POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Deep Reinforcement Learning for Intrusion Detection in IoT Systems: Enhancing Security and Energy Efficiency

SAEID JAMSHIDI
Département de génie informatique et génie logiciel

Thèse présentée en vue de l'obtention du diplôme de *Philosophiæ Doctor*
Génie informatique

Mai 2025

Deep Reinforcement Learning for Intrusion Detection in IoT Systems: Enhancing Security and Energy Efficiency

DEDICATION

I dedicate this work to myself—for my perseverance, effort, and determination. Through challenges and setbacks, I refused to give up. I pushed forward, believed in myself, and made it to this moment.

This is a reminder that hard work and resilience always pay off.

To me—because I deserve it.

ACKNOWLEDGEMENTS

With deep love and gratitude, I dedicate this thesis to my family—the foundation of my journey—whose unwavering support, sacrifices, and encouragement have shaped my academic and personal endeavors.
To my mother and father, whose endless love, wisdom, and resilience have been my most significant source of strength and inspiration.
To my sister, Dr. Leila Jamshidi, whose unwavering belief in me, insightful guidance, and constant encouragement have been invaluable throughout my journey.
To my brothers—Ali, Akbar, and Amir—engineers in their own right, whose passion for innovation, problem-solving, and dedication to their fields have continuously motivated me to strive for excellence and push the boundaries of knowledge.
I extend my deepest gratitude to my esteemed supervisors, Professor Foutse Khomh and Dr. Amin Nikanjam at the SWAT Lab, whose mentorship, wisdom, and unwavering support have been instrumental in my academic and professional growth.
Thanks to my colleague and friend Nafi, whose support and thoughtful insights have added depth to this academic path. I also sincerely thank Professor Samira Keivanpour at the Poly Circle X.0 Lab, whose guidance, collaboration, and support have greatly enriched my research experience, along with Dr. Ashkan Amirnia, whose collaboration has been invaluable in this journey.
I am genuinely grateful to Amin, Amir, and his wife Mehri, whose meaningful and joyful conversations have brought moments of inspiration, clarity, and friendship throughout this journey.
I offer my most profound appreciation to each of you. This achievement is as much yours as it is mine.

RÉSUMÉ

L'Internet des Objets (IoT) transforme les industries en permettant l'automatisation en temps réel et la prise de décision intelligente dans les villes intelligentes, les systèmes de santé et les environnements industriels. Toutefois, son expansion rapide introduit des défis de cybersécurité majeurs, aggravés par les contraintes de ressources, la diversité des protocoles de communication et l'évolution des menaces, telles que les attaques par déni de service distribué (DDoS). Les systèmes de détection d'intrusion (IDS) traditionnels souffrent de taux élevés de faux positifs, d'une adaptabilité limitée et d'une charge computationnelle excessive, soulignant ainsi la nécessité de solutions de sécurité intelligentes et auto-adaptatives conciliant détection des menaces et efficacité énergétique.

Cette thèse répond à ces défis en proposant trois modèles innovants d'IDS basés sur l'apprentissage par renforcement profond (DRL) : DeepEdgeIDS, AutoDRL-IDS et EdgeShield-DRL. Ces modèles détectent et atténuent dynamiquement les cybermenaces à grande échelle grâce à l'utilisation de DRL supervisé et non supervisé, de la relecture d'expérience (experience replay) et de fonctions de récompense adaptatives. De plus, six modèles de sécurité IoT largement utilisés; à savoir: Personal Zone Hub (PZH), Trusted Communication Partner (TCP), Outbound-Only Connection (OOC), Blacklist (BL), Whitelist (WL) et Secure Sensor Node (SSN), sont évalués en termes de charge CPU, d'utilisation mémoire, de consommation énergétique et de robustesse en matière de sécurité. Ces analyses ont conduit au développement d'un cadre de sélection de patrons de sécurité basé sur le DRL, capable d'ajuster dynamiquement les configurations de sécurité en fonction des conditions en temps réel des passerelles de périphérie ("real-time edge gateway"), améliorant ainsi l'évolutivité et l'efficacité du système.

