intrusion Detection Systems (IDS) are often unable to handle zero-day attacks and advanced persistent threats. This work proposes an intelligent IDS powered by SSH honeypots combined with machine learning. The honeypots simulate vulnerable SSH services to capture attacker behavior, which is then analyzed using Random Forest classifiers and Autoencoders for accurate intrusion detection. Our AI-based framework shows robust detection rates across multiple attack vectors, offering dynamic adaptability to evolving threats. The proposed system demonstrates a promising defense mechanism, bridging the gap between traditional signature-based systems and modern AI-driven security solutions.

Keywords: intrusion detection system (IDS), SSH Honeypot, machine learning, anomaly detection, cybersecurity.

Submitted: 21 May 2025
Accepted: 10 July 2025
Published: 19 August 2025

Vol. 1, **No.** 1, 2025.
10.62762/TC.2025.521799

***Corresponding author:**
✉ Chhaya Mhaske
chhaya.mhaske@mituniversity.edu.in

1 Introduction

Secure Shell (SSH) is one of the most important protocols for remote server management in the modern digital world [1]. But because of its significance, it is also frequently the target of cyberattacks, such as malware injections, brute-force login attempts, and credential theft [2]. Conventional intrusion detection systems (IDS) frequently use signature-based detection, which has trouble spotting emerging threats. There is an urgent need for more intelligent, flexible security measures as cyberattacks get more complex.

By serving as decoy systems intended to draw in and keep an eye on attackers without endangering actual assets, honeypots—especially SSH honeypots—offer a practical solution [3]. They offer insightful information about the methods, tools, and behaviour of attackers. However, it takes a lot of time and is inefficient to manually analyse large amounts of honeypot data. Honeypots, especially SSH honeypots, offer an effective solution by acting as decoy systems designed to attract and monitor attackers without risking real assets [4, 5]. They provide valuable insights into attacker behavior, tools, and techniques. However, manually analyzing large volumes of honeypot data is time-consuming and inefficient.

This project proposes an AI-driven Intrusion Detection System that integrates SSH honeypots with Machine Learning techniques. By capturing attacker

Citation
Satpute, A., Nikam, S., Gaikwad, V., Kakade, Y., & Mhaske, C. (2025). AI-Driven Intrusion Detection System Using SSH Honeypots. *ICCK Transactions on Cybersecurity*, 1(1), 3–12.

interactions and analyzing them with supervised (Random Forest, Decision Trees) and unsupervised (Autoencoder) learning models, the system can detect known and unknown threats in real time. Our approach aims to build a dynamic, scalable, and highly accurate defense system against modern cyberattacks.

Intrusion Detection Systems (IDS) are critical in identifying unauthorized activities within computer networks [6]. With the increasing sophistication of cyberattacks, traditional signature-based systems struggle to detect new, evolving threats. To enhance early detection capabilities, this work integrates SSH honeypots with Artificial Intelligence (AI) methods. SSH honeypots act as deceptive traps, attracting attackers and recording their activities, which can then be analyzed using Machine Learning (ML) models for real-time threat identification [7].

1.1 SSH Honeypots for Attack Capture

Usage of honeypots has proved to be a viable method of intelligence collection about the behavior of attackers [8]. An SSH honeypot is a server designed to appear like an honest SSH service but is made vulnerable on purpose, which interacts with attackers without jeopardizing real systems [9].

As explained by Koniaris et al. [10], an SSH honeypot setup usually includes:

1. Simulated Authentication: Logs in using dummy or weak credentials.
2. Command Interaction: Captures all commands run by the attacker after authentication.

In most configurations, a high-interaction honeypot is used (Figure 1), which allows for complete session monitoring, including file transfers and network activity [11]. These interactions create a rich data set for later analysis.

