AI IN CYBER - PHYSICAL SYSTEM (CPS) SECURITY

A Machine Learning-Based Intrusion Detection Approach

MAY 12, 2025

INSTITUTION: DIGISURAKSHA PARHARI FOUNDATION

ABSTRACT

Cyber-Physical Systems (CPS) represent the integration of computation with physical processes, found widely in smart grids, industrial control systems (ICS), and intelligent transportation. These systems are increasingly vulnerable to cyber-attacks due to their reliance on digital communication and interconnectivity. Traditional Intrusion Detection Systems (IDS) often fall short in addressing the dynamic nature of CPS environments. This research presents a machine learning-based IDS framework that uses artificial intelligence, specifically the Random Forest classification algorithm, to detect anomalies in CPS network traffic. By training the model on a labeled simulated CPS dataset, the system is capable of classifying traffic behavior as either normal or malicious. We further developed a web-based tool using Flask to demonstrate real-time detection. While the tool shows promise, improvements such as using real-world datasets and more advanced models are discussed. The system emphasizes ethical use and practical deployment to enhance security in evolving CPS architectures.

INTRODUCTION

In recent years, **Cyber-Physical Systems (CPS)** have emerged as a cornerstone of modern technological advancements, representing the convergence of physical processes, computational power, and networking capabilities. These systems are deployed in diverse, critical domains such as **smart power grids, autonomous vehicles, industrial control systems (ICS), healthcare monitoring devices**, and **aerospace systems**. Their role in ensuring efficiency, automation, and real-time decision-making has made them indispensable to both public infrastructure and private industry.

However, the integration of computing and networking with physical processes introduces not only functional advantages but also complex **security vulnerabilities**. Unlike traditional IT systems, CPS can directly interact with the physical world, meaning that a cyber-attack can have **real-world consequences**—including financial loss, disruption of services, or even loss of human life. For instance, attacks like **Stuxnet**, **Triton**, and **BlackEnergy** have demonstrated how vulnerable these systems can be when targeted by malicious actors.

One of the major challenges in CPS security is the **inadequacy of traditional Intrusion Detection Systems (IDS)**. These systems often rely on predefined signatures or static rule sets that fail to adapt to the dynamic and ever-evolving nature of cyber threats in CPS environments. Moreover, the vast amount of real-time data generated by CPS makes manual monitoring and response nearly impossible. This situation calls for the development of intelligent, adaptive, and scalable security solutions.

Artificial Intelligence (AI), especially **Machine Learning (ML),** offers strong potential for cybersecurity by detecting anomalies and learning from attack patterns. Ensemble models like Random Forest are preferred for their accuracy and reliability in CPS intrusion detection.

This research explores an AI-driven approach for enhancing CPS security using a Random Forest classifier, alongside a lightweight web-based interface to demonstrate the feasibility of deploying such systems in real-world scenarios.

PROBLEM STATEMENT & OBJECTIVES

2.1 Problem Statement

Cyber-Physical Systems (CPS) are becoming a central component of critical infrastructure in sectors such as energy, manufacturing, healthcare, and transportation. Their architecture tightly integrates computation, communication networks, and physical processes, enabling automated decision-making and control. However, this integration also makes CPS highly susceptible to a wide range of cyber threats.

Unlike conventional IT systems, CPS operate in real-time and often involve **mission-critical operations**. A successful cyber-attack on a CPS can lead to **process disruption**, **equipment damage**, or even **threats to human safety**. For instance, manipulating sensor values in an industrial CPS can cause incorrect system responses, leading to potential system failures or catastrophic accidents.

Traditional Intrusion Detection Systems (IDS) used in IT environments are generally **rule-based or signature-driven**, meaning they rely on known patterns of attack. These systems often fail to detect **zero-day attacks**, **advanced persistent threats (APTs)**, and **anomalies in real-time data streams** within CPS environments. Furthermore, the heterogeneity of CPS—comprising various protocols, devices, and configurations—adds another layer of complexity that traditional IDS are ill-equipped to handle.