L'un des apports majeurs de cette recherche est SecuEdge-DRL, un cadre de cybersécurité auto-adaptatif intégrant le DRL avec le modèle MAPE-K ("Monitor-Analyze-Plan-Execute-Knowledge") afin d'optimiser dynamiquement les politiques de sécurité. Cette thèse introduit également une suite de test IDS modulaire, offrant une plateforme d'évaluation standardisée pour les modèles IDS basés sur l'apprentissage automatique dans les environnements IoT et Software-Defined Networking (SDN). Des expérimentations approfondies sur banc d'essai démontrent que les IDS basés sur le DRL améliorent significativement la précision de détection des menaces, réduisent le temps de réponse, diminuent le taux de faux positifs et améliorent la consommation énergétique, surpassant ainsi les approches IDS conventionnelles. En réduisant la charge computationnelle et la consommation énergétique tout en limitant les émissions de carbone, cette recherche contribue à l'essor de solutions de cybersécurité durables pour les déploiements IoT à grande échelle.

ABSTRACT

The Internet of Things (IoT) transforms industries by enabling real-time automation and intelligent decision-making in smart cities, healthcare, and industrial systems. However, its rapid expansion introduces critical cybersecurity challenges, exacerbated by resource constraints, diverse communication protocols, and evolving threats such as distributed Denial of Service (DDoS) attacks. Traditional Intrusion Detection Systems (IDS) suffer from high false positive rates, limited adaptability, and excessive computational demands, underscoring the need for intelligent, self-adaptive security solutions that balance threat detection and energy efficiency.

This dissertation addresses these challenges by proposing three novel Deep Reinforcement Learning (DRL)-driven IDS models—DeepEdgeIDS, AutoDRL-IDS, and EdgeShield-DRL-that dynamically detect and mitigate large-scale cyber threats using supervised and unsupervised DRL, experience replay, and adaptive reward functions. Additionally, six widely used IoT security patterns—Personal Zone Hub (PZH), Trusted Communication Partner (TCP), Outbound-Only Connection (OOC), Blacklist (BL), Whitelist (WL), and Secure Sensor Node (SSN)—are evaluated in terms of CPU load, memory usage, energy consumption, and security robustness. These insights inform the development of a DRL-driven security pattern selection framework that autonomously adjusts security configurations based on real-time edge gateway conditions, enhancing scalability and efficiency.

A key contribution of this research is SecuEdge-DRL, a self-adaptive cybersecurity framework integrating DRL with the MAPE-K (Monitor-Analyze-Plan-Execute-Knowledge) model to optimize security policies dynamically. This dissertation also introduces a plugin-based IDS test suite, providing a standardized evaluation platform for ML-based IDS models in IoT and Software-Defined Networking (SDN) environments. Comprehensive real-world testbed experiments demonstrate that DRL-based IDSs significantly improve threat detection accuracy, reduce response time, lower false positive rates, and optimize energy consumption, outperforming conventional IDS approaches. By minimizing computational overhead and energy consumption while mitigating carbon emissions, this research advances sustainable cybersecurity solutions for large-scale IoT deployments.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

Acronym	**Description**
AC	Actor-Critic
AFB	Adaptive Feature Boosting
AI	Artificial Intelligence
APT	Advanced Persistent Threat
CI	Confidence Interval
CICIDS	Canadian Institute for Cybersecurity IDS Dataset
CPU	Central Processing Unit
CRN	Cognitive Radio Network
DDPG	Deep Deterministic Policy Gradient
DDQN	Double Deep Q-Network
DL	Deep Learning
DoS	Denial of Service
DQL	Deep Q-Learning
DRL	Deep Reinforcement Learning
DQN	Deep Q-Network
FL	Federated Learning
GAN	Generative Adversarial Network
GCN	Graph Convolutional Network
GOA	Grasshopper Optimization Algorithm
HIL	Hardware-in-the-Loop
IDS	Intrusion Detection System
IEC	International Electrotechnical Commission
IIoT	Industrial Internet of Things
IoT	Internet of Things
IoV	Internet of Vehicles
IRL	Inverse Reinforcement Learning
ITS	Intelligent Transportation Systems
kNN	k-Nearest Neighbors
LFA	Link Flooding Attack
MADDPG	Multi-Agent Deep Deterministic Policy Gradient
MAB	Multi-Armed Bandit

