Advances in Artificial Intelligence and Machine Learning;Research (Ahead of Print) Received 12-12-2025; Accepted 04-03-2026; Published online 11-03-2026

# Honeypot-Driven Hybrid Continuous Learning Platform for Proactive Cyber Threat Detection

**Fahad Alkamli** falkamli@stu.kau.edu.sa
*Department of Information Technology*
*Faculty of Computing and Information Technology*
*King Abdulaziz University*
*Jeddah, Saudi Arabia*

**Morched Derbali** mderbali@kau.edu.sa
*Department of Information Technology*
*Faculty of Computing and Information Technology*
*King Abdulaziz University*
*Jeddah, Saudi Arabia*

**Tariq Mohamed Ahmed** tmahmad@kau.edu.sa
*Department of Information Technology*
*Faculty of Computing and Information Technology*
*King Abdulaziz University*
*Jeddah, Saudi Arabia*

**Corresponding Author:** Fahad Alkamli



## Abstract

This research paper introduces a hybrid continuous learning platform using honeypots that will improve proactive cyber threat detection in dynamic environments. The proposed system combines a monitored warm-up step and an incremental semi- supervised learning route that constantly adapts to actual web traffic that has been gathered using a honeypot. The platform makes use of River framework and Multinomial Naive Bayes to process labeled and unlabeled HTTP request logs. The experimental findings across seventeen cycles of learning depict a definite enhancement in AUC, accuracy, precision and F1 following a shift from supervised to semi-supervised training. Confidence based filtering ensures that only trustworthy pseudo labels are used to update the model, which helps stabilize performance and reduces the risk of catastrophic forgetting. The paper has proven the hypothesis that incremental learning applied to real attack traffic offers quantifiable benefits over static batch learning. The findings show that a scalable, flexible and autonomous intrusion detection mechanism is practical and can improve itself over the long run.

**Keywords:** Incremental learning, Batch learning, Data streams, River framework, Online machine learning, Concept drift, Real-time processing.



**Citation:** Fahad Alkamli, et al. Honeypot-Driven Hybrid Continuous Learning Platform. **Advances in Artificial Intelligence and Machine Learning**. 2026. (Ahead of Print). https://dx.doi.org/10.54364/AAIML.2026.62287

# 1. INTRODUCTION

Cybersecurity is becoming increasingly critical, especially in the era of Artificial Intelligence (AI). To be able to defend against this emerging threat, security experts need to utilize all available resources and integrate them with Artificial Intelligence to create intelligent, self-learning tools. In this paper we propose a system that actively learns and evolves, addressing the limitations of current tools which depend solely on statically trained models at specific point in time. Having a system that is leveraging AI to continuously learn and evolve is critical. The system we propose will use an incremental learning model using MultinomialNB algorithm that will be based on River platform to train the model initially on static dataset while actively learning from a honeypot using semi-supervised learning. The semi-supervised learning model will utilize a honeypot to collect and categorize attacks based on confidence threshold, which then will be integrated frequently into the model. This platform will ensure the model is actively learning from real-life attacks collected and processed in a controlled environment.

## 1.1 Intrusion Detection & Honeypot

Intrusion Detection is the process of monitoring and analyzing a network traffic, system logs and user activities to identify unauthorized access, policy violations or malicious activities. Intrusion detection constitutes a key component of security, enabling organizations to detect and respond to threats before they inflict major damage. Intrusion Detection Systems(IDS) are classified into Host Based IDS (HIDS) and Network Based IDS (NIDS). NIDS scrutinizes network traffic to find anomalies or known attack signatures and HIDS considers activities on individual devices and log system activities as well as file integrity.

The honeypot works as a security deception tool that simulates vulnerable targets to detect and analyze malicious activities in information technology environments. The deceptive system attracts potential attackers to deliver security teams vital information about malicious techniques and enhance their defensive capabilities. The two primary sectors of honeypots are low-interaction honeypots that run minimal services for basic attack recording with reduced consequences and high-interaction honeypots that imitate genuine systems to draw attackers deeply while revealing comprehensive network information. Honeypots deployed by organizations to help them discover new threats while studying attack activities and strengthening their intrusion detection capabilities while protecting their genuine assets.

## 1.2 Machine Learning Approaches

Machine Learning Approaches in cybersecurity refers to the use of algorithms to process the data, work out patterns, predict, and detect security related threats. Machine learning has many approaches depending on the use case of the system as shown in FIGURE 1, and a brief explanation of the purpose of each algorithm is presented in TABLE 1. These approaches help the intrusion detection, malware classification, phishing detection and the fraud prevention by the better accuracy and adaptivity to the dynamically changing cyber space.

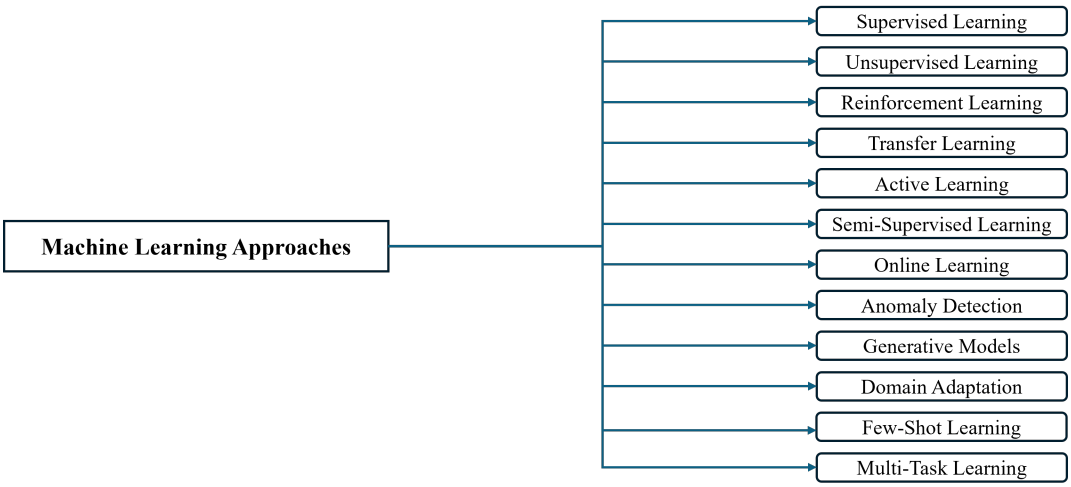


Figure 1: Machine learning approaches diagram

Table 1: Machine Learning Approaches Description

| Approach | Description |
|---|---|
| Supervised Learning | Uses labeled data to predict outcomes. |
| Unsupervised Learning | Finds patterns or groupings in unlabeled data. |
| Reinforcement Learning | Learns by interacting with an environment, receiving rewards or penalties as feedback. |
| Transfer Learning | Transfers knowledge from one task to another. |
| Active Learning | Actively selects the most informative data points for labeling or focuses on uncertainty. |
| Semi-Supervised Learning | Uses a mix of labeled and unlabeled data. |
| Online Learning | Continuously updates models with new incoming data. |
| Anomaly Detection | Identifies unusual patterns not conforming to expected behavior. |
| Generative Models | Creates new content like images or text. |
| Domain Adaptation | Transfers knowledge between domains with distribution shifts. |
| Multi-Task Learning | A single model learns multiple tasks simultaneously, leveraging shared features. |
| Few-Shot Learning | Focuses on learning from a small number of examples per class. |

The rest of this paper will be structured as follows.

In Section 2, the literature focuses on review of existing literature on incremental and batch learning techniques.

The problem and the research objectives are outlined in Section 3. Section 4 briefly discusses the motivation and the objectives of the research. In Section 5, the experimental setup is discussed by providing a description of the dataset preparation, feature engineering, and learning pipelines. Section 6 discusses the design choices and the different layers in the platform, while the flowchart presents the flow of data between the honeypot and the IDS. Section 7 documents and discusses

the experimental performance, benchmarking metrics such as AUC, F1 score, accuracy, recall and precision.

Under Section 8, the important observations, which include the effects of imbalance of classes on behavior of models are raised.

In the end, the study is concluded in Section 9 with a recommendation of future research directions.