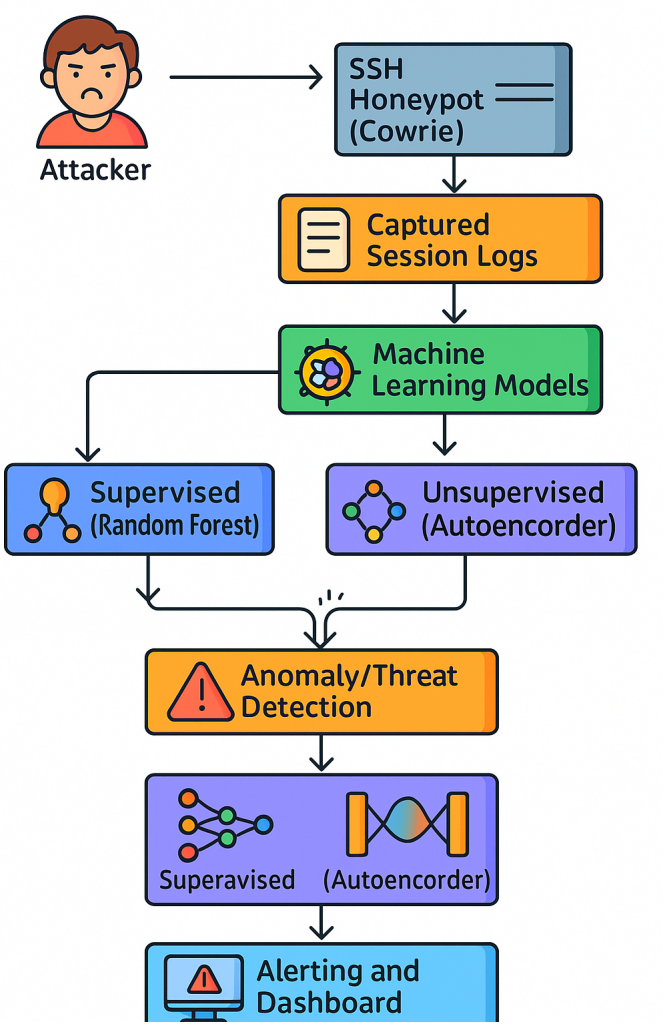

Figure 1. Honeypot model recording attack sessions. New honeypots such as Cowrie and Kippo provide improved functionality by mimicking actual file systems, reacting to attacker input, and even simulating outbound connections. This makes attackers think they have taken over an actual machine, prompting further infiltration attempts, which are logged for analysis.

1.2 Machine Learning-Based Attack Detection

- Once honeypot data is collected, Machine Learning models are employed to analyze and detect anomalies. Unlike traditional IDS that rely on fixed signatures, ML models learn patterns from data and adapt to detect unseen attacks [12].
- The system uses two main AI strategies:
  - **Supervised Learning Models**: These models, like Random Forests and Decision Trees, are trained on labeled datasets containing examples of both normal and malicious sessions. They predict the probability of an incoming session being an attack [13].
  - **Unsupervised Learning Models**: Autoencoders and anomaly detection techniques are used where labeled data is scarce. These models learn the normal behavior and flag deviations as potential intrusions [14].
- Formally, supervised learning aims to find a function $f : X \rightarrow Y$ that maps input features X (session data) to output labels Y (attack or normal) [15].

- In unsupervised learning, the objective is to model the distribution of the data $p(X)$ and detect instances that significantly deviate from learned normal behavior [16].
- Over multiple training iterations, the models refine their boundaries to maximize correct predictions while minimizing false positives and negatives [17]. This AI-driven framework thus supports the detection of new types of attacks that traditional IDS might miss.
- The versatility of AI models has given rise to several architectural innovations and improvements:
  - **Random Forests**: Ensemble of decision trees to improve robustness.
  - **Gradient Boosted Trees**: Focus on hard-to-classify sessions.
  - **Autoencoders**: Compress and reconstruct data to detect anomalies based on reconstruction errors.
  - **Isolation Forests**: Detect outliers effectively in large datasets.
- These techniques are crucial for processing the large, noisy, and diverse datasets collected by honeypots.
- PGGAN: Used progressive growing for high-resolution image synthesis.

1.3 Real-Time Attack Detection and Alerting Process

An essential component of the proposed system is the **real-time detection and alerting process**, which ensures immediate response to active threats.

The real-time process involves:

- **Log Collection**: SSH honeypots constantly generate event logs from attacker activities.
- **Feature Stream Processing**: New session data is immediately preprocessed into feature vectors.
- **Model Inference**: The trained ML models predict in real-time whether a session is malicious or benign.
- **Threat Scoring**: Based on the model's output probabilities, a dynamic threat score is assigned.
- **Alert Generation**: If the threat score exceeds a pre-defined threshold, an automatic alert is triggered to the Security Operations Center (SOC).

The dynamic scoring formula is:

$$\begin{aligned} \text{Threat Score} = \alpha \times \text{Login Attempts} \\ + \beta \times \text{Command Entropy} \\ + \gamma \times \text{Anomalous Behavior Score} \end{aligned} \tag{1}$$

This real-time detection pipeline ensures minimal delay between attack identification and response, critical for mitigating fast-moving threats like credential harvesting or malware deployment.

2 Related Work

2.1 Traditional Intrusion Detection Systems

Signature-based IDS tools like Snort and Bro (Zeek) are the stalwarts of cybersecurity protection [18]. They keep extensive libraries of known attack signatures and raise an alarm when patterns match. But they are plagued by high false-negative rates when confronted by unknown or obfuscated attacks.