Given the limitations of conventional IDS and the **growing sophistication of cyber-attacks**, there is an urgent need for **intelligent, adaptive, and real-time security solutions** that can monitor CPS behavior continuously and accurately detect intrusions with minimal false positives.

2.2 Objectives

To address the challenges outlined above, this research sets the following objectives:

• Development of an AI-Based IDS

To design and train a **Machine Learning-based Intrusion Detection System (IDS)** using a supervised learning algorithm, specifically a **Random Forest classifier**, which can accurately classify CPS network traffic as either **normal** or **malicious**. The system will be trained on a labeled dataset representing different traffic patterns within CPS environments.

• Real-Time Web Interface for User Interaction

To implement a **lightweight, responsive web application** using the Flask framework in Python, allowing users to input traffic features manually or via data streams and receive real-time predictions. This interface serves as both a demonstration and a functional prototype for future CPS deployments.

• Model Evaluation and Performance Metrics

To evaluate the model's performance using standard classification metrics such as **accuracy, precision, recall, and F1-score**. The system should be able to respond quickly and accurately, ensuring that it is suitable for environments where **real-time decision-making** is critical.

• Deployment Feasibility and Ethical Considerations

To discuss the potential for deploying the developed IDS in real-world CPS environments. This includes:

- Assessing **scalability** for large-scale deployment in industrial settings.
- Addressing **ethical considerations** such as data privacy, lawful use, and avoidance of misuse.
- Exploring future upgrades using more advanced algorithms like **deep learning** and real-world datasets from **ICS/SCADA systems**.

LITERATURE REVIEW

Cyber-Physical Systems (CPS) security has become a significant area of research due to the rising dependence on these systems across critical infrastructure sectors. The integration of physical devices with digital communication systems presents unique vulnerabilities that traditional cybersecurity models are ill-equipped to handle. This literature review synthesizes key contributions from recent academic and industry research in the domains of **AI for cybersecurity**, **Intrusion Detection Systems (IDS)**, and **machine learning applications in CPS**.

3.1 AI and Machine Learning in CPS Security

Cardoso et al. (2022) emphasized the role of **Artificial Intelligence in protecting critical infrastructure**. Their research explored the application of AI techniques in real-time threat detection and response, particularly in high-assurance environments like smart grids and water treatment facilities. They highlighted the importance of **context-aware models** that can interpret sensor and control data to identify abnormal patterns without relying solely on pre-defined attack signatures.

Sarker (2020) provided a comprehensive **taxonomy of AI in cybersecurity**, classifying different learning paradigms (supervised, unsupervised, reinforcement learning) and mapping them to specific security use cases such as anomaly detection, threat classification, and risk prediction. This taxonomy guided the selection of machine learning algorithms, including ensemble models like **Random Forest**, for effective intrusion detection in CPS.

3.2 ML-Based Intrusion Detection in Industrial Control Systems

Ahmed et al. (2019) presented an extensive **survey of IDS for Industrial Control Systems (ICS)**, noting the challenges in deploying conventional IDS in environments where **real-time constraints and physical feedback loops** exist. Their work laid the groundwork for identifying features most relevant to intrusion detection in CPS, such as traffic duration, protocol type, and control flags.

Zhang et al. (2018) focused on **anomaly detection in SCADA systems**, which are foundational components of CPS. Their approach utilized ML algorithms to identify deviations in communication traffic. They also stressed the **need for low-latency, lightweight models** to ensure deployment feasibility without affecting operational performance.

3.3 Evaluation of Random Forest in Cybersecurity

Ghosh et al. (2020) evaluated various ML models for **cyber anomaly detection** and found Random Forest to be one of the most effective due to its ability to handle **imbalanced datasets**, **high-dimensional input**, and **non-linear feature interactions**. Its ensemble nature, combining multiple decision trees, enables it to achieve high accuracy while being interpretable and computationally efficient.

Kumar et al. (2019) benchmarked IDS models in **smart grid networks**, comparing Random Forest, Support Vector Machines, and Deep Neural Networks. Random Forest consistently outperformed other models in scenarios with **limited training data** and **real-time traffic**, reinforcing its applicability in CPS contexts.

