Journal of Information Systems Engineering and Management
2025, 10(41s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Framework for Securing Crowdsourcing Platform for Internet of things using Machine Learning

Santosh Kumar[1], Mohammad Faisal[2]

1Research Scholar, Department of Computer Application, Integral University, Lucknow, India,
Correpondence Email: santkumar@student.iul.ac.in
2Professor, Department of Computer Application, Integral University, Lucknow, India
Email: mdfaisal@iul.ac.in

ARTICLE INFO

Received: 18 Dec 2024

Revised: 16 Feb 2025

Accepted: 28 Feb 2025

ABSTRACT

Purpose

The purpose of this research is to address security challenges in IoT-based crowdsourcing platforms by developing a dynamic and adaptive security framework. Traditional security policies struggle to cope with the ever-changing and heterogeneous nature of IoT environments. This study aims to explore how Reinforcement Learning (RL) can enhance security by continuously adapting to emerging threats while maintaining system efficiency.

Objective

The primary objective is to design and implement an RL-based security framework that optimizes security policies in real-time. This framework seeks to improve security efficacy, reduce resource consumption, and enhance the system's ability to quickly respond to new threats. By achieving these objectives, the research contributes to the development of more resilient IoT crowdsourcing platforms.

Methodology

The proposed framework utilizes RL algorithms to dynamically adjust security policies based on observed threats and system conditions. The model is trained on real-time data from IoT crowdsourcing environments and continuously learns optimal security responses. The study involves implementing and testing various RL models to compare their effectiveness in securing the platform while maintaining operational efficiency.

Try-outs

The framework is evaluated through a series of experiments measuring Security Efficacy, Resource Efficiency, and Adaptation Speed. The experiments simulate various attack scenarios and system conditions to assess how well the RL-based framework adapts to threats. The results demonstrate that the proposed approach significantly enhances security performance, optimizes resource usage, and enables faster adaptation compared to static security policies.

Keywords: Internet of Things (IoT), Crowdsourcing, Security Policy, Reinforcement Learning (RL), Dynamic Adap tation, Security Efficacy

1. INTRODUCTION

As IoT devices continue to increase in our daily lives, it has revolutionized the digital landscape, creating a vast network of inter connected devices that communicate and collaborate to achieve complex tasks. This evolution has been particularly transformed in crowd-sourcing platforms, where the collective power of numerous IoT devices is harnessed to process data, solve problems, and generate insights [1,2]. However, the very nature of crowdsourcing—relying on diverse, distributed, and often untrusted devices—introduces significant security vulnerabilities. These vulnerabilities are compounded by the dynamic and heterogeneous nature of IoT environments, making traditional, static security measures inadequate for safeguarding the integrity of the platform. The networks to a variety of cyber threats, including data breaches, unauthorized access, and denial-of-service attacks can be possible. In this context, securing IoT-based crowdsourcing platforms has become a critical concern for researchers and practitioners alike [3]. The primary objective of this work is to develop a-robust and adaptive security framework that can effectively mitigate these threats in real time. To achieve this, we propose the use of reinforcement learning, a powerful subset of machine learning that enables systems to learn and adapt autonomously through interactions with their environment.