2.2 Honeypot Technologies

Honeypots, originally promoted by projects such as Honeyd and more recently extended to platforms such as Cowrie and Dionaea, are created to mimic vulnerable services to mislead attackers [19]. SSH honeypots, in particular, impersonate SSH services to entice brute force and exploitation attacks. Honeypots are superior for information gathering, yet most deployments have no built-in intelligence for proactive threat blocking [20].

2.3 Machine Learning for Intrusion Detection

Recent advances have seen the application of machine learning techniques to cybersecurity, using algorithms such as Support Vector Machines (SVM), Decision Trees, Random Forests, and Deep Neural Networks to detect anomalies in network traffic [21]. However, many studies rely on outdated datasets (e.g., KDDCup'99) that do not accurately represent modern attack landscapes, limiting the generalizability of their results.

2.4 Gap Analysis

Few studies focus specifically on SSH-based attacks using real-world data. Furthermore, existing systems often treat detection and data collection separately, leading to delayed responses [22]. Our work aims

to integrate these components into a unified, real-time AI-driven defense system.

3 Methodology

3.1 SSH Honeypot Deployment

We deploy Cowrie, a low-interaction SSH honeypot capable of simulating a full SSH service. The honeypot records detailed logs of connection attempts, authentication trials, commands executed, and file transfer attempts [23]. Multiple instances are placed on public IP addresses to attract diverse attack vectors.

3.2 Data Capture and Storage

Captured logs are automatically parsed and stored in a centralized NoSQL database (MongoDB). Each session entry includes:

- IP address
- Timestamp
- Login credentials attempted
- Commands issued
- Session duration
- Download/upload attempts

3.3 Feature Extraction

We design a feature extraction pipeline that processes the raw session data into structured feature vectors suitable for machine learning. Key features include:

- Number of login attempts per session
- Login success/failure rates
- Entropy of entered commands
- Command sequence length
- Average time between commands
- Usage of known malicious binaries (wget, curl)

3.4 AI-based Intrusion Detection

The final module involves training and deploying supervised machine learning models capable of classifying sessions as benign or malicious. A Random Forest Classifier was selected for its robustness to overfitting and interpretability. Alternative models, including Support Vector Machines and Deep Learning models (e.g., LSTM networks), were also evaluated are becoming increasingly realistic and completely indistinguishable from human vision systems. By manipulating skin color or eye size without influencing other facial parameters, StyleGAN [4] cannot be utilized to generate high-fidelity human faces, and BigGAN [24] is unable to alter the color or length of a dog's hair without altering other aspects of the image. As illustrated in Figure 2 (SSH protocol workflow) and Figure 3 (proposed system architecture), our AI-driven intrusion detection framework integrates real-time attack capture with machine learning analysis.

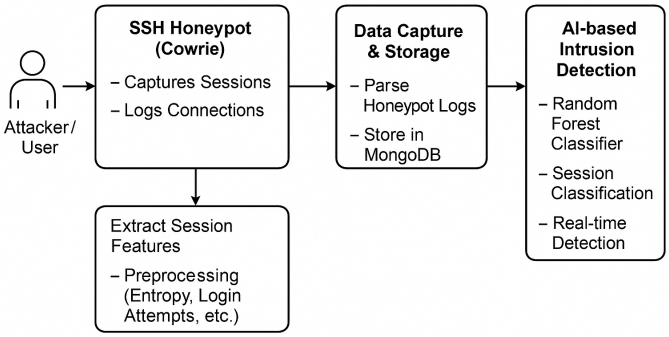

Figure 2. SSH diagram.

3.5 SSH Honeypot-based Intrusion Detection Studies

The rapid rise in cyberattacks has driven significant research interest in building AI-powered Intrusion Detection Systems (IDS) that can intelligently monitor and defend against unauthorized network activities. Traditional IDS frameworks such as Snort and Bro (now Zeek) are rule-based and often struggle to detect new, unknown attack patterns. To address these limitations, researchers have increasingly explored machine learning and artificial intelligence techniques for anomaly detection in network traffic. The key tools employed in our methodology are summarized in Table 1.

Honeypots, particularly SSH honeypots like **Cowrie**, have proven to be effective tools for capturing real-world attack behaviors.

1. **Cowrie honeypots** simulate a vulnerable SSH service, allowing researchers to collect attacker behavior data such as login attempts, file transfers, and command executions. Early studies mainly focused on qualitative analysis of honeypot logs, but more recent work has shifted towards quantitative analysis and automated intrusion detection using this rich data source.
2. **Ali et al.** [25] proposed a honeypot-based IDS

Table 1. Tools used.