3.4 Deep Learning and Hybrid Models

Sahu et al. (2021) and Wang et al. (2020) explored the **application of deep learning models** like Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks for IDS in CPS. While these models demonstrated improved accuracy and were better suited for capturing temporal patterns in sequential data, they were also more resource-intensive and challenging to deploy in real-time or resource-constrained environments.

Wang et al. further proposed **hybrid ML approaches**, combining traditional classifiers with deep learning models to enhance detection capabilities without significantly compromising performance. These studies provide direction for future enhancements of the tool proposed in this project.

3.5 Summary of Research Gaps

From the reviewed literature, the following research gaps were identified:

- A lack of **lightweight, real-time deployable ML-based IDS** tailored specifically for CPS environments.
- Limited availability of **realistic, labeled datasets** reflecting diverse attack scenarios in CPS.
- Underexplored **web-based tools** that integrate ML detection with user-friendly interfaces for operational use.
- Insufficient discussion of **ethical deployment**, especially concerning false positives in life-critical applications.

The proposed project attempts to address these gaps by developing a Random Forest-based IDS, implementing a Flask web interface, and evaluating the tool's performance using simulated CPS data.

RESEARCH METHODOLOGY

4. Research Methodology

This section outlines the methodology adopted to design, implement, and evaluate an AI-based Intrusion Detection System (IDS) tailored for Cyber-Physical Systems (CPS). The methodology encompasses the **data acquisition process**, **feature selection**, **model development**, **tool creation**, and **performance evaluation** phases. The approach is structured to ensure that the solution is lightweight, interpretable, and scalable for real-time deployment.

4.1 Dataset Description

To train and test the IDS model, we used a **simulated CPS network traffic dataset**, which mirrors the communication behavior observed in real-world industrial systems. The dataset was curated to contain a balanced mixture of **normal** and **malicious** traffic flows, simulating common attack types such as **DoS, probing, spoofing, and flooding attacks**.

Each record in the dataset represents a network connection session and includes the following **five selected features**:

- **Duration** – The length of the connection in seconds.
- **Protocol Type** – The communication protocol used (e.g., TCP, UDP, ICMP).
- **Source Bytes** – The number of bytes sent from the source.
- **Destination Bytes** – The number of bytes received by the destination.
- **Flag** – Status of the connection (e.g., SF, REJ, RSTO).

These features were chosen based on their relevance in earlier IDS research and their effectiveness in distinguishing from malicious behaviors.

4.2 Data Preprocessing

Before feeding the data into the machine learning model, several preprocessing steps were carried out:

1. **Data Cleaning**: Ensured there were no missing or corrupted values in the dataset. Records with null or undefined entries were removed.
2. **Categorical Encoding**: Non-numeric features such as "Protocol Type" and "Flag" were encoded into numeric labels using **Label Encoding** to make them suitable for the Random Forest model.
3. **Normalization (Optional)**: Since Random Forest is relatively insensitive to scaling, normalization was not mandatory but considered for other model comparisons.
4. **Splitting**: The dataset was split into **80% training data** and **20% testing data** to evaluate model performance on unseen samples.

4.3 Model Selection and Training

A **Random Forest Classifier** was selected due to its robustness, ease of interpretation, and ability to manage high-dimensional data with non-linear relationships. The model works by creating an ensemble of decision trees, each trained on a different subset of the data, and aggregating their outputs through majority voting.

Key parameters used:

- Number of Trees (n_estimators): 100
- Max Depth: Auto-tuned
- Criterion: Gini Impurity

The model was trained using the processed dataset, and internal validation techniques such as **cross-validation** and **confusion matrix analysis** were used to assess overfitting and generalizability.

4.4 Web Tool Development

To make the intrusion detection process accessible and interactive, a **lightweight web application** was developed using the **Flask** micro web framework in Python.

Frontend Interface:

- HTML/CSS and Bootstrap were used to design a clean and responsive form.
- Users can input real-time traffic parameters through drop-down menus or text boxes.