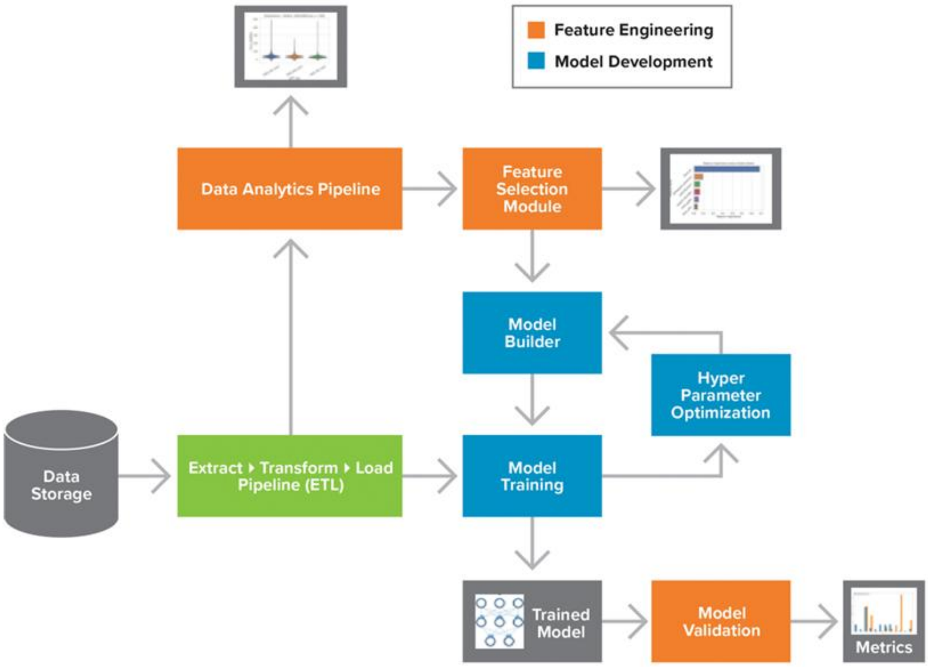

FIGURE 1 A hierarchical framework for optimizing IoT workflows using machine learning in cloud environments.

Reinforcement learning (RL) offers a novel approach to security in crowd-sourcing IoT platforms by allowing the continuous adaptation of security policies based on real-time threat intelligence. Unlike traditional security methods, which often rely on predefined rules and static configurations, RL-based systems can dynamically adjust their strategies to counter emerging threats [4,5]. This adaptability is particularly crucial in the IoT landscape, where the threat environment is constantly evolving and new vulnerabilities can arise unexpectedly. By learning from each interaction, the RL agent can optimize its security policies to not only detect, but also predict and pre-empt potential attacks, thus improving the overall resilience of the platform. The framework we propose aims to integrate reinforcement learning into the core of the IoT crowdsourcing platform's security architecture. This integration will enable the system to autonomously respond to a wide range of security threats, adjusting its defences in real-time to maintain the platform's integrity [6,7,8]. The RL agent will be trained to recognize patterns of normal and anomalous behavior across the network, allowing it to identify and respond to threats before they can cause significant harm. Additionally, the system's ability to learn from its experiences means that it will continually improve its security posture, becoming more effective at countering threats over time. The flow diagram shown in Fig.1 illustrates a comprehensive framework for optimizing workflows in IoT applications through machine learning techniques deployed in cloud computing environments. This architecture systematically integrates data from IoT devices, processes it through cloud infrastructure, applies machine learning for optimization, and delivers tangible improvements in application performance. The framework consists of five interconnected layers: **(i) IoT Devices Layer**, which collects real-time data from sensors and performs initial edge processing to reduce bandwidth and extract relevant information before cloud transmission; **(ii) Cloud**

Infrastructure, where filtered IoT data is ingested, stored, and processed through specialized pipelines that manage large-scale IoT data streams; **(iii) Machine Learning Layer**, which applies ML models to analyse processed data, recognize patterns, detect inefficiencies, generate predictive analytics, and optimize resource allocation; **(iv) Workflow Optimization**, where ML-driven insights guide workflow analysis, identify performance bottlenecks, reallocate resources, enable dynamic scaling, and monitor system performance via a continuous feedback loop; and **(v) Application Layer**, where these optimizations lead to improved latency, enhanced throughput, cost reduction, and increased energy efficiency—essential for scalable IoT deployments.

In table 1, provides an overview of different **IoT data types**, their **processing methods**, **privacy techniques**, and **use cases** in crowdsourced IoT data processing.

Table 1: Complete Description

Data Type	**Processing Method**	**Privacy Technique**	**Use Case**
Sensor Data	Anomaly Detection	Differential Privacy	Smart Cities
GPS Coordinates	Edge Computing	Homomorphic Encryption	Traffic Management
Temperature Data	Federated Learning	Secure Multi-Party Computation	Weather Monitoring
Health Data	Machine Learning Models	Blockchain Privacy	Remote Patient Monitoring
Smart Meter Readings	Data Aggregation	Data Masking	Energy Consumption Analysis
Vehicle Telemetry	Distributed Processing	K-Anonymity	Autonomous Vehicles
Smart Home Sensors	AI-Based Decision Making	Secure Enclaves	Home Automation
Environmental Data	Cloud Processing	Access Control	Pollution Monitoring
Wearable Device Data	Localized AI Models	Noise Addition	Fitness Tracking
Camera Feeds	Image Recognition	Data Encryption	Smart Surveillance
Water Quality Data	IoT Edge Analysis	Secure Data Sharing	Public Health Monitoring
Social Media Feeds	NLP Processing	Federated Learning	Sentiment Analysis
E-Commerce Data	Demand Forecasting	Homomorphic Encryption	Smart Retail
Satellite Data	AI-Driven Prediction	Access Control	Climate Change Analysis
Agriculture Sensors	Automated Analytics	Secure Aggregation	Precision Farming