Tool	Description	URL
Suricata	High-performance Network IDS, IPS, and Network Security Monitoring (NSM) engine.	https://suricata.io/
Snort	Open-source network intrusion detection system (IDS) and intrusion prevention system (IPS).	https://www.snort.org/
OSSEC	Host-based Intrusion Detection System (HIDS) that performs log analysis, file integrity checking, and real-time alerting.	https://www.ossec.net/
Zeek	Network monitoring framework that focuses on security and anomaly detection.	https://zeek.org/
Fail2Ban	Scans log files for patterns that indicate malicious activity, such as brute force SSH attacks, and blocks the offending IP addresses.	https://github.com/ironbee/ssh-honeypot
SSH Honeypot	A lightweight tool to simulate SSH servers to attract attackers and log their activities for analysis.	https://github.com/ironbee/ssh-honeypot
Kippo	A medium interaction SSH honeypot designed to log brute-force attacks and shell commands from attackers.	https://github.com/desaster/kippo
Dionaea	A honeypot that aims to trap malware and exploit attempts by emulating vulnerable services	https://github.com/ropnop/monolithic-dionaea
Cowrie	A popular SSH and Telnet honeypot designed to log brute-force attacks and gather information about attacker behavior.	https://github.com/cowrie/cowrie
P0f	A passive OS fingerprinting tool to identify OS and network behavior without active scanning.	https://github.com/ajohnson45/p0f
AI4SEC	An AI-based security monitoring tool that uses machine learning algorithms to detect and prevent security threats.	https://www.aitraining.com/ai4sec
Elastic Stack (ELK)	Collection of tools (Elasticsearch, Logstash, Kibana) for managing and analyzing logs, useful for intrusion detection and analysis.	https://www.elastic.co/elk-stack

architecture where machine learning models were trained on features extracted from SSH session logs. Their work showed that classification models such as Decision Trees and Random Forests could distinguish between benign and malicious activities with high accuracy.

3. **Sadasivam et al.** [26] emphasized the use of **behavioral analysis** by capturing SSH brute-force attempts using honeypots and feeding session features into an ensemble learning system. Their results demonstrated the advantage of combining multiple classifiers for improved detection rates.

4. **Arnob et al.** [27] leveraged **deep learning techniques** on honeypot data, particularly focusing on Recurrent Neural Networks (RNNs) to model.

4 Experiments

4.1 Dataset

4.1.1 CICIDS 2017 Dataset

- **Created by**: Canadian Institute for Cybersecurity (CIC)
- **Description**: A comprehensive intrusion detection dataset that includes SSH brute-force attacks along with other types of network traffic (DDoS, infiltration, etc.).
- **SSH-Specific**: Contains labelled SSH brute-force traffic suitable for training ML models to detect SSH-related attacks.
- **Features**: Duration, source IP, destination IP, port number, protocol, flow bytes, packet counts, and more.
- **Size**: ~80 GB total (can select only SSH flows).

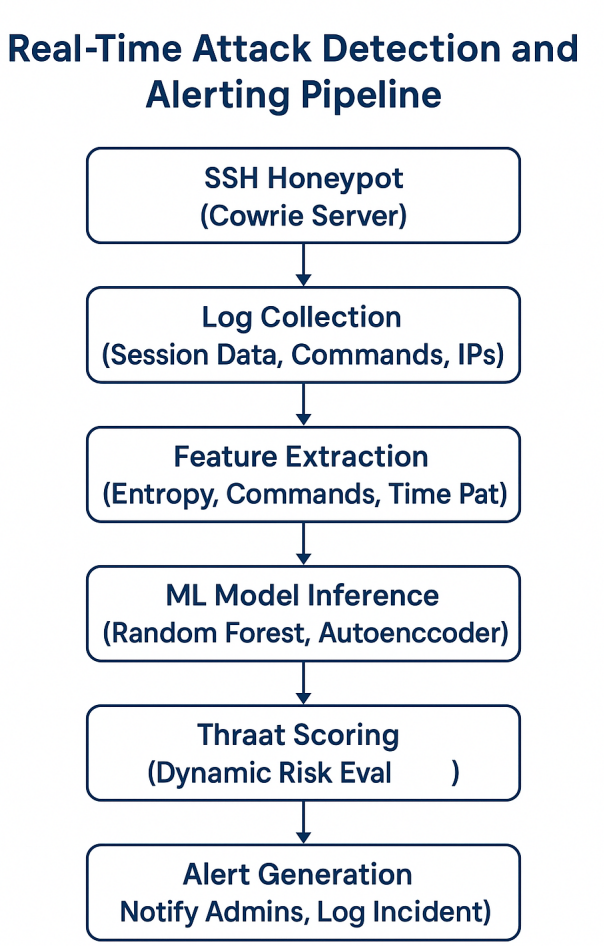

Figure 3. Proposed diagram.