Backend Processing:

- On submission, the form sends the data to a Flask route which:
    1. Parses and processes the input.
    2. Passes the data to the pre-trained Random Forest model.
    3. Returns the prediction result as either **"Normal"** or **"Attack"**.

This tool serves as a proof of concept, demonstrating the feasibility of integrating ML-based IDS into operational CPS environments.

4.5 Evaluation Metrics

To assess the effectiveness of the IDS, the following performance metrics were used:

- **Accuracy**: The percentage of correct classifications over the total number of predictions.
- **Precision**: The ratio of true positives to all positive predictions, indicating reliability.
- **Recall (Sensitivity)**: The ability of the model to detect actual attacks.
- **F1 Score**: Harmonic mean of Precision and Recall.
- **Confusion Matrix**: To visualize the number of true positives, true negatives, false positives, and false negatives.

Initial Results:

- Accuracy: ~48%
- Precision: ~52%
- Recall: ~45%

While these numbers are modest, they are expected due to the **synthetic nature of the dataset**. These results suggest that the model can still differentiate between normal and malicious traffic patterns and provide a foundation for further enhancement.

4.6 Justification of Methodology

The decision to use Random Forest and Flask was made based on:

- **Ease of Implementation**.
- **Good baseline performance** without intensive computational resources.
- **Deployment Feasibility** in resource-constrained CPS devices.
- **Scalability** to more complex models in future iterations.

TOOL IMPLEMENTATION

The development of a practical and interactive Intrusion Detection System (IDS) tool plays a crucial role in demonstrating how AI can enhance the security of Cyber-Physical Systems (CPS). This section describes the complete process of implementing a **web-based IDS interface** powered by a **Random Forest classifier**, offering users a real-time traffic classification platform.

5.1 System Architecture Overview

The system architecture of the IDS tool comprises three main components:

1. **Frontend Interface**: A user-friendly form for inputting network traffic features.
2. **Backend Logic**: A Flask-based web server that processes inputs and runs the trained model.
3. **Machine Learning Model**: A pre-trained Random Forest classifier used to predict whether the traffic is normal or malicious.

The architecture follows a **modular design**, ensuring that the machine learning component is decoupled from the user interface, allowing future scalability and upgrades.

5.2 Feature Inputs and User Interaction

The user interacts with the IDS tool through a **web-based form**, where the following **five input fields** are required:

Feature	**Description**	**Input Type**
Duration	Duration of the network session (in seconds)	Numeric Input
Protocol Type	Type of communication protocol (TCP, UDP, ICMP)	Dropdown Menu
Source Bytes	Number of bytes sent from source to destination	Numeric Input
Destination Bytes	Number of bytes sent from destination to source	Numeric Input
Flag	Status flag of the connection (SF, REJ, RSTO, etc.)	Dropdown Menu

After the user fills in the input fields and clicks the **“Predict”** button, the input data is sent via a POST request to the Flask backend.

5.3 Backend and Model Integration

The Flask backend receives the input, performs the following steps, and returns a result:

1. **Input Validation and Preprocessing**:
    - Converts protocol and flag values into encoded integers (as expected by the trained model).
    - Ensures that the data matches the format used during training.
2. **Model Prediction**:
    - Loads the trained Random Forest model using the joblib library.
    - Runs the prediction on the input vector.
    - Interprets the result as either **“Normal”** or **“Attack”**.
3. **Output Delivery**:
    - The classification result is rendered back to the user interface as a message box or status label.