## 2. RELATED WORK

Incremental learning has emerged as an essential method for processing real-time streams of continuous data, as compared to the batch learning technique that presupposes a complete set of data before starting the training process. Batch learning tends to have excessive retraining costs and fails to work with non-stationary data, which excludes it, especially in the cybersecurity and IoT fields.

In [1], L. Huang and Q. Zhu propose an adaptive honeypot engagement strategy based on Semi-Markov Decision Processes (SMDP) for attacker behavior analysis and defense strategy optimization. Rewards and risks are quantified as model, and reinforcement learning is used to update policies around when to engage. The study illustrates through numerical simulations effective threat engagement while minimizing risk of exploitation. The second section briefly encompasses active cyber defense methodologies, particularly the intersection between intelligence gathering and security risk. The approach is supported by mathematical formulations and experimental results. Moreover, the research outlines various engagement strategies for increasing threat intelligence gathering. The methodology is formalizing the SMDP-based honeypot strategy and its parameters through reinforcement learning. To evaluate the effectiveness of the proposed engagement model, an in-depth study of adversarial behaviors is carried out.

Then, P. Lanka et al. [2] introduce an AI based solution to analyze honeypot data to fight against cyber threats. The paper discusses using large language models (LLMs) for real time anomaly detection in network security environment. Through real world deployments, the research validates the model's effectiveness, increasing detection accuracy and reducing response time. The results show how machine learning can play a part in threat intelligence and cybersecurity automation. Experimental results for several datasets are presented which show the robustness of the model. In addition, the research explores the computational efficiency of the AI based detection model and its feasibility in real time application. A comparative analysis of the proposed model with existing AI based detection models is presented, highlighting the unique contributions of the work. Nevertheless, the performance is highly contingent on the quality and diversity of training data, and it may not be sufficiently robust to unseen attack scenarios.

In R. H. R. Al-Ruwaili and O. M. Ouda [3], a hybrid classification model based on both supervised and unsupervised learning is developed for network attack detection. Standards are imposed by the study by undertaking an initial clustering based labeling phase before implementing classification algorithms. The performance evaluation on the NSL-KDD dataset shows that the proposed approach achieves competitive accuracy with fully supervised methods while overcoming class imbalance issue. It was shown that mixing learning paradigms improves detection capabilities. A clustering based data preprocessing methodology, feature selection and classifier evaluation technique is used.

The computational complexity of the proposed hybrid method is also investigated in this paper to determine its efficiency in large applications.

In [4] X. Yang et al. explore a highly interactive honeypot based network threat management framework. By designing such realistic attack environments, the system is meant to attract and analyze sophisticated cyber threats. The research then compares the advantages of high-interaction honeypots to traditional low interaction systems, and explains how they provide better threat intelligence. The results show the modular honeypot architectures improve adaptability and effectiveness in cyber defense. It also describes attack behavioral patterns and adaptive honeypot response. In addition, this paper evaluates different honeypot configurations to capture advanced persistent threats (APTs).

S. Dowling et al. [5] propose an adaptive and agile honeypot framework based on reinforcement learning. The dynamic cybersecurity solutions discussed in the study are necessary as malware threats evolve. It shows how machine learning can improve honeypot interactions to capture APTs. A validation of the framework's ability to adapt to emerging attack patterns while maintaining effective deception is achieved through experimental evaluation. The most important strength of the framework is its adaptability, but the study does not provide comparative analysis against other honeypot techniques, which makes it difficult to evaluate its relative performance gains.

Y. Ahmed et al. [6], investigate securing smart cities by exploring three main goals: It identified which datasets are being generated by IoT-focused honeypots, analyzed machine learning algorithms for threat detection and suggested the holistic security strategies. The study shows improved IoT cyberattack detection and mitigation of cyberattacks through incorporating honeypot data into IoT security frameworks using both machine learning and neural network techniques on real-world cyber-attack data collected across a range of different honeypots representing various types of IoT devices. While this research expands current knowledge of the field, it also provides actionable guidance for implementing security that is resilient within a range of IoT contexts, thereby creating a major gap in the field.

In [7], D. A. Firmansyah and A. Zahra explored the possibility of using machine learning to improve honeypot based threat detection. As honeypots generate large amounts of data, they can overwhelm human analysis. Then, this research investigates how machine learning can automate this analysis and improve threat detection accuracy. Using real world honeypot data the performance of various machine learning algorithms is evaluated on knowing whether or not a given computer is compromised by malware. Based on their results, their findings suggest that the Random Forest algorithm reached the highest accuracy score of 99.20%, suggesting how machine learning can greatly enhance honeypot based threat detection as well as facilitate faster incident response.

Using deception technology, an adaptive honeypot framework for IoT security is proposed by D.S. Morozov et al. [8]. Based on observed attack vectors, the framework dynamically adapts its behavior to be able to engage real world threats. To evaluate honeypots effectiveness in critical infrastructure protection, a methodology of study is introduced. The results bolster the case for adaptable security mechanisms in the IoT. Yet, the work lacks field implementation and does not perform long term adversarial adaptation to a honeypot system.

M. R. Siddique et al. [9] study the combination of machine learning based smart agent into honeypot architectures. The first part of the study is to propose an intelligent honeypot system that employs

machine learning models to predict and remedy cyber threats in real time. Models like decision trees, regression, and random forests are evaluated by the research for the detection accuracy. The paper presents adaptive learning algorithms which improve the security defenses. The system's real time feasibility and the computational efficiency are not explored well. Additionally, the study does not discuss adversaries of machine learning based honeypot systems.

Y. T. Abewa and S. Z. Melese [10] propose a web application honeypot designed to adapt dynamically to evolving cyber threats. It presents a module, and scalable honeypot framework that combines real time targeted attack, logging, and builds a request fingerprinting mechanism. In theoretical and experimental results average engagement time is 769.38 seconds and throughput is 105 requests per second. The research reveals that dynamic interaction improves honeypot from the attacker's viewpoint by tricking attackers and catching advanced intrusion patterns. Despite this, the study neither compares with traditional honeypot models nor evaluates the long-term adaptability of the system in subject to variants of the attack scenarios.

A comparison of threat detection approaches is described in TABLE 2.

Table 2: Comparison of Honeypot-Based Threat Detection Approaches

| Paper | Year | Dataset | Model | Honeypot | Learning | Disadvantages |
|---|---|---|---|---|---|---|
| [1] | 2019 | Simulated Honeynet Data | Reinforcement Learning (SMDP) | No | Incr. | High computational cost |
| [2] | 2024 | Honeypot Data (Multiple Attacks) | AI-Driven LLMs | Yes | Static | High processing resource demand |
| [3] | 2023 | NSL-KDD Dataset | Hybrid Supervised/Unsupervised | No | Static | Requires careful preprocessing |
| [4] | 2023 | High-Interaction Honeypot Data | Modular High-Interaction Honeypot | No | Incr. | Resource-intensive |
| [5] | 2020 | IoT Malware Dataset | Adaptive Honeypot (RL) | No | Incr. | Requires frequent policy updates |
| [6] | 2024 | Real-World IoT Honeypot Data | ML-Based IoT Threat Detection | No | Static | Vulnerable to evolving attacks |
| [7] | 2023 | Honeypot Malware Data | Random Forest-Based Detection | Yes | Static | Risk of overfitting |
| [8] | 2024 | IoT Honeypot Data | Adaptive Honeypot with ML | No | Incr. | Needs real IoT integration |
| [9] | 2024 | Cyber Honeypot Data | ML-Powered Smart Agents | Yes | Static | High implementation complexity |
| [10] | 2024 | Web App Honeypot Logs | Dynamic Interactive Honeypot | No | Static | Depends on attack predictability |

## 3. PROBLEM DEFINITION

The modern cybersecurity defense is overwhelmed by the growing sophistication of cyber threats, especially the zero-day attacks. Traditional IDS, rule-based firewalls and heuristic anomaly detection approaches are reactive in nature. These approaches are also very much reliant on manual updates to adapt and detect new patterns of attacks [1, 5]. These static models are not flexible and cannot cope with changing dangers. While machine learning is becoming more popular in the field of cybersecurity for anomaly detection and attack classification, most models still rely on static, pre-labeled datasets [2, 3, 7].