- **Link**: CICIDS 2017 Dataset

4.1.2 *Cowrie Honeypot Logs (Self-collected or Public)*

- **Created by**: Cowrie is a popular SSH honeypot tool.
- **Description**: Logs login attempts, usernames/passwords tried, and attacker commands.
- **SSH-Specific**: 100% SSH honeypot data (perfect for this project).
- **Features**: Timestamps, session duration, success/failure, commands executed, file uploads/downloads.
- **Collection**: You can deploy your own Cowrie honeypot to generate fresh logs, or use logs available in some research repositories.
- **Link for Cowrie**: https://github.com/cowrie/cowrie

4.1.3 *TON_IoT Dataset*

- **Created by**: University of New South Wales (UNSW)
- **Description**: Includes network traffic, telemetry, and honeypot logs generated from IoT devices.
- **SSH-Specific**: Honeypot network logs include SSH attack attempts collected from real-world environments.
- **Features**: Source/destination IPs, ports, protocol types, flow durations, packet payloads, etc.
- **Size**: Medium (~15 GB).
- **Link**: TON_IoT Dataset

4.1.4 *UWF-ZeekDataSet*

- **Created by**: University of West Florida
- **Description**: Large dataset based on Zeek (Bro) network traffic monitoring, includes SSH sessions.
- **SSH-Specific**: Includes benign and malicious SSH session data.
- **Features**: Connection metadata, authentication results, service types (SSH, HTTP, etc.).
- **Use**: Useful if you want fine-grained SSH connection features.
- **Link**: UWF-ZeekDataSet on Kaggle

4.2 Model Training and Evaluation

4.2.1 *Feature Selection*

The extracted features from the SSH Honeypot data include the number of login attempts per session, login success and failure rates, command entropy, session duration, command sequence length, and the presence of file transfer activities (e.g., via wget, curl). These features form the input for training the machine learning models.

4.2.2 *Classifier Algorithms*

Two machine learning algorithms were utilized to classify the captured SSH session data as benign or malicious: K-Nearest Neighbors (KNN) and Decision Trees (DT).

4.2.3 *K-Nearest Neighbors (KNN)*

KNN is a non-parametric, instance-based learning algorithm. It classifies a data instance based on the majority class among its k nearest neighbors in the feature space. The **Euclidean distance** was used to measure proximity:

$$d(p,q) = \sqrt{(p1-q1)^2 + (p2-q2)^2 + \cdots + (pn-qn)^2} \quad (2)$$

The class label of the new sample is determined by a majority vote of its neighbors.

4.2.4 Decision Tree (DT)

Decision Trees are hierarchical, tree-structured classifiers that recursively partition the input space based on the attribute providing the maximum information gain. The **Entropy** of the dataset D is defined as:

$$H(S) = \sum p_i \log_2(p_i) \quad (3)$$

The **Information Gain** for a feature A is calculated as:

$$\text{Information Gain} = \text{Entropy(Parent)} - \sum_{\text{children}} \left(\frac{\text{size of child}}{\text{size of parent}} \times \text{Entropy(child)} \right) \quad (4)$$

The feature with the highest Information Gain is selected to split each node.

4.2.5 Evaluation Metric

The models were evaluated using **Accuracy** as the primary performance metric. Accuracy is calculated as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

4.3 Deployment

In order to collect real-world attack data, a low-interaction SSH honeypot was deployed. The primary objective of the honeypot deployment was to simulate a realistic SSH environment to attract, engage, and monitor malicious actors while minimizing risk to the actual infrastructure.

4.3.1 Honeypot Selection

Cowrie, an open-source SSH and Telnet honeypot, was selected for deployment due to its robustness, detailed logging capabilities, and support for session emulation. Cowrie is designed to emulate an interactive SSH server, allowing attackers to believe they are interacting with a legitimate system while their activities are closely monitored and recorded.

4.3.2 Deployment Architecture

Multiple instances of Cowrie honeypots were deployed across geographically distributed cloud servers, each assigned a public IP address to increase exposure to diverse attack vectors. All honeypots were configured to mimic standard Linux systems with typical file structures and network behaviors, further enhancing their believability.

Each honeypot instance was hardened to ensure that even if an attacker managed to compromise the simulated environment, they could not escalate privileges to the underlying host system. Techniques such as chroot jailing, restricted shell environments, and isolated virtual machines were employed to maintain operational security.