ML	Machine Learning
MITM	Man-in-the-Middle
NFQ	Neural Fitted Q-Iteration
NSL-KDD	Network Security Lab Knowledge Discovery Database
OOC	Out-of-Context
PG-DRL	Policy Gradient Deep Reinforcement Learning
PPO	Proximal Policy Optimization
PZH	Privacy Zone Handler
QoS	Quality of Service
RFC	Random Forest Classifier
RL	Reinforcement Learning
R2L	Remote-to-Local
RSU	Roadside Units
SAC	Soft Actor-Critic
SARSA	State-Action-Reward-State-Action
SCADA	Supervisory Control and Data Acquisition
SDG	Sustainable Development Goals
SDN	Software-Defined Networking
SSN	Secure Sensor Network
SVM	Support Vector Machine
TPR	True Positive Rate
TRE	Time Relative Error
U2R	User-to-Root
UAV	Unmanned Aerial Vehicles
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-Everything
WL	Whitelist
WSN	Wireless Sensor Network

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

The Internet of Things (IoT) is revolutionizing industries by integrating billions of connected devices into smart cities, healthcare, industrial automation, and critical infrastructure [3], [4]. By 2025, IoT adoption is projected to exceed 75 billion devices, enabling automation, real-time data exchange, and intelligent decision-making [5] [6]. However, this rapid expansion has introduced significant cybersecurity challenges, driven by IoT's decentralized nature, lack of standardized security protocols, and resource constraints [7]. Many IoT devices operate with minimal processing power, limited memory, and constrained energy capacity, making them highly susceptible to cyberattacks [8]. Adversaries exploit these vulnerabilities through Distributed Denial of Service (DDoS), Denial of Service (DoS), Mirai and Gafgyt botnets, DoS GoldenEye, DoS Hulk attacks, and Port Scanning [9] [10] [11] [12]. These attacks can disrupt network operations, compromise sensitive data, and cause large-scale service failures, highlighting the urgent need for advanced and adaptive security solutions.

Ensuring IoT network security is a complex and demanding task due to several critical factors [13]. Many IoT devices lack computational resources, using traditional security mechanisms, e.g., encryption, firewalls, and real-time monitoring, infeasible [14] [15]. This necessitates the development of lightweight, energy-efficient security solutions that do not overwhelm device capabilities. Additionally, heterogeneous communication protocols, cloud-edge architectures, and legacy systems introduce security gaps that attackers exploit via malware propagation, unauthorized access, and data manipulation [16] [17]. The dynamic nature of cyber threats, including zero-day vulnerabilities and adversarial Machine Learning (ML) techniques, further complicates the detection and mitigation of attacks [18] [19] [20]. Traditional Intrusion Detection Systems (IDS), including signature-based and anomaly-based models, face fundamental limitations in IoT systems. Signature-based IDS effectively detect known threats but fail against novel and evolving attacks [21] [22]. Anomaly-based IDS can identify unknown threats, but they suffer from high false positive rates, reducing reliability. Centralized IDS architectures introduce latency and scalability issues, making them impractical for large-scale IoT deployments [23].

ML has emerged as a promising approach for enhancing IoT security by enabling IDS to analyze network traffic, identify anomalies, and predict cyber threats [24] [25] [26]. Techniques, e.g., Decision Trees (DT), K-nearest neighbors (KNN), and Neural Networks, have demonstrated effectiveness in threat detection [27] [28]. However, ML-based IDSs face critical limitations, including high computational overhead, making them incompatible with resource-limited IoT devices [29, 30]. ML models also often depend on extensive labeled