Scientific Value: This research presents a Honeypot-Driven Hybrid Continuous Learning Platform that combines supervised learning and semi-supervised learning in an incremental learning platform to better eliminate false positives and better detect emerging threats.

Practical Value: The proposed system is designed to enable the creation of a simple, scalable, and affordable threat detection system by automating learning from data collected from honeypots and making the learning process incremental, this provides a more robust shield against unknowns and constantly evolving cyber-attacks.

## 4. MOTIVATION & OBJECTIVES

This research will be mainly driven by the gap of the modern IDS with regard to its static model. The traditional systems are based on fixed datasets which are incapable of reflecting the alarming rate of drift of the zero-day attacks. As much as honeypots are giving real time data there is not a framework that exists to connect the gap between a honeypot raw data collection and independent and continuous model update. This paper will respond to this challenge by creating an integrative loop which will convert data under the influence of the deception approaches in real-time into instantaneous model intelligence, namely targeting the adaptive aspect of the web-based threats.

## 5. EXPERIMENTAL METHODOLOGY

This section discusses the experimental setup that was used to measure the performance of the incremental model.

### 5.1 Hardware & Software

- **Operating System**: Ubuntu 20.04 (64-bit)
- **Processor**: 2 x 1.8 GHz
- **Memory**: 4096 MB
- **Storage**: 40 GB
- **Bandwidth**: 100 Mbps (Monthly limit: 4 TB)

- **Python Version**: 3.13.1.
- **River Version**: 0.22.0.

### 5.2 Dataset Construction

The Apache Web App dataset [11] consists of HTTP request logs containing features such as the request method, URL, status code, referer, and browser.

The dataset was designed to work with both the incremental training model, which requires files of the same size to showcase the nature of the incremental process, and batch learning, which needs a dataset consisting of accumulated files, meaning batches will double in size as the gathering of the data increases.

- Supervised dataset construction: a script was used to split of the Apache dataset[11] into small files with a size of 2513 records. The total generated files are seven parts and the eighth part was used as a test file to evaluate the model, consisting of 7539 records.
- Honeypot collected dataset construction: For the honeypot collected dataset, a script was executed to split them by month as the honeypot was running for almost ten months. The files vary in size depending on the attacks that occurred in that month. The total number of files is seven files of unsupervised records.

### 5.3 Feature Extraction and Preprocessing

The experiment includes a preprocessing function to transform the raw HTTP request logs into a feature-rich format for learning. The script applies the following steps:

- Label Generation: If the "Label" column is missing, it is created by marking all non-200 HTTP status codes as malicious (1) and 200 codes as benign (0).
- Request Parsing: If "Method" is missing, the HTTP method and URL are extracted from the raw "request" string using regular expressions.
- Missing Value Handling: Fills missing values in "Status", "URL", and "Method" with default placeholders to prevent processing errors.
- Error Flag: A binary feature named "shows_error" is created, where requests with status codes between 300 and 499 are marked as 1.
- Combined Text Field: A placeholder "combined_text" is created from the "URL" field to support potential NLP-based extensions.
- Injection Detection: Two custom features, "sql_injection" and "command_injection", are derived using external functions that analyze the URL for known attack patterns.
- Uncommon Method Flag: Another binary flag "uncommon_method" is generated using a function that checks if the HTTP method is rare or suspicious (e.g., DELETE, TRACE).

This transformation is applied consistently before each learning cycle to ensure uniform input formatting for both supervised and semi-supervised learning.

### 5.4 Pipeline Framework Justification

The experimental pipeline was implemented using the `River` library, which provides tools for real-time and incremental machine learning [12]. Unlike traditional batch-learning frameworks such as `scikit-learn` [13] or `imblearn` [14], which require the entire dataset to be loaded into memory, `River` supports instance-by-instance learning using streaming APIs.

### 5.5 Class Imbalance Strategy

To reduce the risk of class imbalance leading to poor model performance, we used `imblearn.RandomSampler` with the incremental learning mode. This component serves as an interface to classifiers, it trains the underlying model by undersampling and oversampling the received stream such that the effective distribution of classes follows a given distribution.

We set this target distribution at 0.5 per class. For `batch` models, we applied `RandomUnderSampler` using a `majority` class reduction strategy.

## 6. PLATFORM DESIGN AND ARCHITECTURE

FIGURE 2 demonstrates the design of the architecture. The platform consists of three layers: the honeypot layer, which includes a honeypot that collects attacks and stores them in a database while offering a secure API for the IDS to request logs and check the honeypot's status. The fetch_logs API returns a JSON-formatted list of recent access logs collected from the NGINX server while maintaining a state to ensure that only new records are provided to the IDS.

The second layer is the intelligence IDS layer, where the hybrid IDS engine resides. The hybrid IDS performs the supervised learning, periodically fetch requests from the honeypot and execute the semi-supervised training. Furthermore, the IDS will offer an API to accept requests from clients, in which the model will evaluate an access request and return a JSON response to the client. The returned JSON response will consist of the verdict and the confidence.

The action of accepting or rejecting the request is left to the client according to their security sensitivity, in which services that require high security may reject requests that are above a specific threshold like 70%, while other services like an email service will reject only requests with 90% confidence. This feature was implemented to give services a flexible security policy mechanism.

Lastly, the third layer contains the services cluster, which in this figure consists of three services: mail server, website server, and authentication server. The services in this cluster showcase the varying security policy that is needed for different services.