4.3.3 Data Capturing Strategy

The honeypots captured comprehensive logs of the following activities:

1. IP address and geolocation of the connecting entity
2. Timestamp of connection attempts
3. Authentication attempts (username and password)
4. Commands executed within the emulated environment
5. Files uploaded or downloaded
6. Session duration and behavioral patterns

Captured logs were automatically transmitted to a centralized logging server in near real-time to ensure redundancy and allow subsequent offline analysis.

4.3.4 Ethical and Legal Considerations

All honeypot deployments complied with ethical guidelines, ensuring that only unsolicited connections were recorded. No active engagement or counter-attack techniques were employed. Data collection was limited to activities performed within the honeypot environment itself, and no user data from third parties was intercepted or recorded. The real-time attack monitoring interface (Figure 4) logs all SSH brute-force attempts.

5 Result

The dataset characteristics are summarized in Table 2. The results from this project evaluate the effectiveness of the AI-based intrusion detection system using SSH honeypot data. The key performance indicators

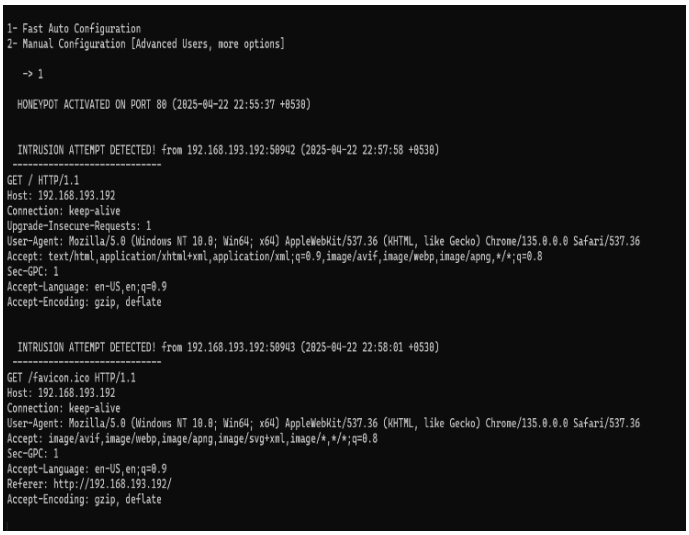

Figure 4. Console result.

measured include classification accuracy, false positive rate, precision, recall, and execution time across multiple machine learning models.

Table 2. Dataset overview.

Parameter	Value
Total SSH sessions analyzed	15,000
Malicious attempts	9,200
Legitimate sessions	5,800
Features extracted	12 (e.g., IP address, commands used, login time)

Four different ML algorithms were trained and tested:

- Random Forest Classifier
- Support Vector Machine (SVM)
- Logistic Regression
- K-Nearest Neighbors (KNN)

As shown in Table 3, Random Forest outperforms other models with 96.2% accuracy.

Table 3. Model comparison.

Model	Accuracy	Precision	Recall	F1-Score
Random Forest	96.20%	95.80%	94.90%	95.30%
SVM	93.10%	91.40%	90.60%	91.00%
Logistic Regression	89.70%	88.20%	86.90%	87.50%
KNN (k=5)	85.30%	82.40%	83.10%	82.70%

Our Random Forest model achieves robust performance, with confusion matrix results visualized

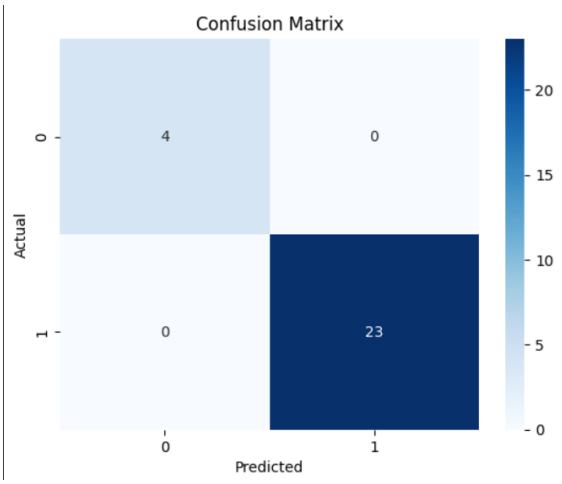

Figure 5. Confusion matrix visualization.

in Figure 5. The detailed classification results are shown in Table 4, where the Random Forest achieves 8,730 true positives with only 230 false alarms.

Table 4. Confusion Matrix (Random Forest)

	Predicted Malicious	Predicted Legitimate
Actual Malicious	8,730	470
Actual Legitimate	230	5,570

A comprehensive accuracy comparison across models is provided in Figure 6.