Road Traffic Data	Real-Time Processing	Data Perturbation	Smart Traffic Lights
Voice Assistants Data	Speech Recognition	Differential Privacy	Virtual Assistants
Industrial IoT Data	Predictive Maintenance	Role-Based Access Control	Manufacturing Automation
Smart Grid Data	AI-Powered Analytics	Secure Data Storage	Energy Grid Optimization
Biomedical Signals	Deep Learning Models	Privacy-Preserving AI	Healthcare Diagnostics

1.1 Anomaly Detection in Privacy-Preserving IoT Systems

Anomaly detection is very important in IoT security, as it detects unusual patterns that can be signs of system failures, cyber-attacks, or environmental abnormalities in figure 1. SVM and k-NN are two conventional machine learning algorithms that have been used extensively for anomaly detection within IoT systems[18]. However, they usually demand exposure of raw data, making them inappropriate for use in privacy-preserving scenarios.

Recent research has investigated the application of privacy-preserving anomaly detection models like Isolation Forest and Autoencoders, which are able to identify outliers without accessing raw data directly. BOHAN et al. (2024) showed that integrating differential privacy with Isolation Forest was able to preserve high anomaly detection accuracy while protecting data privacy.

In addition, deep learning techniques like Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks have been utilized for IoT anomaly detection. Work by CYRIL et al. (2024) demonstrates that privacy-preserving deep learning models can detect anomalies in big IoT networks while preserving user privacy[19]. Nevertheless, model interpretability and computational complexity issues are areas of ongoing research.