training data, which may not always be available, and they struggle against adversarial attacks, where attackers modify behaviors to evade detection [31] [32] [33].
Deep Reinforcement Learning (DRL) offers a more dynamic and adaptive approach to IoT security [34] [35] [36]. In contrast to traditional ML models, DRL-based IDS can continuously learn and update security policies [37] [38] [39]. DRL enables automatic adaptation to evolving cyber threats, improving detection accuracy and system resilience. Moreover, it optimizes threat detection while balancing energy efficiency, making it suitable for resource-constrained IoT systems [40] [41]. Given the constraints of IoT devices, security solutions must minimize computational overhead while maintaining high detection accuracy [42] [43]. DRL-based IDS models provide an intelligent, adaptive solution for securing IoT deployments against large-scale cyber threats. Balancing security with computational efficiency aligns with global sustainability efforts, particularly the United Nations [44] Sustainable Development Goals (SDGs). Strengthening IoT security enhances digital infrastructure (SDG 9), ensures resilient and secure smart cities (SDG 11), and supports climate action by reducing the energy impact of cybersecurity mechanisms (SDG 13). This research integrates adaptive IDS models with energy-aware decision-making to improve cybersecurity resilience and ecological sustainability in IoT systems.

1.1 Problem Statements

Although significant advancements have been made in applying ML and DRL to IDS for the IoT, several critical challenges remain unresolved—particularly in real-world, resource-constrained edge gateways. To begin with, most existing IDS frameworks are not optimized for edge gateways, which are inherently limited in CPU capacity, memory, and energy. Traditional security mechanisms, such as firewalls and encryption, often impose high computational and energy overhead, making them unsuitable for real-time operation on these devices. This highlights the need for lightweight, energy-efficient IDS models to maintain strong detection capabilities without overwhelming the system. In parallel, the nature of cyber threats continues to evolve rapidly. Modern adversaries exploit zero-day vulnerabilities, launch large-scale botnet attacks (e.g., Mirai and Gafgyt), and even use adversarial ML to bypass conventional defenses. Signature-based IDS are ineffective against these unknown threats, while anomaly-based approaches often produce high false positive rates, limiting their reliability. Additionally, many ML-based IDS models rely on offline training, static features, and fixed datasets, making them ill-suited for adaptive threat detection and increasingly susceptible to concept drift in dynamic environments.
Compounding this issue is the computational complexity of many existing IDS models, which

makes them difficult to deploy on edge devices. These solutions frequently lack mechanisms for real-time policy enforcement and dynamic response—capabilities essential for timely threat mitigation. Although Software-Defined Networking (SDN) offers a promising foundation for scalable and adaptive defense, its integration with IDS remains underexplored, particularly regarding energy consumption, resource efficiency, and overall performance in edge contexts. Moreover, the effectiveness of most IDS approaches is typically validated using offline datasets or simulated environments, which fail to reflect the complexity of live IoT deployments. As a result, key operational metrics—such as CPU usage, memory demand, energy consumption, and carbon footprint—are often overlooked, even though they are critical for designing sustainable and scalable IoT security solutions. At the same time, although several IoT security design patterns exist (e.g., Whitelist, Blacklist, PZH), their impact on system performance and energy efficiency remains poorly understood—especially when applied in real-time, attack-prone scenarios. There is currently a lack of intelligent frameworks capable of dynamically selecting and adapting these patterns based on evolving network conditions. Equally important, most existing IDS solutions focus narrowly on threat detection and do not include automated or adaptive mitigation strategies. Without capabilities such as isolating compromised devices, filtering malicious traffic, or reconfiguring policies in real time, IDS remain passive and reactive, limiting their effectiveness in ensuring resilient and autonomous edge security.

In response to these challenges, this thesis proposes, implements, and validates a suite of DRL-based IDS frameworks specifically tailored for IoT edge environments. These include DeepEdgeIDS, AutoDRL-IDS, EdgeShield-DRL, and the self-adaptive SecuEdge-DRL. Together, they support real-time detection, dynamic response, intelligent security policy adaptation, and energy-aware operation. Through rigorous validation on real-world IoT-SDN testbeds and under live attack scenarios, this work advances scalable, sustainable, and high-performance IDS solutions that address the practical limitations of current systems and meet the growing demands of modern IoT ecosystems.