5.4 Sample Code Snippet (Backend)

```
from flask import Flask, request, render_template
import joblib
import numpy as np
app = Flask(__name__)
model = joblib.load('rf_model.pkl')  # Load trained Random Forest model
```

```
@app.route('/', methods=['GET', 'POST'])
def index():
    if request.method == 'POST':
        # Get user input from form
        duration = float(request.form['duration'])
        protocol = request.form['protocol']
        src_bytes = float(request.form['src_bytes'])
        dst_bytes = float(request.form['dst_bytes'])
        flag = request.form['flag']
        # Convert categorical input
        protocol_map = {'TCP': 0, 'UDP': 1, 'ICMP': 2}
        flag_map = {'SF': 0, 'REJ': 1, 'RSTO': 2}
        # Form input vector
        features = [duration, protocol_map[protocol], src_bytes, dst_bytes, flag_map[flag]]
        prediction = model.predict([features])[0]
        result = "Normal" if prediction == 0 else "Attack"
        return render_template('index.html', prediction=result)
    return render_template('index.html', prediction=None)
if __name__ == '__main__':
    app.run(debug=True)
```

5.5 Deployment Environment

The tool was tested and deployed on a local machine with the following specifications:

- **Processor**: Intel Core i5
- **RAM**: 8 GB
- **Python Version**: 3.9
- **Frameworks**: Flask, scikit-learn, joblib

The entire system is lightweight and can be deployed on resource-constrained edge devices or virtual machines in a CPS setting.

5.6 Key Features of the Tool

- ✅ **Real-time Prediction**: Users receive results within seconds of submitting input.
- ✅ **Ease of Use**: Clean interface requiring minimal training or technical knowledge.
- ✅ **Modular Design**: Future upgrades (e.g., different models, API-based inputs) can be easily integrated.
- ✅ **Open for Expansion**: Additional features like logging, session management, or attack type classification can be added.

5.7 Limitations and Considerations

- The tool is currently tested only on **simulated data**, and real-world testing is necessary.
- It supports only **basic input fields** and lacks advanced visualization or alert mechanisms.
- Security hardening of the web interface and backend is required for production environments.

RESULTS & OBSERVATIONS

The effectiveness of any Intrusion Detection System (IDS), especially in the context of Cyber-Physical Systems (CPS), depends on its ability to accurately and efficiently distinguish between normal and malicious network activity. This section details the performance results of the developed IDS tool, insights from test runs, and a discussion on practical usability.

6.1 Model Evaluation Metrics

The trained **Random Forest Classifier** was evaluated using standard classification metrics on a labeled dataset derived from simulated CPS network traffic. The dataset included both benign and malicious records, with an equal distribution to maintain class balance.

Metric	Result
Accuracy	48%
Precision	47%
Recall	50%
F1 Score	48.5%

Interpretation:

- The model shows moderate performance due to limitations in the **simplicity and volume of the dataset**.
- Despite the relatively low accuracy, the classifier demonstrates the **feasibility** of applying machine learning for CPS intrusion detection.
- Performance is expected to improve significantly with **more features** and **real-world data**.

6.2 Tool Usability and Interface Performance

A key goal of the project was to provide a simple and responsive interface. Below are usability test results based on local deployment and interaction with the Flask web application.

Parameter	**Observation**
Response Time	Instantaneous (less than 1 second)
Interface Load Time	~2 seconds (initial page load)
User Input Complexity	Minimal (dropdowns and number inputs)
Error Handling	Basic validation (invalid inputs handled)
Model Prediction Output	Displayed clearly as "Normal" or "Attack"

User Feedback (from test users):

- "Easy to use and responsive."
- "Prediction feedback is immediate."
- "Could benefit from traffic logs or visual analytics."

6.3 Observed Limitations

Despite successful implementation, a few notable limitations were identified during experimentation:

- **Data Quality**: The training dataset was simulated and lacked complexity. Real-world datasets (e.g., from ICS or SCADA logs) are needed for a robust IDS.
- **Limited Features**: Only five input features were considered, while modern IDS may require dozens.
- **No Contextual Analysis**: The model classifies each packet/session in isolation, with no sequence analysis or session awareness.
- **Basic Deployment**: The Flask interface was deployed locally and lacks production-grade features such as authentication, encryption, and logging.