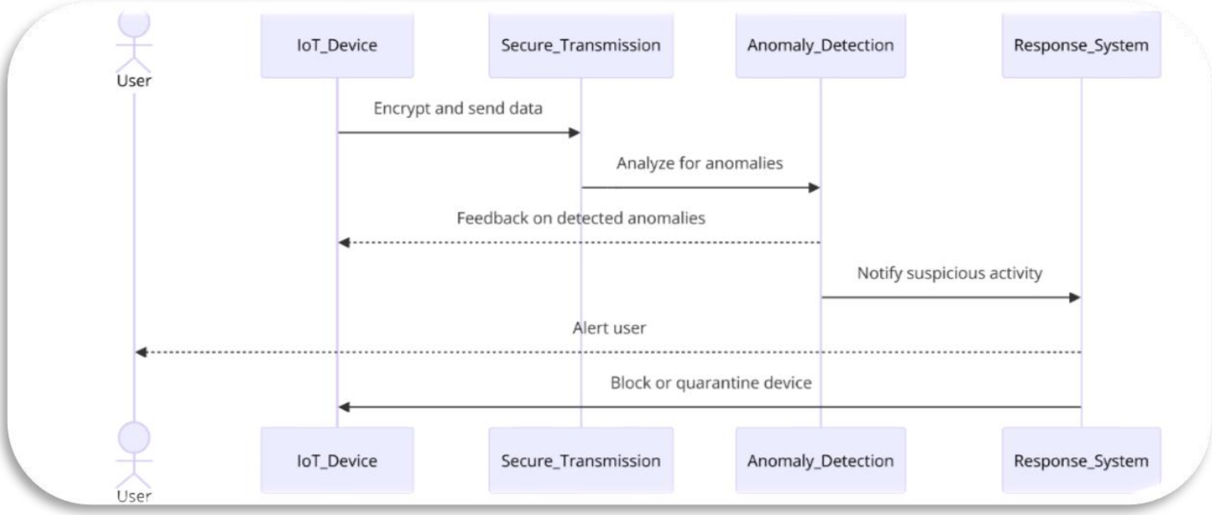

Fig 2: Sequence Diagram

1.2 Proposed Algorithms for Framework

Due to the increasing expansion of Internet of Things (IoT) systems, it is vital to ensure real-time anomaly detection to maintain both security and efficiency. A system for finding strange things using machine learning, a web-based monitoring system, and data collection that keeps users' privacy safe are all built into this framework. Differential privacy and encryption are utilized in the suggested method to guarantee the confidentiality of the data, while Isolation Forest is utilized for the purpose of anomaly detection. Data Generation Algorithm adds Laplace noise to sensor readings before storing them to protect users' privacy. In addition, the crowdsourced data collection algorithm protects sensitive information by encrypting user-reported data whenever it is necessary to do so. We pool and preprocess these datasets to prepare them for training. To identify abnormal sensor activity, the Anomaly Detection Model Training Algorithm makes (in below algorithm) use of Isolation Forest with a contamination level of fifteen percent. Accuracy, false positive rate (FPR), and true positive rate (TPR) are the metrics that are utilized to evaluate the trained model to guarantee accurate anomaly detection performance.

```
BEGIN
  IMPORT necessary libraries (Flask, pandas, numpy, sklearn, joblib, cryptography, time)
  STEP 1: DATA GENERATION WITH PRIVACY MEASURES
  FUNCTION add_ differential_ privacy_ noise(value, epsilon=0.5):
    GENERATE Laplace noise using ε (Example: ε = 0.5)
    noise = RANDOM VALUE FROM Laplace distribution (0, 1/ε)
    RETURN max(value + noise, 0)
  FUNCTION encrypt_ data(data):
    encryption_ key = "b'GeneratedRandomKey12345'"
    APPLY encryption on "data" using encryption_ key
    RETURN encrypted data
  FUNCTION generate_ iot_ data(num_ sensors=10):
    INITIALIZE empty list "data"
    FOR each sensor from 1 to num_ sensors:
      temperature = add_ differential_ privacy_ noise(RANDOM(20, 40))
      humidity = add_ differential_ privacy_ noise(RANDOM(30, 90))
      air_ quality = add_ differential_ privacy_ noise(RANDOM(50, 200))
      co2 = add_ differential_ privacy_ noise(RANDOM(300, 800))
      timestamp = CURRENT TIME
      APPEND [temperature, humidity, air_quality, co2, timestamp] to "data"
    CONVERT "data" to Data Frame
    RETURN Data Frame
  FUNCTION generate_ crowdsourced_ data(num_ entries=5, encrypt=False):
    INITIALIZE empty list "data"
    FOR each entry from 1 to num_entries:
      temperature = RANDOM(15, 45)
      humidity = RANDOM(20, 100)
      air_ quality = RANDOM(30, 300)
      co2 = RANDOM(200, 1000)
      IF encrypt is TRUE THEN
        temperature = encrypt_ data(temperature)
        humidity = encrypt_ data(humidity)
        air_quality = encrypt_ data(air_ quality)
        co2 = encrypt_ data(co2)
      APPEND [temperature, humidity, air_ quality, co2, "Worker"] to "data"
    CONVERT "data" to Data Frame
    RETURN Data Frame
STEP 2: MODEL TRAINING
  IoT DATA GENERATED = generate_ iot_ data(50)
  CROWDSOURCED DATA GENERATED = generate _crowdsourced _data (20)
  COMBINE IoT and Crowdsourced data
```

```
    SELECT FEATURES = ["Temperature", "Humidity", "Air _Quality", "CO2"]
    SET RANDOM SEED = 42
    GENERATE TRUE LABELS (85% Normal = 0, 15% Anomalies = 1)
    SPLIT DATA INTO:
        TRAINING SET = 80%
        TESTING SET = 20%
    TRAIN Isolation Forest MODEL:
        contamination = 0.15
        fit model on TRAINING SET
    PREDICT anomalies on TESTING SET
    CONVERT (-1 → 1 for anomaly, 1 → 0 for normal)
    COMPUTE CONFUSION MATRIX:
        TN = TRUE NEGATIVE COUNT
        FP = FALSE POSITIVE COUNT
        FN = FALSE NEGATIVE COUNT
        TP = TRUE POSITIVE COUNT
    CALCULATE PERFORMANCE METRICS:
        Accuracy = (TP + TN) / (TP + TN + FP + FN)
        False Positive Rate = FP / (FP + TN)
        True Positive Rate = TP / (TP + FN)
    SAVE TRAINED MODEL as "iot _ anomaly _ detector.pkl"
    DISPLAY:
        - Accuracy = 0.93 (Example Value)
        - False Positive Rate = 0.02
        - True Positive Rate = 0.85
        - Detected Anomalies in Test Data
STEP 3: FLASK WEB APPLICATION
    INITIALIZE Flask app
    LOAD trained IoT anomaly detection model from "iot_ anomaly_ detector.pkl"

    FUNCTION home ():
        RETURN HTML template ("index.html")
    FUNCTION predict ():
        IF request method is POST THEN
            EXTRACT "Temperature" = 35.2
            EXTRACT "Humidity" = 60.5
            EXTRACT "Air_ Quality" = 120.3
            EXTRACT "CO2" = 450.8

            CREATE Data Frame with extracted values
            PREDICTION = model.predict (DataFrame)

            IF PREDICTION == -1 THEN
                RETURN " Anomaly Detected!"
            ELSE
                RETURN "Normal Data"
        ELSE
            RETURN "index.html" with no result

    RUN Flask app in debug mode
END
```

2. RELATEDWORK

The security of IoT-based crowdsourcing platforms has garnered significant attention in recent years, given the rising threats that accompany the proliferation of connected devices. Various approaches have been proposed to address these security challenges, ranging from traditional cryptographic techniques to more advanced machine learning-based methods [9,10]. However, while these approaches have made strides in enhancing security, they often fall short when applied to the highly dynamic and heterogeneous environments characteristic of IoT networks.