1.2 Contributions

To address the challenges identified in the problem statement, this research makes the following key contributions:

- **Development of a DRL-Based IDS for IoT Security**
  We propose four DRL-based IDS models (DeepEdgeIDS, AutoDRL-IDS, EdgeShield-DRL, and SecuEdge-DRL) for real-time cyber threat detection in edge gateway. The proposed solutions dynamically learn network behaviour, adapt to emerging attack pat-

terns, and optimize security policies, ensuring robust and adaptive defense mechanisms.

- **Evaluation and Optimization of IoT Security Patterns**
  We analyze six IoT security patterns (Personal Zone Hub, Trusted Communication Partner, Outbound-Only Connection, Blacklist, Whitelist, and Secure Sensor Node) and their effectiveness against cyber threats, e.g., DDoS, MITM, and Brute-force attacks. Furthermore, we assess the impact of these patterns on energy consumption and CPU usage, highlighting key trade-offs between security robustness and resource efficiency.

- **Development of a Dynamic Security Pattern Selection Framework Using DRL**
  We introduce an intelligent security framework that dynamically selects security patterns based on real-time network conditions. Using DRL, the system autonomously adapts security strategies to optimize resource efficiency, threat detection accuracy, and overall system resilience in the edge gateway.

- **Performance Analysis of ML-Based IDS at the IoT Edge**
  We assess the impact of seven ML-based IDS on CPU usage, CPU load, and energy consumption in real-time cyber threat scenarios at the edge gateway. Through ANOVA statistical analysis, we evaluate how integrating SDN enhances IDS efficiency and scalability.

- **Comparative Analysis of Supervised and Unsupervised DRL-Based IDS Models**
  This study systematically compares two distinct DRL-based IDS models. AutoDRL-IDS is a supervised model leveraging Long Short-Term Memory (LSTM)-based DRL for structured attack detection. Moreover, DeepEdgeIDS is an unsupervised model that integrates Autoencoders (AE) with DRL for behavior-driven anomaly detection. The evaluation provides insights into detection accuracy, adaptability, false positive rates, computational efficiency, and real-time threat mitigation.

- **Development of an IDS Test Suite**
  We develop a modular IDS test suite that allows researchers and practitioners to evaluate ML and DRL-based IDS under various IoT and SDN networks. This test suite comprises pre-configured datasets, pre-trained IDS models, and customizable attack scenarios for benchmarking. It facilitates comparative analysis and reproducibility. The test suite is publicly available for future research and development.

- **Green Computing and Energy-Efficient Security Mitigation**
  This dissertation enhances cybersecurity resilience by integrating energy-efficient IDS mechanisms that optimize CPU usage, memory consumption, and carbon emissions while maintaining high detection performance.

This thesis led to the publication/submission of the following research papers:

1. **S. Jamshidi**, A. Nikanjam, K. W. Nafi, F. Khomh, and R. Rasta, "Application of Deep Reinforcement Learning for Intrusion Detection in Internet of Things: A Systematic Review," *Internet of Things*, vol. 31, 2025.
2. **S. Jamshid**, A. Nikanjam, K. W. Nafi, and F. Khomh, "Understanding the Impact of IoT Security Patterns on CPU Usage and Energy Consumption: A Dynamic Approach for Selecting Patterns with Deep Reinforcement Learning," *International Journal of Information Security* 24.2 (2025): 1-40.
3. **S. Jamshidi**, A. Nikanjam, K. W. Nafi, and F. Khomh, "Comparative Analysis of Supervised and Unsupervised DRL-Based IDS for DDoS Detection at the Edge of the Internet of Things," *IEEE Internet of Things Journal*, Submitted, 2025.
4. **S. Jamshidi**, A. Nikanjam, K. W. Nafi, and F. Khomh, "Evaluating Machine Learning-Driven Intrusion Detection Systems in IoT: Performance and Energy Consumption," *Computers and Industrial Engineering*, Accepted, 2025.
5. **S. Jamshidi**, A. Nikanjam, K. W. Nafi, and F. Khomh, "A Dynamic Security Pattern Selection Framework Using Deep Reinforcement Learning," *IEEE International Conference on Software Services Engineering (SSE)*, Submitted, 2025.
6. **S. Jamshidi**, A. Amirnia, A. Nikanjam, K. W. Nafi, F. Khomh, and S. Keivanpour, "Self-Adaptive Cyber Defense for Sustainable IoT: A DRL-Based IDS Optimizing Security and Energy Efficiency," *Journal of Network and Computer Applications*, 2025.
7. **S. Jamshidi**, A. Nikanjam, K. W. Nafi, and F. Khomh, "Deep Reinforcement Learning-Based Intrusion Detection System: Defending Edge Gateways Against Mirai and Gafgyt," *IEEE international conference FiCloud*, Submitted, 2025.