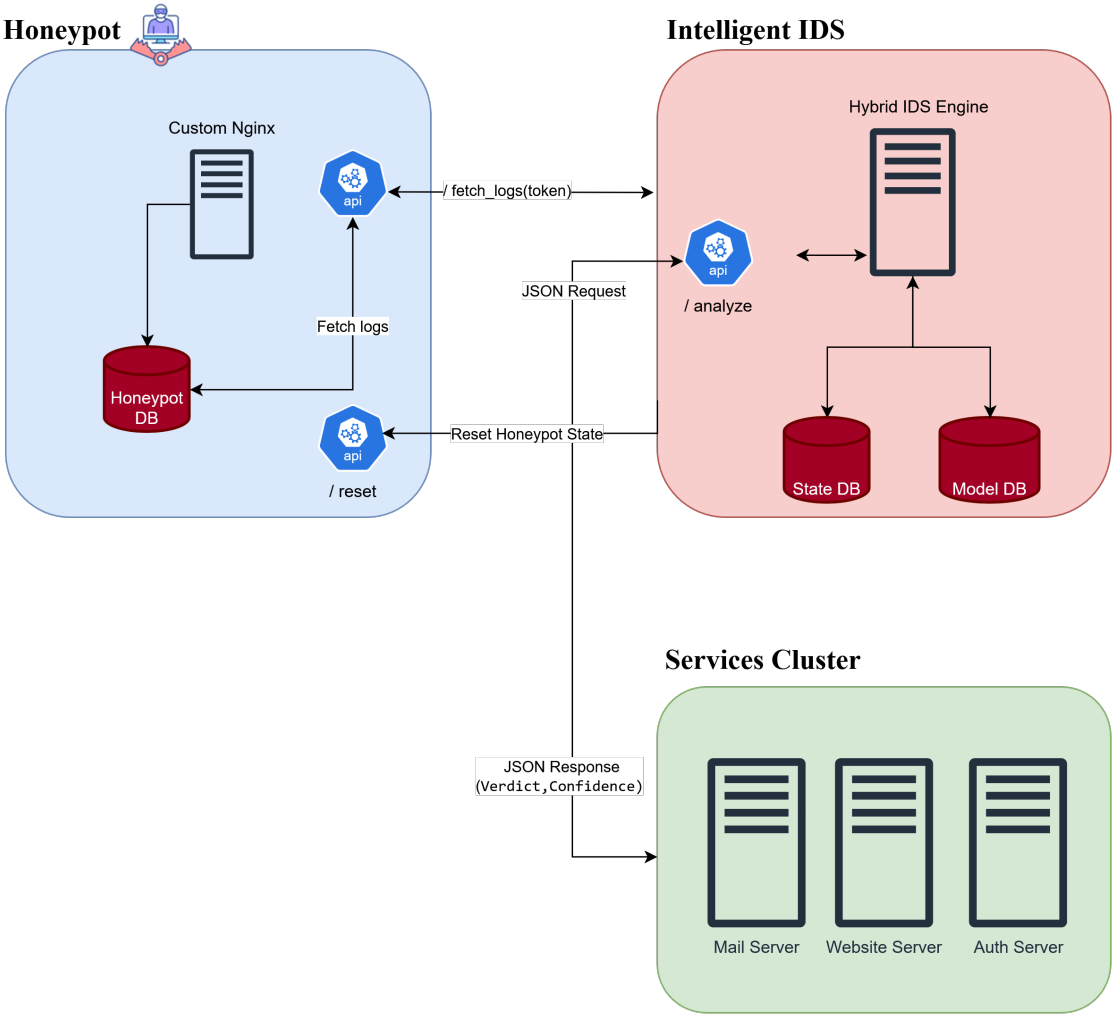


Figure 2: Overview system architecture of the proposed Platform

The flowchart in FIGURE 3, shows the general data movement and communication of system elements of the Honeypot-Driven Hybrid Continuous Learning Platform. The first step is the honeypot that logs in the incoming attack traffic to a database. The IDS server periodically retrieves new semi-supervised records via the secure API offered by the honeypot and they are used by the server to update the model in addition to the supervised learning process.

This hybrid learning process is a continuous improvement of the trained model. After training, the IDS opens an evaluation API which receives a client request to access. The model analyzes each request and returns a response in the form of a JSON with a verdict and a score. Depending on this level of confidence, the client systems are able to make the decision to accept or deny the request based on their predetermined level of security and as such will obtain a flexible and adaptive defense system.

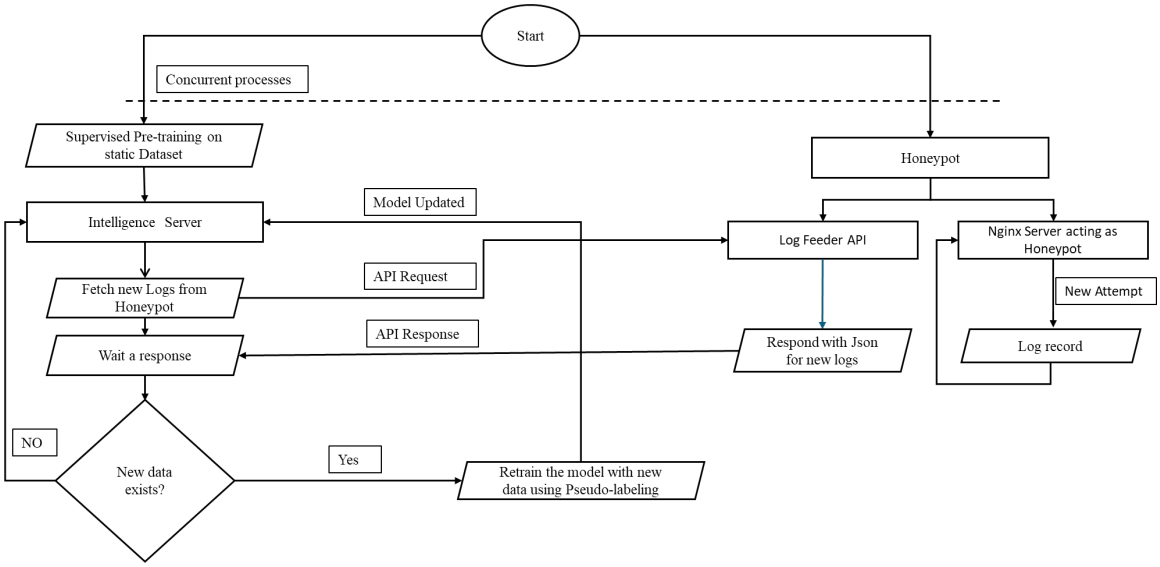


Figure 3: Flowchart for Honeypot-Driven Hybrid Continuous Learning Platform

# 7. RESULTS

## 7.1 Detection Performance

In this section we evaluate whether a honeypot driven incremental learning pipeline with semi-supervised updates can sustain or improve detection performance over time compared with an initial fully supervised warm up stage.

As shown in FIGURE 4 and FIGURE 5, the switch to the semi-supervised learning (Cycle 3 and further) indicates a definite stabilization of the performance. The AUC and F1-scores indicate that the model is able to consume unlabeled live traffic without degrading its performance. In particular, the uniform increase in the Accuracy is evidence that the confidence-based filtering is effective to select high-quality pseudo-labels to update the incremental model and it confirms that the platform is adaptable to live attack patterns.

### 7.1.1 Supervised warm up phase (cycles 1 to 7)

In the supervised warm up stage, the model is trained purely on labeled data. In FIGURE 4, the curves for AUC, accuracy, and F1 during cycles 1 to 7 fluctuate around moderate values. The mean AUC across these cycles is approximately 0.68, with individual values between 0.60 and 0.74. The accuracy curve remains in a narrow band around 0.61, and the F1 curve is similarly stable around 0.59.

FIGURE 5 shows that precision and recall are imbalanced during this phase. Precision stays relatively low, around 0.43–0.47, while recall is consistently high, with an average of about 0.89 and