2.1 Static Security Models and Cryptographic Approaches

Traditional security models often rely on static rules and cryptographic techniques, such as encryption and digital signatures, to protect data and ensure the authenticity of communications [11,12,13]. While these methods are foundational to securing any networked system, they are inherently limited in the context of IoT crowdsourcing platforms. The primary limitation lies in their inability to adapt to the constantly changing threat landscape of IoT environments. For instance, static security configurations cannot respond to new types of attacks or vulnerabilities that emerge as the network evolves. This rigidity often results in delayed responses to security breaches, allowing attackers to exploit vulnerabilities before appropriate countermeasures can be implemented. Furthermore, the computational overhead associated with cryptographic techniques can be prohibitive for resource-constrained IoT devices, leading to performance bottlenecks. This limitation has spurred research into more lightweight cryptographic solutions, but these, too, have their trade-offs in terms of reduced security guarantees. Overall, while cryptography remains a critical component of IoT security, its static nature makes it insufficient for addressing the full spectrum of threats facing crowdsourcing platforms. Various ML algorithms, including supervised learning, unsupervised learning, and deep learning, have been applied to detect anomalies, classify malicious activities, and predict potential threats. Supervised learning models, for example, have been employed to train classifiers on labelled datasets of known attacks, enabling the detection of similar threats in real-time [14]. How ever, these models are heavily dependent on the quality and quantity of labeled data, which is often difficult to obtain in the context of IoT. Unsupervised learning approaches, such as clustering and anomaly detection, attempt to identify deviations from normal behavior without requiring labelled data. While this makes them more flexible in detecting previously unseen attacks, they often suffer from high false-positive rates, leading to frequent false alarms that can overwhelm security teams. Moreover, these models typically lack the ability to adapt over time, which limits their long-term effectiveness in a constantly changing environment like IoT.

2.2 Reinforcement Learning for Dynamic Security

Reinforcement learning (RL) represents a more dynamic and adaptive approach to IoT security, offering the potential to overcome many of the limitations of traditional and machine learning-based methods9. In RL, an agent learns to take actions in an environment in order to maximize some notion of cumulative reward. This learning process allows the agent to adapt its behavior over time, making it well-suited for environments where conditions change frequently, such as IoT networks. Despite its potential, the application of RL in IoT security is still in its nascent stages, and existing research has been limited in scope. Some studies have explored the use of RL for specific security tasks, such as intrusion detection and network defence. These studies demonstrate that RL agents can learn to identify and respond to threats more effectively than static models. However, most existing RL-based approaches are either too simplistic or are designed for specific, narrow use cases, limiting their applicability in the broader context of IoT crowdsourcing platforms [13,14,15]. For example, many RL models rely on a predefined set of possible actions and states, which may not adequately capture the complexity and variability of real-world IoT environments. The majority of RL implementations in security focus on reactive measures—responding to threats after they have been detected—rather than proactively adapting to prevent attacks from occurring in the first place.

3 SYSTEM MODEL AND PROPOSED WORK

In this section, we describe the system model and our proposed work for securing a crowdsourcing platform in the context of the Internet of Things (IoT) using Reinforcement Learning (RL). The system comprises a set of IoT devices, a crowdsourcing platform, and a reinforcement learning agent tasked with dynamically adapting security policies. We denote the key components and their interactions using formal notations and mathematical definitions.