1.3 Additional Publications

The following papers have been written during my PhD. While they all investigated ML-based IDS, they did not fit within the specific narrative of the thesis and were therefore excluded.

1. **S. Jamshidi**, A. Nikanjam, M. A. Hamdaqa, and F. Khomh, "Attack Detection by Using Deep Learning for Cyber-Physical Systems," *Artificial Intelligence for Cyber-Physical Systems Hardening*, Cham: Springer International Publishing, 2022, pp. 155-179.

2. **S. Jamshidi**, A. Amirnia, A. Nikanjam, and F. Khomh, "Enhancing Security and Energy Efficiency of Cyber-Physical Systems Using Deep Reinforcement Learning," *Procedia Computer Science*, vol. 238, pp. 1074-1079, 2024.

3. **S. Jamshidi**, A. Nikanjam, and F. Khomh, "Leveraging Machine Learning Techniques in Intrusion Detection Systems for Internet of Things," *Springer Handbook of Data Engineering*, 2025.

4. **S. Jamshidi**, N. Shahabi, K. W. Nafi, A. Nikanjam, and F. Khomh, "The Role of Large Language Models in IoT Security: A Systematic Review of Advances, Challenges, and Opportunities," *Internet of Things*, Submitted, 2025.

1.4 Thesis Organization

This dissertation is organized into ten chapters.

Chapter 1 introduces the research problem, motivation, and objectives, emphasizing using DRL to enhance intrusion detection capabilities in IoT edge gateways. Chapter 2 provides the foundational background on IoT architectures, key security challenges, and the role of IDS, along with essential concepts related to DRL and SDN. Chapter 3.1 presents a comprehensive literature review of existing DRL-based IDS approaches, identifying research gaps and motivating the need for adaptive, efficient, and real-time solutions. Chapter 4 evaluates the performance and energy efficiency of conventional ML-based IDS models deployed in IoT-SDN systems under real-time attack scenarios. Chapter **??** investigates the impact of six well-known IoT security patterns on CPU usage, energy consumption, and detection effectiveness across various threat vectors. Chapter 6 introduces a DRL-based framework that dynamically selects optimal security patterns at runtime, adapting to changing network conditions to improve threat detection and resource efficiency. Chapter 7 presents SecuEdge-DRL, a self-adaptive IDS framework built on the MAPE-K model, designed to enable continuous monitoring, automated decision-making, and sustainable threat mitigation at the edge. Chapter 8 details the development and evaluation of EdgeShield-DRL, a lightweight IDS targeting Mirai and Gafgyt malware, validated through real-world testbed experiments focusing on resource-aware deployment. Chapter 9 concludes the dissertation by summarizing key findings, contributions, and limitations and outlining potential directions for future research in sustainable, adaptive IoT security.

CHAPTER 2 BACKGROUND AND DEFINITIONS

2.1 Chapter Overview

This chapter provides an overview of IoT edge architecture, its security challenges, and the role of IDS. Furthermore, we discuss SDN and DRL approaches, e.g., Deep Q-Network (DQN) and Proximal Policy Optimization (PPO), to enhance security and resource management in IoT systems.