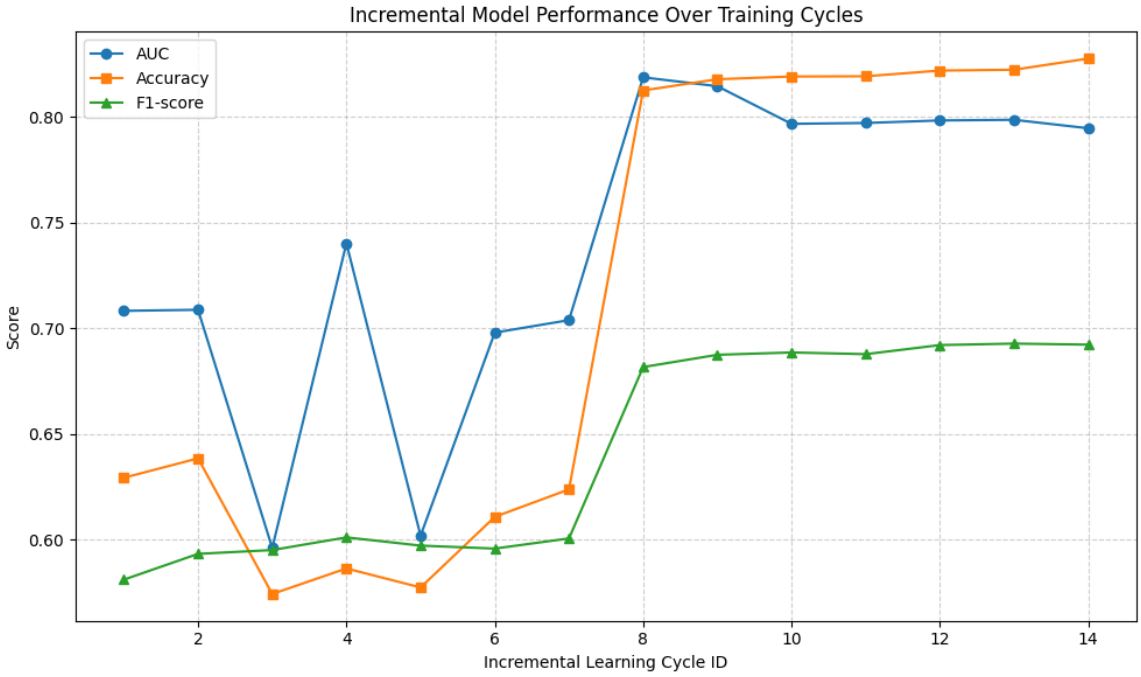


Figure 4: Incremental Model Performance Over Training Cycles

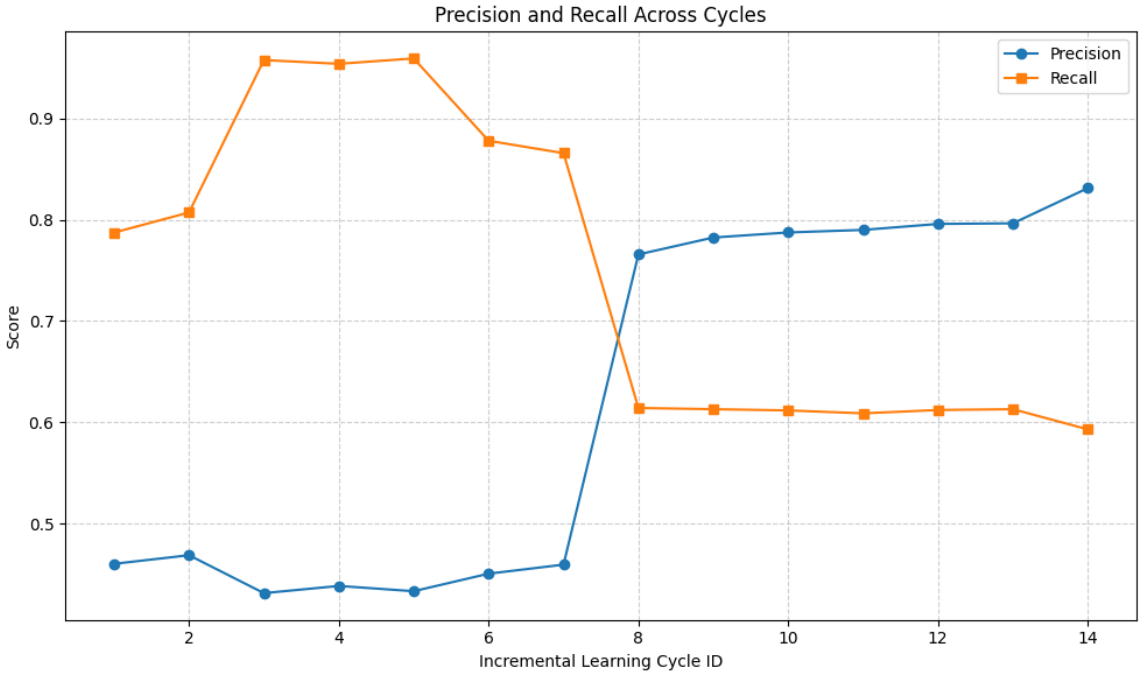


Figure 5: Precision and Recall Across Cycles

peaks close to 0.96. This shape of the curves confirms that the supervised warm up model operates in a recall oriented regime: it aggressively flags suspicious traffic and captures most attacks, but at the cost of many false positives and limited precision. This initial operating point is suitable as a starting detector that errs on the side of caution before exposing the model to real honeypot traffic.

#### 7.1.2 Semi-supervised incremental phase (cycles 8 to 14)

From cycle 8 onward, the system switches to the honeypot driven semi-supervised loop. The same performance curves in FIGURE 4, clearly show a transition to a higher and more stable regime once the semi-supervised cycles start. The AUC line rises to approximately 0.80 and remains in a narrow band between 0.79 and 0.82 across cycles 8 to 14. The accuracy curve shows a similar jump and stabilizes around 0.81–0.82. The F1 curve also increases and stabilizes around 0.68.

Compared with the supervised phase, the averages confirm this shift. Mean AUC increases from 0.68 in cycles 1–7 to about 0.80 in cycles 8–14 (roughly an 18 percent relative improvement). Mean accuracy rises from 0.61 to about 0.81 (around 34 percent relative improvement), and mean F1 increases from 0.59 to about 0.68 (about 15 percent relative improvement). At the same time, mean precision increases from about 0.45 to about 0.76, while mean recall decreases from about 0.89 to about 0.62, as visible in FIGURE 5. The precision and recall curves therefore cross and then separate into a more balanced region in the semi-supervised phase.

These trends indicate that the semi-supervised honeypot loop moves the classifier away from the highly aggressive recall oriented operating point and toward a more conservative, balanced point with fewer false positives. The model trades some recall in exchange for higher precision and overall discriminative power. The last semi-supervised cycles (for example cycle 14) still maintain high accuracy (around 0.82) and AUC (around 0.79) with F1 around 0.69, which suggests that the performance does not degrade over time and there is no sign of catastrophic forgetting in the plotted metrics.

### 7.2 Confidence behavior and training efficiency

The confidence focused plots further justify the use of semi-supervised updates. FIGURE 6 shows that average confidence for the supervised warm up cycles is zero, because confidences are not meaningful in that stage, then jumps above 0.91 once the model begins to process honeypot logs. During cycles 8 to 14, the average confidence per cycle stays between 0.91 and 0.99, and in some months (for example the large batch in cycle 14) it approaches 0.99. This trend indicates that as the model is exposed to more data, it becomes more certain about its decisions.

The same figure overlays the number of high confidence samples per cycle. During the semi-supervised phase the majority of records fall into the high confidence bucket. Aggregating across cycles 8 to 14, the model sees 167,840 honeypot records in total, of which 137,406 (about 82 percent) are high confidence, 24,421 (about 15 percent) are medium confidence, and only 6,013 (about 4 percent) are low confidence. These proportions match the bar heights in FIGURE 6, and support the claim that the confidence based filter is selective but not excessively restrictive.

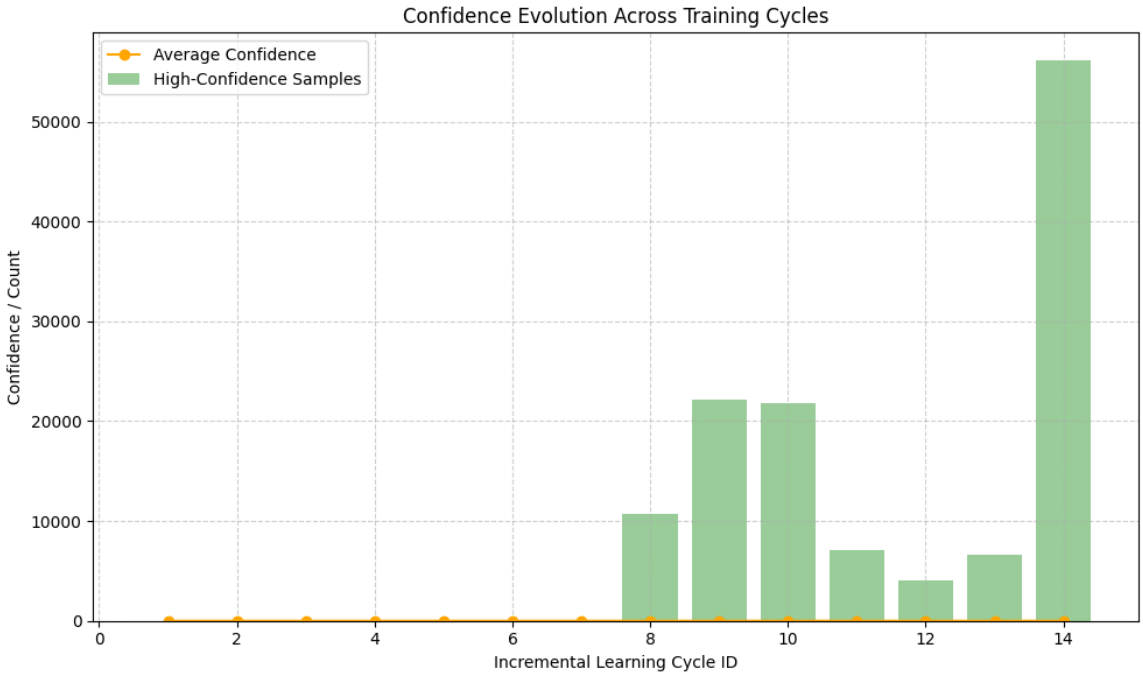


Figure 6: Confidence Evolution Across Training Cycles

FIGURE 7 shows that the number of records used for training closely tracks the number of records fetched during semi-supervised cycles, while the skipped curve remains much lower. Summing over cycles 8 to 14, 146,859 out of 167,840 records are actually used for training, which corresponds to about 87.5 percent of the available honeypot data. Only about 12.5 percent of records are skipped because their confidence is below the 0.8 threshold. This pattern demonstrates that the confidence filter is effective at discarding uncertain pseudo labels while still exploiting most of the available stream.