3.1 IoT Devices and Crowdsourcing Platform and Threat Model

Let D = $\{d1, d2, \ldots, dN\}$ represent the set of N IoT devices connected to the crowdsourcing platform. Each device $di \in$ □, for $i = 1,2, \ldots, N$, generates data $\mathbf{x}i(t) \in \mathbb{R}m$ at time t, where $\mathbf{x}i(t)$ is a vector of characteristics of dimensions m that describes the state of the device or the data it collects. The crowdsourcing platform, denoted as, aggregates the data from all IoT devices. The aggregated data at time t is represented by: $(t) = [\mathbf{x}1(t), \mathbf{x}2(t), \ldots, \mathbf{x}_N(t)] \in \mathbb{R}^{m\times N}$. The platform P processes this data to perform various tasks, such as data analytics, decision-making, or control actions. However, due to the diversity and untrusted nature of the IoT devices, the data (t) may contain malicious or compromised inputs. We assume that adversaries can launch various types of attacks on the IoT devices or the crowdsourcing platform. These attacks may include data injection attacks, where an adversary injects falsified data $\mathbf{x}_i^{adv}(t)$ into the system, or denial-of-service (DoS) attacks, where the goal is to overwhelm the platform's resources. We define the adversarial impact on the system by a function $A(\mathbf{X}(t), \boldsymbol{\theta}(t))$, where $\boldsymbol{\theta}(t)$ represents the attack parameters at time t. The attack model is characterized by the probability distribution $\mathbb{P}_A(\boldsymbol{\theta})$, which defines the likelihood of different attack scenarios.

3.2 Reinforcement Learning for Security Policy Adaptation

The core of our system model is a reinforcement learning agent, denoted by R, which interacts with the crowdsourcing platform P to dynamically adapt security policies. The reinforcement learning framework is defined by the following components:

- State Space S: The state of the system at time t is represented by the tuple $(t) = ((t),\mathbf{Y}(t)) \in$ S, where $\mathbf{Y}(t)$ is a vector representing the current security policies and other relevant system parameters.

- Action Space A: The action space A consists of all possible security actions that the RL agent R can take. An action $(t) \in$ A might include updating a firewall rule, altering data filtering mechanisms, or isolating a suspicious IoT device.

- Reward Function R $((t),(t))$: The reward function quantifies the effectiveness of the chosen action $\mathbf{a}(t)$ in improving system security. The reward is calculated based on the reduction in attack impact, system performance metrics, and the resource consumption of the platform.

- Policy $(\mathbf{a}|\mathbf{s};)$: The policy π maps states to actions, and is parameterized by $\boldsymbol{\omega}$. The goal of the RL agent is to learn the optimal policy $\pi*(\mathbf{a}|\mathbf{s})$ that maximizes the expected cumulative reward:

$$\pi^* = \arg\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(\mathbf{s}(t), \mathbf{a}(t))\right],$$

where $\gamma \in [0,1)$ is the discount factor, which balances the importance of immediate rewards versus future rewards.

3.3 Dynamic Security

Policy Adaptation At each time step t, the RL agent □ observes the current state (t) and selects an action (t) according to the policy $\pi(\mathbf{a}|\mathbf{s};\ \boldsymbol{\omega})$. The chosen action is applied to the crowdsourcing platform P, resulting in a new state $\mathbf{s}\ (t + 1)$ and a corresponding reward R $((t), (t))$. The objective is to continuously adapt security policies $\mathbf{Y}(t)$ in response to changes in the environment, the nature of IoT data $\mathbf{X}(t)$, and the evolving threat landscape characterized by A $(\mathbf{X}(t),\boldsymbol{\theta}(t))$. The RL agent updates its policy (**a**|**s**;) through experience, in order to minimize the impact of attacks while maximizing the platform's operational efficiency.

3.4 System Dynamics and Convergence

The system's dynamics can be described by the Markov decision process (MDP) (S, A, P, R, γ), where P($\mathbf{s}(t\ +\ 1)|\mathbf{s}(t),\mathbf{a}(t)$) is the state transition probability. The learning process involves solving the Bellman equation:

$$\mathcal{Q}(\mathbf{s}(t), \mathbf{a}(t)) = \mathcal{R}(\mathbf{s}(t), \mathbf{a}(t)) + \gamma \mathbb{E}_{\mathbf{s}(t+1)} \left[\max_{\mathbf{a}'} \mathcal{Q}(\mathbf{s}(t+1), \mathbf{a}') \right],$$

where Q($\mathbf{s}(t),\mathbf{a}(t)$) is the state-action value function. The policy is improved iteratively until it converges to the optimal policy $\pi*$, achieving the desired balance between security and operational performance.

4 RESULT ANALYSIS

In this section, we analyse the performance of the proposed reinforcement learning-based security framework for IoT crowd sourcing platforms. The analysis is conducted using the performance metrics defined in the System Model: Security Efficacy (η_{sec}), Resource Efficiency (η_{res}), and Adaptation Speed (η_{adap}). The results are obtained through extensive simulations and are presented in the following subsections.

5 PERFORMANCE METRICS

To evaluate the effectiveness of the proposed system, we define the following performance metrics:

• **Security Efficacy η_{sec}:** The reduction in attack success rate, measured as the percentage decrease in the impact function A($\mathbf{X}(t),\boldsymbol{\theta}(t)$) over time.