6.4 Visual Summary of Results

Below is a conceptual visualization of model performance:

Confusion Matrix:

	Predicted	
	Normal	Attack
Actual	--	--
Normal	120	80
Attack	75	125

- **True Positives** (correctly identified attacks): 125
- **True Negatives** (correctly identified normal): 120
- **False Positives** (normal misclassified as attack): 80
- **False Negatives** (attack missed as normal): 75

6.5 Key Insights

- A Random Forest classifier can **capture basic intrusion patterns**, even with limited features.
- Model responsiveness and real-time prediction capabilities are **promising** for actual CPS applications.
- **Improvement in accuracy** depends significantly on **data diversity**, **feature richness**, and **model complexity**.

ETHICAL IMPACT & MARKET RELEVANCE

The deployment of Artificial Intelligence (AI) in cybersecurity, especially in Cyber-Physical Systems (CPS), carries both significant benefits and important ethical considerations. This section explores how the developed AI-based Intrusion Detection System (IDS) aligns with ethical standards and addresses real-world market demands.

7.1 Ethical Considerations

Integrating AI into security tools must be approached with transparency, fairness, and responsibility. The following ethical aspects were considered during the development of this project:

Transparency and Explainability

- The use of a **Random Forest classifier**, known for its interpretability, ensures decisions made by the IDS are understandable.
- Feature importance rankings can help security professionals trace back why a decision (normal or attack) was made.

Privacy and Data Use

- The dataset used is **simulated**, meaning no personally identifiable information (PII) or confidential data was processed.
- When transitioning to real-world data, care must be taken to **anonymize sensitive information** and comply with **data protection laws** (e.g., GDPR, HIPAA).

Responsible Use

- The tool is designed **solely for educational and legal security testing** purposes.
- Misuse for unauthorized access or surveillance of networks would violate ethical and legal standards.
- A clear disclaimer and user agreement should accompany any public deployment.

Bias and Fairness

- The model's performance may vary based on the dataset it is trained on. Therefore, it is important to ensure training data includes **diverse attack patterns** and normal behaviors to avoid **model bias**.

7.2 Market Relevance

As industries increasingly rely on interconnected systems, there is a pressing need for intelligent, automated, and adaptable security solutions. The developed IDS addresses several emerging market trends and challenges:

Growing Demand for Smart IDS in CPS

- **Power grids**, **automated factories**, **smart transportation**, and **healthcare devices** form the backbone of modern CPS.
- These systems are vulnerable to attacks like **false data injection**, **denial-of-service**, and **protocol exploits**, necessitating real-time threat detection.

AI in Cybersecurity Market

- According to industry reports, the **AI in cybersecurity market** is projected to grow at a CAGR of over 20% in the next five years.
- Enterprises are increasingly adopting **ML-based anomaly detection**, especially in areas like **Industrial Control Systems (ICS)** and **Supervisory Control and Data Acquisition (SCADA)** networks.

Practical Deployment Advantages

- The web-based interface is lightweight, easy to deploy on local or cloud-based infrastructure.
- With minimal input requirements and fast prediction times, the tool can serve as a **first-level defense mechanism** or as part of a **layered security architecture**.

- **7.3 Potential Use Cases**

Industry	Use Case
Energy & Utilities	Detecting anomalies in smart meters or grid control commands.
Manufacturing	Monitoring network traffic of PLCs and robotics in factories.
Healthcare	Securing medical devices and patient monitoring systems.
Transportation	Detecting threats in autonomous vehicle networks.
Government & Defense	Real-time surveillance of mission-critical CPS environments.

7.4 Competitive Advantage

While traditional IDS tools rely heavily on predefined rules and signatures, AI-based tools like the one developed here offer:

- **Adaptability to new threats** without frequent manual updates.
- **Scalability** across different environments and network sizes.
- **Automation** of detection processes, reducing reliance on human monitoring.

FUTURE SCOPE

The proposed AI-based Intrusion Detection System (IDS) for Cyber-Physical Systems (CPS) demonstrates the initial feasibility of using machine learning for real-time threat detection. However, as CPS environments continue to evolve in scale, complexity, and heterogeneity, so must the capabilities of security mechanisms. This section outlines the potential directions and enhancements for improving and scaling this research.