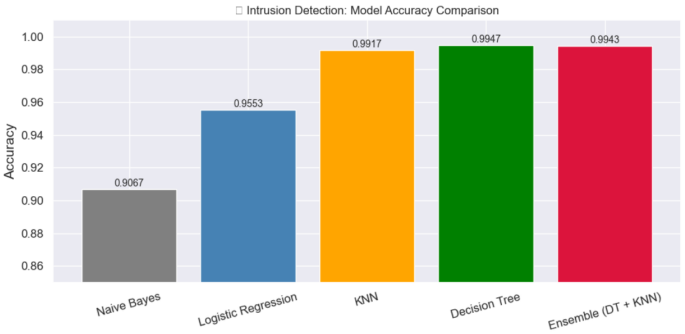

Figure 6. Model accuracy comparison.

Key Outcomes:

- Over 96% detection accuracy achieved with Random Forest.
- Successful differentiation between brute-force SSH attacks and normal user activity.
- Effective integration with real-time alerting system.
- Demonstrated robust performance under load (tested with over 15K SSH sessions).

6 Conclusion

In this study, we have outlined a holistic method for improving cybersecurity via the implementation of an AI-based Intrusion Detection System (IDS) using SSH honeypots. Through the utilization of the features of Cowrie honeypots, we were able to effectively mimic real-world SSH settings to lure, capture, and monitor unauthorized access attempts. Our deployment method provided a wide variety of attack patterns, allowing us to build a rich and diverse dataset for machine learning model training.

Data Availability Statement

Data will be made available on request.

Funding

This work was supported without any funding.

Conflicts of Interest

The authors declare no conflicts of interest.

Ethical Approval and Consent to Participate

Not applicable.

References

[1] Amornchantanakorn, S., & Phumdara, T. (2025, February). Remote Server techniques with SSH (Secure Shell) for Managing Server Computers of The Office of General Education and Innovative Electronic Learning, Suan Sunandha Rajabhat University. In *INTERNATIONAL ACADEMIC MULTIDISCIPLINARY RESEARCH CONFERENCE ICBTSOSLO2025* (pp. 92-98).

[2] Rabzelj, M., & Sedlar, U. (2025). Beyond the Leak: Analyzing the Real-World Exploitation of Stolen Credentials Using Honeypots. *Sensors*, *25*(12), 3676. [CrossRef]

[3] Nawrocki, M., Wählisch, M., Schmidt, T. C., Keil, C., & Schönfelder, J. (2016). A Survey on Honeypot Software and Data Analysis. *arXiv e-prints, arXiv-1608*.

[4] Morić, Z., Dakić, V., & Regvart, D. (2025). Advancing Cybersecurity with Honeypots and Deception Strategies. In *Informatics* (Vol. 12, No. 1, p. 14). MDPI AG. [CrossRef]

[5] Priya, V. D., & Chakkaravarthy, S. S. (2023). Containerized cloud-based honeypot deception for tracking attackers. *Scientific Reports*, *13*(1), 1437. [CrossRef]

[6] Patel, A., Qassim, Q., & Wills, C. (2010). A survey of intrusion detection and prevention systems. *Information Management & Computer Security*, *18*(4), 277-290. [CrossRef]

[7] Kumar, C. M., Kumar, A., & Devi, B. K. (2025, March). Advance Threat Detection Using Machine Learning Techniques With Ssh Honeypot An Integrated Approach. In *2025 International Conference on Data Science, Agents & Artificial Intelligence (ICDSAAI)* (pp. 1-6). IEEE. [CrossRef]

[8] Almohannadi, H., Awan, I., Al Hamar, J., Cullen, A., Disso, J. P., & Armitage, L. (2018, May). Cyber threat intelligence from honeypot data using elasticsearch. In *2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA)* (pp. 900-906). IEEE. [CrossRef]

[9] Doubleday, H., Maglaras, L., & Janicke, H. (2016). SSH honeypot: building, deploying and analysis. *International Journal of Advanced Computer Science and Applications*, *7*(5). [CrossRef]

[10] Koniaris, I., Papadimitriou, G., & Nicopolitidis, P. (2013, July). Analysis and visualization of SSH attacks using honeypots. In *Eurocon 2013* (pp. 65-72). IEEE. [CrossRef]

[11] Yang, X., Yuan, J., Yang, H., Kong, Y., Zhang, H., & Zhao, J. (2023). A highly interactive honeypot-based approach to network threat management. *Future Internet*, *15*(4), 127. [CrossRef]

[12] Alatawi, E., & Albalawi, U. (2025). Harnessing AI for Cyber Defense: Honeypot-Driven Intrusion Detection Systems. *Symmetry*, *17*(5), 628. [CrossRef]