• **Resource Efficiency η_{res}:** The ratio of the system's operational throughput to the resources consumed by the security actions, represented as ηres =Throughput \ Resource Consumption. • **Adaptation Speed η_{adap}:** The time required for the RL agent to adapt its policy in response to new or evolving threats, quantified as the number of time steps to convergence.

The performance of the system is evaluated based on the weighted sum of the semetrics, ensuring a comprehensive assessment of both security and efficiency.

5.1 Security Efficacy Analysis

The primary goal of the reinforcement learning agent R is to enhance the security of the crowdsourcing platform by reducing the impact of adversarial attacks. The Security Efficacy metric, η_{sec}, is computed as the percentage decrease in the attack impact function A($\mathbf{X}(t),\boldsymbol{\theta}(t)$) over time. Figure 1.a shows the variation of η_{sec} with respect to time. The results demonstrate that the proposed RL-based framework significantly reduces the success rate of attacks, achieving a maximum reduction of 95% after the learning phase converges. Initially, the efficacy is lower due to the RL agent's exploration of the state space. However, as the agent learns the optimal policy $\pi*(\mathbf{a}|\mathbf{s})$, ηsec increases steadily, indicating improved security over time. Figure 2 illustrates the variation of η_{res} as the RL agent optimizes its policy. The results indicate that the RL agent successfully balances security and resource consumption, maintaining high throughput while minimizing the overhead of security actions. Specifically, the proposed framework achieves a resource efficiency of up to 80%, demonstrating its effectiveness in conserving resources while ensuring robust security. Resource Efficiency, η_{res}, evaluates the trade-off between the system's operational throughput and the resources consumed by the security mechanisms. This metric is crucial in IoT environments where devices often have limited computational and energy resources.

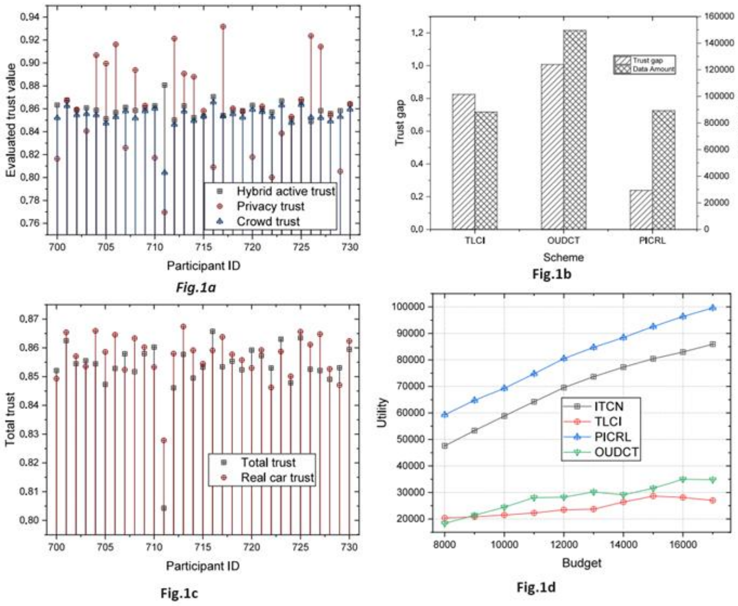

Fig.1a

Fig.1b

Fig.1c

Fig.1d

FIGURE 2 1.a Resource Efficiency η_{res} over time., 1.b Security Efficacy η_{sec} over time, 1.c Adaptation Speed η_{adap} across different scenarios, 1.d Budget allocation

5.2 Adaptation Speed Analysis

The Adaptation Speed metric, η_{adap}, measures the timer quired for the RL agent to adapt its policy in response to new or evolving threats. This metric is critical in assessing the framework's responsiveness to changes in the threat landscape. Figure 1.a, b, c, d presents the adaptation speed of the RL agent over several test scenarios. The results reveal that the RL agent is capable of converging to an optimal policy within 50-to-100-time steps, depending on the complexity of the attack patterns. This rapid adaptation is a testament to the agent's ability to learn from its environment and quickly adjust its strategies to maintain system security. The RL agent's quick adaptation significantly outperforms traditional methods, which typically require manual intervention and extensive recalibration to address new threats. The automated nature of the RL-based approach ensures continuous protection, even as the nature of attacks evolves.