Lastly, FIGURE 8 shows the computation profile of the updates. This scatter distinctly separates the supervised cycles that cluster tightly around 2500 records with cycle duration of around 6–7 seconds and the semi supervised cycles which work on larger batches of around 6000 up to nearly 60000 records. The regression line shows that there is an almost linear correlation between the number of records and the cycle period. As an illustration, the last semi supervised cycle processes almost 58000 records within approximately 45 seconds. This, together with the AUC of 0.80 and accuracy of 0.82 above, confirms that the incremental pipeline is scale efficient in terms of batch size and practical for periodic updates driven by honeypots.

Taken together, the performance curves (FIGURE 4 and FIGURE 5), confidence and training behavior (FIGURE 6 and FIGURE 7), and the cycle duration scatter (FIGURE 8) provide a consistent picture. The honeypot-driven semi-supervised loop:

- Uses a high confidence threshold to select reliable pseudo labels from real traffic.

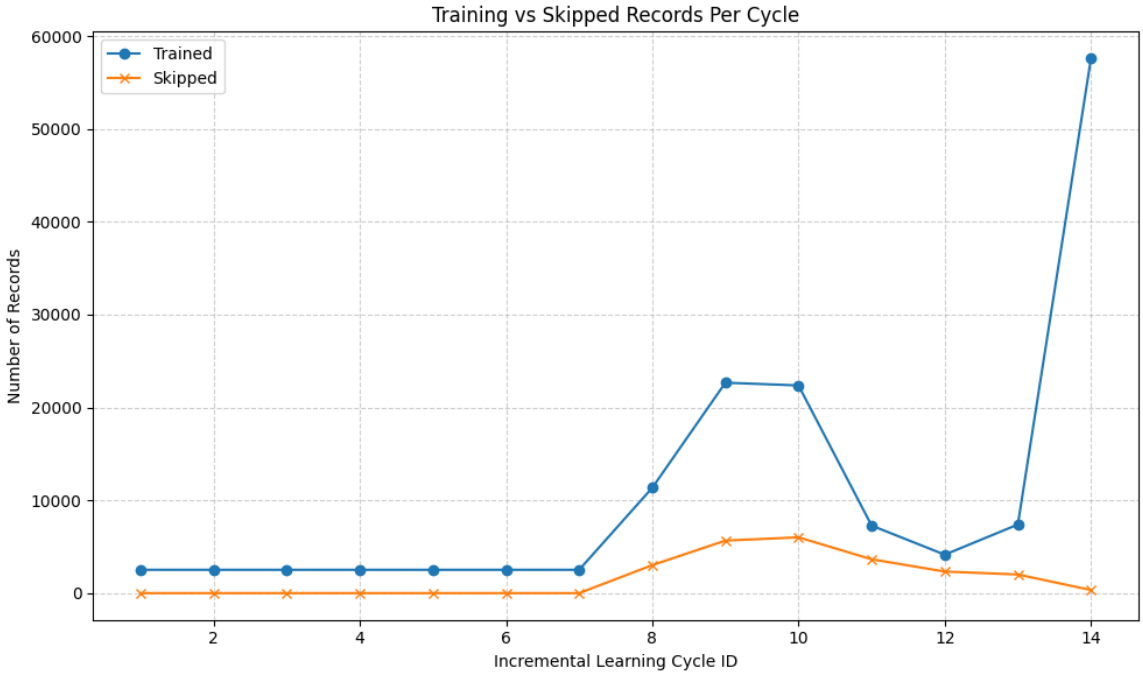


Figure 7: Training vs Skipped Records Per Cycle

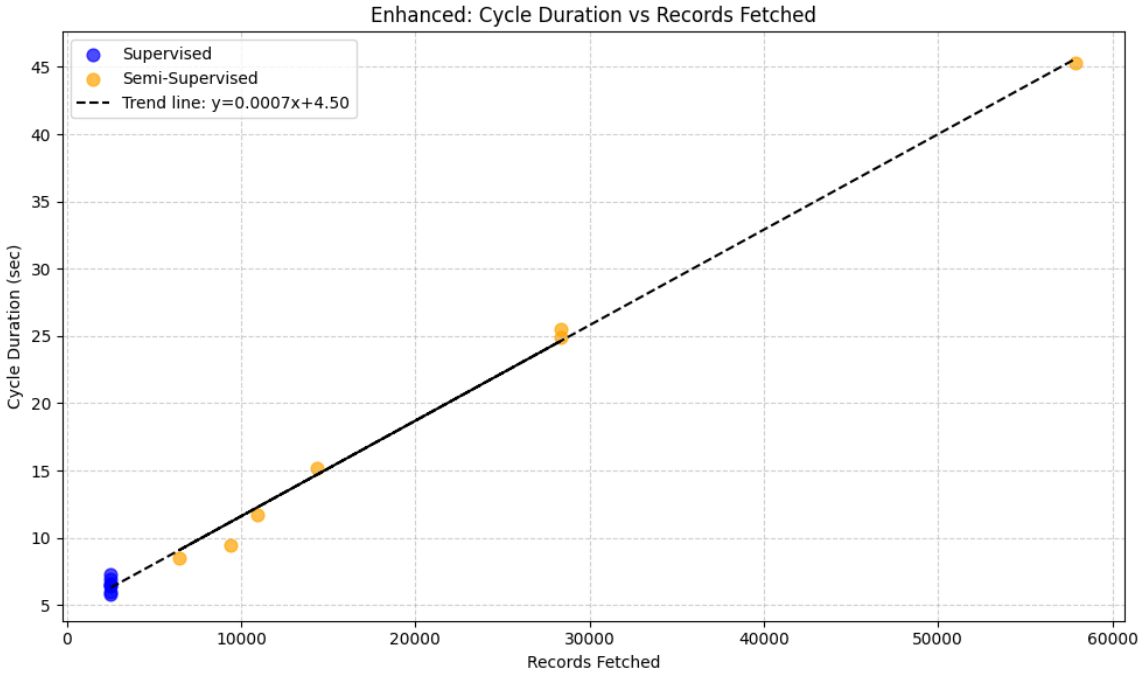


Figure 8: Enhanced Cycle Duration vs Records Fetched

- Ingests the majority of honeypot records while discarding only a small low confidence fraction.
- Shifts the classifier from a purely recall oriented supervised baseline to a more balanced operating point with higher AUC, accuracy, precision, and F1.
- Maintains stable performance across multiple months without visible degradation in the plotted metrics.
- Scales linearly in time with the amount of honeypot data processed.

These empirical observations directly support a positive answer to our honeypot driven proposal: integrating honeypot logs into a semi-supervised incremental learning platform is beneficial, since it allows the model to refine its decision boundary using real traffic while keeping both performance and computational cost under control.

### 7.3 Real World Deployment Proof of Concept (One Week Run)

We deployed the framework on two publicly accessible VPSs and kept it running for one week. During this period, we monitored both the honeypot collection and serving API, as well as the intrusion detection model, which operated on a separate VPS. In this section, we will showcase the learning cycles observed during the one week deployment.

#### 7.3.1 Honeypot VPS specs

- **Operating System**: Ubuntu 22.04 (64-bit)
- **Processor**: 1 x 2.6 GHz
- **Memory**: 2048 MB
- **Storage**: 20 GB
- **Bandwidth**: 100 Mbps (Monthly limit: 2 TB)

#### 7.3.2 Intelligence VPS specs

- **Operating System**: Ubuntu 20.04 (64-bit)
- **Processor**: 2 x 1.8 GHz
- **Memory**: 4096 MB
- **Storage**: 40 GB
- **Bandwidth**: 100 Mbps (Monthly limit: 4 TB)

To examine the behaviour of the system during the one week deployment the analysis focuses on learning cycles 15 to 21 which represent daily incremental updates. FIGURE 9 shows the number

of records fetched per day with values ranging from 57 to 262. This reflects normal day to day variation in real world attack traffic and confirms that the honeypot continued to supply new data throughout the deployment period.