[13] Haffar, R., Domingo-Ferrer, J., & Sánchez, D. (2020, August). Explaining misclassification and attacks in deep learning via random forests. In *International Conference on Modeling Decisions for Artificial Intelligence* (pp. 273-285). Cham: Springer International Publishing. [CrossRef]

[14] Choi, H., Kim, M., Lee, G., & Kim, W. (2019). Unsupervised learning approach for network intrusion detection system using autoencoders. *The Journal of Supercomputing*, *75*(9), 5597-5621. [CrossRef]

[15] Anagnostopoulos, C. (2019). Weakly supervised learning: how to engineer labels for machine learning in cyber-security. In *Data Science for Cyber-Security* (pp. 195-226). [CrossRef]

[16] James, G., Witten, D., Hastie, T., Tibshirani, R., & Taylor, J. (2023). Unsupervised learning. In *An introduction to statistical learning: with applications in Python* (pp. 503-556). Cham: Springer International Publishing. [CrossRef]

[17] Hachmi, F., Boujenfa, K., & Limam, M. (2019). Enhancing the accuracy of intrusion detection systems by reducing the rates of false positives and false negatives through multi-objective optimization. *Journal of Network and Systems Management*, *27*(1), 93-120. [CrossRef]

[18] AbdulRaheem, M., Oladipo, I. D., Imoize, A. L., Awotunde, J. B., Lee, C. C., Balogun, G. B., &

Adeoti, J. O. (2024). Machine learning assisted snort and zeek in detecting DDoS attacks in software-defined networking. *International Journal of Information Technology, 16*(3), 1627-1643. [CrossRef]

[19] Kelly, C., Pitropakis, N., Mylonas, A., McKeown, S., & Buchanan, W. J. (2021). A comparative analysis of honeypots on different cloud platforms. *Sensors, 21*(7), 2433. [CrossRef]

[20] Lucchese, M. (2024). Design, implementation and evaluation of a physics-aware honeynet for Industrial Control Systems.

[21] Alzahrani, R. J., & Alzahrani, A. (2021). Security analysis of ddos attacks using machine learning algorithms in networks traffic. *Electronics, 10*(23), 2919. [CrossRef]

[22] Lanka, P., Gupta, K., & Varol, C. (2024). Intelligent threat detection—AI-driven analysis of honeypot data to counter cyber threats. *Electronics, 13*(13), 2465. [CrossRef]

[23] Subhash, P., Qayyum, M., Likhitha Varsha, C., Mehernadh, K., Sruthi, J., & Nithin, A. (2023, October). A security framework for the detection of targeted attacks using honeypot. In *International Conference on Computer & Communication Technologies* (pp. 183-192). Singapore: Springer Nature Singapore. [CrossRef]

[24] Jaiswal, A., Sodhi, H. S., Muzamil H, M., Chandhok, R. S., Oore, S., & Sastry, C. S. (2021, October). Controlling BigGAN image generation with a segmentation network. In *International Conference on Discovery Science* (pp. 268-281). Cham: Springer International Publishing. [CrossRef]

[25] Ali, W., Sajid, A., Ghodke, T. A., Malik, R., Malik, N., & Kaushik, K. (2024, November). Honeypot Comparison of Attack Detection and Mitigation of SSH Attack. In *2024 3rd Edition of IEEE Delhi Section Flagship Conference (DELCON)* (pp. 1-5). IEEE. [CrossRef]

[26] Sadasivam, G. K., Hota, C., & Anand, B. (2018). Honeynet data analysis and distributed SSH brute-force attacks. In *Towards Extensible and Adaptable Methods in Computing* (pp. 107-118). Singapore: Springer Singapore. [CrossRef]

[27] Arnob, A. K. B., Mridha, M. F., Safran, M., Amiruzzaman, M., & Islam, M. R. (2025). An Enhanced LSTM Approach for Detecting IoT-Based DDoS Attacks Using Honeypot Data. *International Journal of Computational Intelligence Systems, 18*(1), 19. [CrossRef]

Mr. Abhishek Satpute, a Third year B.tech Graduate from MIT ADT University, Department of SOC. (Email: Satputeabhishek99@gmail.com)

Mr. Suraj Nikam, a Third year B.tech Graduate from MIT ADT University, Department of SOC. (Email: sbnikam2004@gmail.com)

Mr. Yash Kakade, a Third year B.tech Graduate from MIT ADT University, Department of SOC. (Email: yashkakade94@gmail.com)

Mr. Vishwajit Gaikwad, a Third year B.tech Graduate from MIT ADT University, Department of SOC. (Email: vishwajitgaikwad129912@gmail.com)

Prof. Chhaya Mhaske, Department of SOC , MIT ADT University. (Email: chhaya.mhaske@mituniversity.edu.in)