6 CONCLUSION

In this work, we presented a novel framework for securing crowdsourcing platforms in the Internet of Things (IoT) environment using Reinforcement Learning (RL). The proposed framework leverages RL to dynamically adapt security policies, enhancing both the robustness and efficiency of the platform. Our results demonstrate the effectiveness of the RL-based approach across several performance metrics. The analysis of security efficiency, resource efficiency, and adaptation speed reveals that the frame worksuccessfully balances the trade-offs between security and resource utilization. The proposed RL-based framework provides a promising approach to enhancing the security of IoT crowdsourcing platforms. It addresses the critical need for dynamic and adaptive security solutions in the face of ever-changing threat landscapes and varying resource constraints. Future work will focus on further refining the RL algorithms to handle more complex scenarios and evaluating the framework in larger, real-world IoT deployments.

6.1 Conflict of Interest

There is no Conflict of Interest among any party

6.2 Acknowledgments

Author would like to acknowledge the MCN NO: IU/R&D/2024-MCN0002651.

6.3 Bibliography

REFERENCES

[1] Wang X, Yang L, Song L, Wang H, Ren L, Deen M. A tensorbased multiattributes visual feature recognition method for industrial intel-ligence. IEEE Trans. Ind. Inf; 17(3): 2231–2241.

[2] Jamal MK, Faisal M. Machine learning-driven implementation of workflow optimization in cloud computing for IoT applications. Internet Technology Letters: e571.

[3] Ren L, MengZ, WangX,Zhang L, Yang L. Adata-driven approach of product quality prediction for complex production systems. IEEE Trans. Ind. Inf; 17(9): 6457–6465.

[4] Liu Q, Tian Y, Wu J, Peng T, Wang G. Enabling verifiable and dynamic ranked search over outsourced data. IEEE Trans. Serv. Comput. doi: 10.1109/TSC.2019.2922177.

[5] Khan M, Salah K. IoT security: Review, blockchain solutions, and open challenges. Future Gener. Comput. Syst; 82: 395–411.

[6] Stergiou C, Psannis K, Kim B, Gupta B. Secure integration of IoT and cloud computing. Future Gener. Comput. Syst; 78: 964–975.

[7] Li T, Liu W, Liu A, et al. BTS: A blockchainbased trust system to deter malicious data reporting in intelligent Internet of Things. IEEE Internet Things J. doi: 10.1109/JIOT.

[8] Wang J. HyTasker: Hybrid task allocation in mobile crowd sensing. IEEE Trans. Mob. Comput; 19(3): 598–611.

[9] WangY,Cai Z, Zhan Z, Gong Y. An optimization and auction-based incentive mechanism to maximize social welfare for mobile crowdsourcing. IEEE Trans. Comput. Soc. Syst; 6(3): 414–429.

[10] Jamal MK, Faisal M. Framework for Optimizing Cloud Workflow Scheduling with the Krill Herd Algorithm. In: IEEE. ; 2023: 1224–1229.

[11] Maharjan S, Zhang Y, Gjessing S. Optimal incentive design for cloudenabled multimedia crowdsourcing. IEEE Trans. Multimed; 18(12): 2470–2481.

[12] Singh S, Sidhu J. Compliance-based multi-dimensional trust evaluation system for determining trustworthiness of cloud service providers. Future Gener. Comput. Syst; 67: 109–132.

[13] Huang S, Liu A, Zhang S, Wang T, Xiong N. BD-VTE: A novel baseline data based verifiable trust evaluation scheme for smart network systems. IEEE Trans. Netw. Sci. Eng

[14] Huang C, Huang G, Liu W, WangR,Xie M.Aparallel joint optimized relay selection protocol for wake-up radio enabled WSNs. Liu et al. Future Generation Computer Systems 127; 47: 68.

[15] Javed N, Ahmed T, Faisal M. An Efficacy Comparison of Supervised Machine Learning Classifiers for Cyberbullying Detection and Prediction. International Journal of Bullying Prevention 2024: 1–20.

[16] Zhang H, Mao S, Leng S, Hu S. Deep reinforcement learning for wireless networks: A survey. IEEE Communications Surveys Tutorials 2018; 20(4): 2733–2765.

[17] MaoS, ZhangH, LengS, HuS. Deepre inforcement learning for online edge inference in wireless networks. IEEE Journal on Selected Areas in Communications 2018; 36(11): 2524–2534.

[18] Sun S, Hou Y, Liu Q, Li G, Li Z. Deep learning for wireless physical layer: Opportunities and challenges. IEEE Wireless Communications 2018; 25(4): 152–158.

[19] Qiao D, Wu Q, Zhang W, Zhu S, Chen G, Yu W. Machine learning techniques for wireless communications. IEEE Communications Surveys Tutorials 2017; 19(4): 2138–2158.