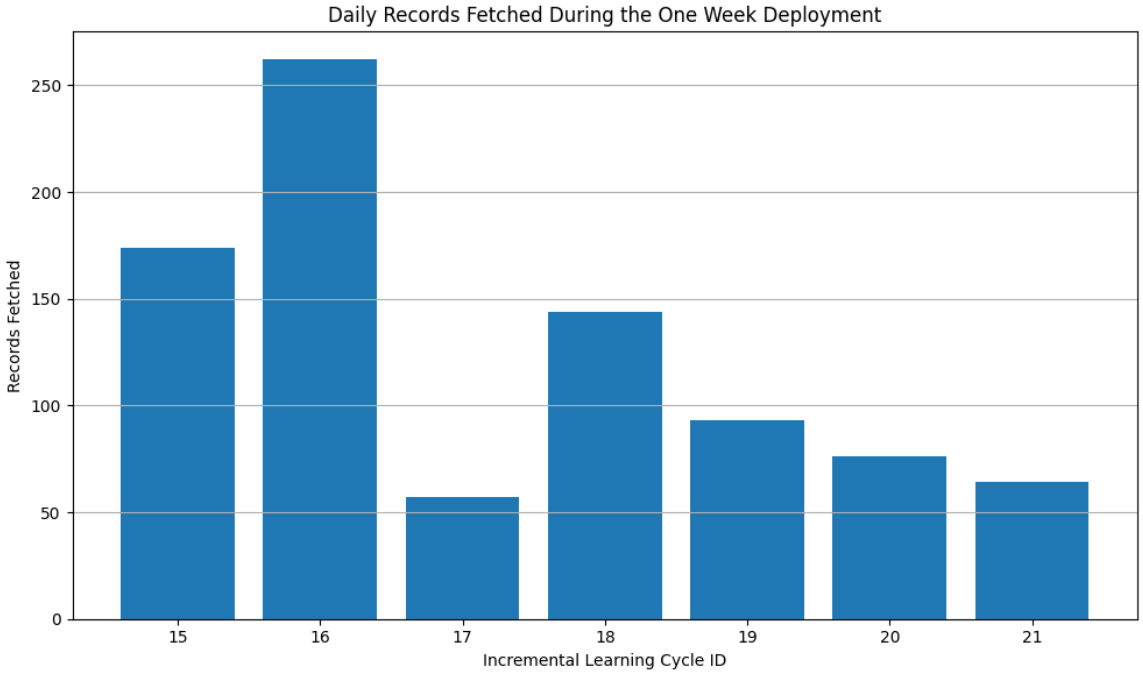


Figure 9: Daily Record Volatility: Total traffic volume captured by the Honeypot during the live VPS deployment phase

FIGURE 10 presents the model performance for the same daily cycles. The AUC accuracy precision recall and F1 curves remain stable across all seven days which indicates that the incremental updates did not introduce fluctuations in the model's behaviour. The stability of these metrics suggests that the model maintained consistent predictive performance even when trained on smaller daily batches.

FIGURE 11 displays the number of trained and skipped samples per cycle. The model trained on most of the incoming samples while skipping only those that fell below the confidence threshold. This shows that the confidence based filtering mechanism operated correctly and that the system was able to process and filter real world inputs without interruption. Combined these observations demonstrate that the platform functioned reliably in a live environment and handled daily incremental updates in a stable and consistent manner.

These findings show that the platform is capable of sustaining continuous incremental learning in a real environment while preserving consistent and reliable model behaviour.

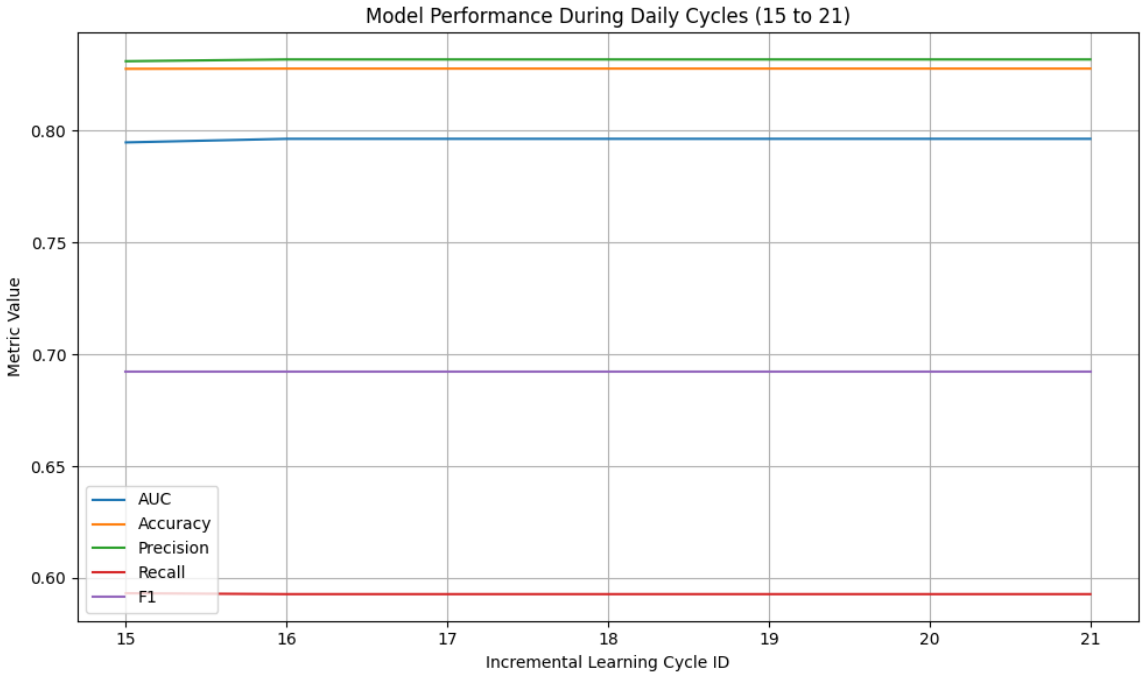


Figure 10: Model performance during daily cycles 15 to 21

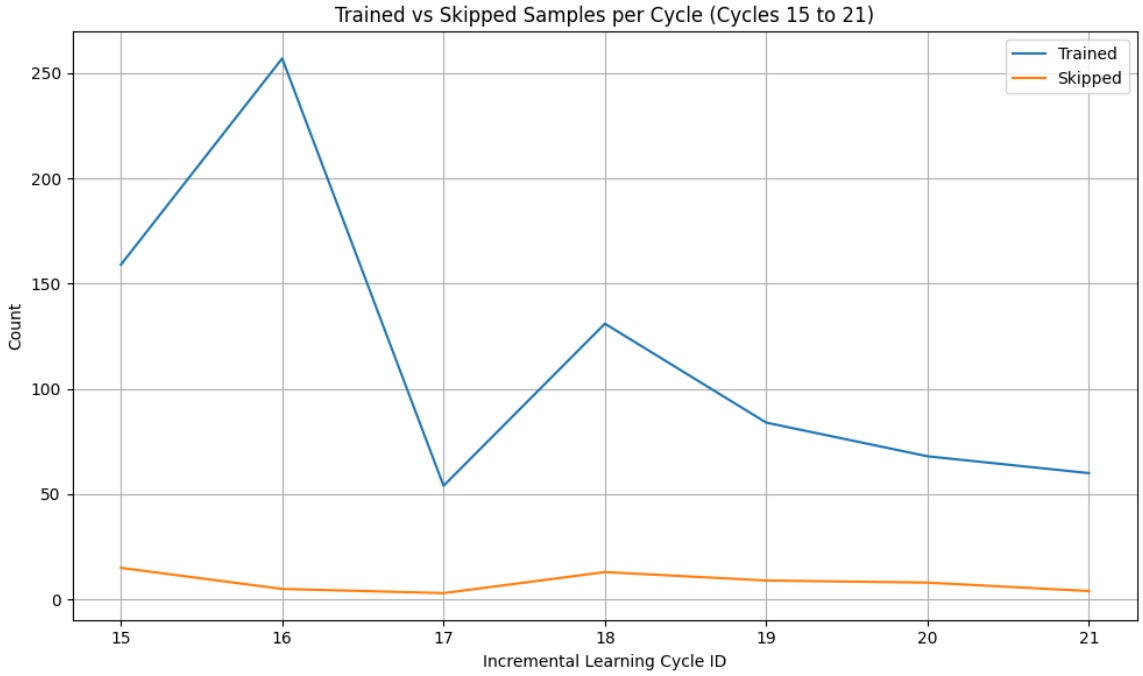


Figure 11: Trained and skipped samples per cycle

## 8. DISCUSSION

The experimental results show that combining an initial supervised warm up phase with a subsequent honeypot driven semi-supervised phase leads to a clear qualitative change in the behavior of the intrusion detection model. The plots of AUC, accuracy, and F1 across the seventeen cycles indicate that the supervised cycles establish a reasonable baseline, while the semi-supervised cycles on honeypot traffic move the classifier into a more stable and higher performing regime. After the transition to semi-supervised updates, AUC and accuracy increase and then fluctuate within a narrow band, which suggests that the model adapts to the incoming traffic without exhibiting performance drift or catastrophic forgetting.