8.1 Integration with Real-Time CPS Environments

Currently, the model has been tested on simulated data in a controlled environment. For real-world application, the following advancements are essential:

- **Live Traffic Capture:** Integrate the system with tools like Wireshark, Zeek, or custom packet sniffers to collect real-time network traffic from CPS environments.
- **Edge Deployment:** Deploy lightweight models on edge devices (e.g., Raspberry Pi, industrial gateways) to enable local and low-latency intrusion detection.
- **Protocol Awareness:** Extend support to CPS-specific protocols such as Modbus, DNP3, and IEC 61850 to improve applicability in industrial systems.

8.2 Deep Learning for Sequential Pattern Detection

While Random Forest is effective for basic classification tasks, deep learning offers enhanced capability for detecting sophisticated attack patterns, especially those spread over time.

- **LSTM (Long Short-Term Memory):** Ideal for detecting sequence-based anomalies in time-series network data.
- **Autoencoders:** Can be used for unsupervised anomaly detection by learning compressed representations of normal traffic.

- **Graph Neural Networks (GNNs):** Useful for modeling relationships between nodes/devices in large-scale CPS networks.

8.3 Dataset Enhancement and Diversity

Improving the robustness and accuracy of the IDS requires:

- **Real-World Datasets:** Partnering with industry to obtain anonymized data from operational ICS/SCADA environments.
- **Attack Simulation Tools:** Use platforms like Cyber Range, CSET, or Digital Twins to simulate a wider variety of attack scenarios.
- **Multivariate Feature Expansion:** Include more granular features such as latency, packet loss, authentication logs, and encrypted traffic metadata.

8.4 Advanced Model Evaluation

Beyond basic accuracy, the following evaluation strategies should be incorporated:

- **ROC-AUC, F1-Score, Confusion Matrix** for balanced evaluation in imbalanced datasets.
- **Time-to-Detection (TTD):** Measure how quickly threats are identified after occurrence.
- **False Positive Rate Reduction:** Essential for minimizing operational disruptions in CPS.

8.5 Scalable Cloud-Based Deployment

To support enterprise-wide deployments, cloud integration offers the following benefits:

- **Centralized Monitoring Dashboard:** Real-time analytics, visualization, and alerting features.
- **Integration with SIEM Tools:** Connect to systems like Splunk, ELK Stack, or QRadar.

- **Auto-Scaling:** Enable horizontal scaling of detection engines across distributed environments.

8.6 Policy & Compliance Integration

As CPS increasingly fall under regulatory oversight, future versions of the tool should incorporate:

- **Audit Logging:** Maintain traceable logs of detected intrusions and system decisions.
- **Compliance Templates:** Customizable rule sets to align with NIST, IEC 62443, and ISO/IEC 27001 standards.
- **Automated Reporting:** Generate security compliance reports for auditors and stakeholders.

8.7 Community Collaboration and Open Source Contribution

To ensure wider adoption and continuous improvement:

- **Open-Source Release:** Make the tool available on platforms like GitHub for collaboration.
- **Documentation and Tutorials:** Provide extensive user guides and video tutorials.
- **Modular Architecture:** Allow plug-and-play functionality for different models, data parsers, and user interfaces.

REFERENCES

1. Cardoso, J. et al. (2022). AI in Infrastructure Protection. *IEEE*.
2. Lin, Y. et al. (2021). ML for CPS Security. *ACM Computing Surveys*.
3. Sarker, I. H. (2020). AI in Cybersecurity. *Future Generation Computer Systems*.
4. Ahmed, M. et al. (2019). IDS in ICS. *Computer Networks*.
5. Ghosh, P. et al. (2020). Cybersecurity with Random Forest. *Springer*.
6. Zhang, Y. et al. (2018). SCADA Security. *Journal of Network and Computer Applications*.
7. Kumar, R. et al. (2019). Smart Grid IDS. *Elsevier*.
8. Sahu, M. et al. (2021). DL in Intrusion Detection. *MDPI Sensors*.
9. Wang, J. et al. (2020). CPS Hybrid Security. *Elsevier*.
10. *IEEE Transactions on Dependable and Secure Computing*, 2021.