The precision and recall curves provide additional insight into the nature of this adaptation. During the supervised phase, the model operates with very high recall and comparatively low precision, which reflects an aggressive decision boundary that favors detecting as many attacks as possible at the cost of frequent false alarms. Once honeypot driven semi-supervised updates are introduced, precision increases substantially while recall decreases to a more moderate level. This shift corresponds to a more conservative and balanced operating point. For a continuously running detection platform that must be integrated into operational workflows, such a shift is often desirable, because higher precision and stable AUC reduce alert overload and increase the trustworthiness of alarms.

The confidence related plots further support the semi-supervised design. In the incremental phase, the average prediction confidence per cycle is consistently high, and most samples fall into the high confidence bucket. At the same time, the trained versus skipped curves show that the confidence threshold of 0.8 filters out only a modest fraction of the stream while still removing low confidence pseudo labels. This indicates that the platform is able to exploit real honeypot data effectively without heavily contaminating the model with uncertain labels, which is one of the main risks in semi-supervised learning settings.

The relationship between cycle duration and the number of processed records also has practical significance. The scatter plot with a regression line shows an approximately linear trend between batch size and update time, with absolute durations remaining low even for the largest monthly honeypot batches. This behavior suggests that the incremental update procedure scales in a predictable way and can be scheduled regularly, for example at monthly intervals, without imposing a high computational cost. Such properties are important for deployment in environments where updates must be frequent but resources are limited.

Overall, the combination of performance trends, confidence distributions and efficiency measurements provides consistent evidence that a honeypot driven semi-supervised incremental learning pipeline can maintain stable detection quality over time, reduce the rate of false positives compared to a purely supervised baseline, and remain computationally efficient when processing realistic volumes of web traffic.

### 8.1 Security and Adversarial Robustness:

Poisoning attacks are a risk that is critical to continuous learning. This platform prevents such risks with its mechanism of Confidence-Based Filtering. Using a large threshold (e.g. 0.8) allows the

system to ignore low-confidence noise or adversarial samples which could otherwise seek to bias the model decision boundary. This guarantees that it only learns the high-certainty attack patterns giving in-built defense against evasion attacks in case of live deployment.

## 9. CONCLUSIONS AND FUTURE DIRECTIONS

The findings in this paper reveal that a combination of a guided warm up process and a semi-supervised incremental learning process that is powered by honeypots is an effective and feasible method of constructing adaptive intrusion detection systems. The semi-supervised cycles result in more balanced and steady performance profile with greater AUC, accuracy, precision, and F1 scores, whereas the supervised cycles result in a recall oriented baseline.

Confidence based filtering guarantees the use of trustworthy pseudo labels in incremental updates to stabilize behavior of models and minimize the possibility of catastrophic forgetting. The analysis of the cycle duration also reveals that the incremental updates scale consistently predictably as the amount of data in the honeypots grows and so the strategy can be easily deployed on a regular basis, as required by automation. All in all, the results validate the hypothesis that a hybrid continuous learning framework is capable of addressing the weaknesses of a static IDS model and providing long term benefits in defensive capabilities.

There are a number of opportunities of expanding and enhancing the suggested platform:

- Check other online learning models Future research can test more online learning models, including Hoeffding Trees, variants of logistic regression, or online ensemble approaches, and establish whether they offer better resilience or detection.
- Thinking and reacting in response to changing traffic conditions or concept drift: A fixed confidence threshold might not be the best, and instead, adaptive confidence thresholding could be considered to achieve a good balance between precision and recall.
- More sophisticated semi-supervised and active learning approaches: The usage of label propagation, uncertainty based self training, or active querying could improve the quality of pseudo labels and decrease false positives even further.

## 10. AVAILABILITY OF DATA AND MATERIALS

The dataset used in this study was obtained from [11].

## 11. COMPETING INTERESTS

The authors declare that they have no competing interests.

## 12. FUNDING

The project was funded by KAU Endowment (WAQF) at king Abdulaziz University, Jeddah, Saudi Arabia. The authors, therefore, acknowledge with thanks WAQF and the Deanship of Scientific Research (DSR) for technical and financial support.

## 13. ACKNOWLEDGEMENT

This study partially contributes to the author's thesis research at King Abdulaziz University, specifically addressing one of the core research questions.

## References

[1] Huang L, Zhu Q. Adaptive Honeypot Engagement Through Reinforcement Learning of Semi-Markov Decision Processes. In International conference on decision and game theory for security. Cham: Springer. 2019:196-216.

[2] Lanka P, Gupta K, Varol C. Intelligent Threat Detection—AI Driven Analysis of Honeypot Data to Counter Cyber Threats. Electronics. 2024;13:2465.

[3] Al-Ruwaili RH, Ouda OM. A Hybrid Classification Approach of Network Attacks Using Supervised and Unsupervised Learning. Int J Adv Comput Sci Appl. 2023;14.

[4] Yang X, Yuan J, Yang H, Kong Y, Zhang H, et al. A Highly Interactive Honeypot-Based Approach to Network Threat Management. Future Internet. 2023;15:127.

[5] Dowling S, Schukat M, Barrett E. New Framework for Adaptive and Agile Honeypots. ETRI J. 2020;42:965-975.

[6] Ahmed Y, Beyioku K, Yousefi M. Securing Smart Cities Through Machine Learning: A Honeypot-Driven Approach to Attack Detection in Internet of Things Ecosystems. IET Smart Cities. 2024;6:180-198.

[7] Firmansyah DA, Zahra A. Honeypot-Based Thread Detection Using Machine Learning Techniques. Int J Eng Trends Tech. 2023;71:243-252.

[8] Morozov DS, Yefimenko AA, Nikitchuk TM, Kolomiiets RO, Semerikov SO. The Sweet Taste of IoT Deception: An Adaptive Honeypot Framework for Design and Evaluation. J Edge Comp. 2024;3:207-223.

[9] Siddique MR, Hussain MZ, Hasan MZ, Nosheen S, Qureshi AM, et al. Integrating Machine Learning-Powered Smart Agents Into Cyber Honeypots: Enhancing Security Frameworks. In 2024 IEEE 9th International Conference for Convergence in Technology. I2CT. IEEE. 2024:1-7.

[10] Solomon YT, Melese Z. Dynamic Interactive Honeypot for Web Application Security. International Journal of Wireless and Microwave Technologies. 2024;14: 1-14.

[11] Cahyanto KA, Al Hilmi MA, Mustamiin M. Pengujian Rule-Based Pada Dataset Log Server Menggunakan Support Vector Machine Berbasis Linear Discriminat Analysis Untuk Deteksi Malicious Activity. Jurnal Teknologi Informasi dan Ilmu Komputer. 2022;9:245-254.

[12] Montiel J, Halford M, Mastelini SM, Bolmier G, Sourty R, et al. River: Machine Learning for Streaming Data in Python. J Mach Learn Res. 2021;22:1-8.

[13] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, et al. Scikit-Learn: Machine Learning in Python. J Mach Learn Res. 2011;12:2825-2830.

[14] LemaÃŽtre G, Nogueira F, Aridas CK. Imbalanced-Learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. J Mach Learn Res. 2017;18:1-5.