2.2 IoT Edge Architecture

IoT edge computing addresses latency issues by processing data closer to its point of origin. In addition to reduced latency, the IoT edge architecture enhances security and provides a smoother end-user experience. As shown in Figure 2.1, the architecture consists of key components, including IoT terminal devices (e.g., smart sensors), IoT edge gateways (e.g., Raspberry Pi, Intel NUC), and edge servers, which serve as small-scale computing centers. In our approach, we deploy lightweight detection models and techniques specifically customized to address the unique security challenges associated with these components and attacks.

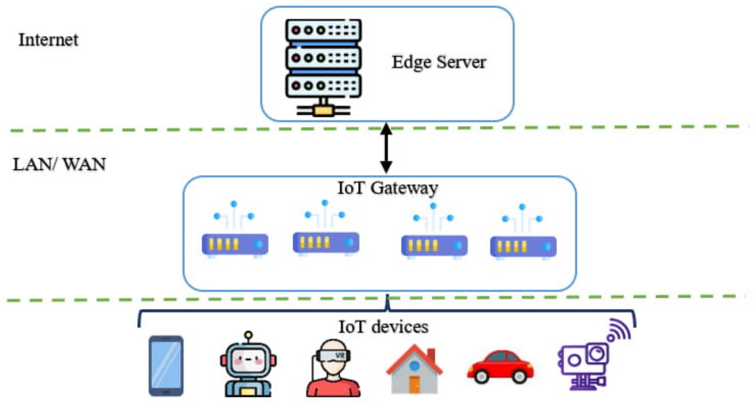

Figure 2.1 IoT edge architecture.

These detectors are implemented on each IoT edge gateway, enabling them to monitor and manage the connected IoT devices for data collection and basic processing tasks. The central computational hub's edge server has greater processing power and memory. This allows it to handle more complex calculations and analysis, with the findings being communicated back to the IoT edge gateway via various network interfaces. The gateway can sense and drive the IoT terminal devices to collect and process data. Moreover, the edge server, being the core processing and computing unit located at the edge of the IoT network, possesses more powerful computing capabilities and relatively rich memory resources [45] [46] [47]. Therefore, the edge server can handle complex calculation and analysis tasks more effectively and send the findings to the IoT edge gateway through various network interfaces. The resource constraints typical of IoT edge devices, e.g., limited energy and processing power, necessitate lightweight, energy-efficient solutions. Addressing these constraints is a technical challenge and a sustainability imperative, as inefficient systems contribute to increased carbon footprints in IoT ecosystems.

2.3 IoT Security

This section examines IoT architecture, layer-specific security challenges, IDS roles, and mitigation strategies [48]. The IoT connects billions of devices, facilitating data exchange and automation across industries. However, this connectivity introduces significant security challenges, necessitating robust IDS to safeguard IoT ecosystems [49]. IDS mitigates security breaches by analyzing network traffic, device behavior, and system events.

2.3.1 IoT Architecture, Security Challenges, and the Role of IDS

IoT systems consist of three primary layers: the Perception Layer, the Network Layer, and the Application Layer. Each layer faces unique security threats, and IDS plays a vital role in identifying and mitigating these risks [50] [51].

2.3.2 Perception Layer: Physical Interaction and Data Collection

The Perception Layer integrates sensors, actuators, RFID tags, and embedded systems to collect real-world data. As IoT devices operate on limited resources (i.e., lower capacity batteries, lower memory, etc.), they are easily vulnerable to threats, e.g., physical tampering, where attackers manipulate devices to disrupt operations [52]. Mitigation strategies, i.e., tamper-resistant hardware to prevent unauthorized physical access and IDS with real-time anomaly detection to identify irregular network behavior [53], provide essential defenses against cyber threats. Data injection and spoofing, where false data misleads systems, can destabilize infrastructures, e.g., smart grids. Anomaly-based IDS effectively detects such deviations. Firmware exploitation, enabling malware injection, is countered using secure boot and signature-based IDS. Lightweight IDS solutions are crucial for monitoring data streams and maintaining robust security without overburdening devices [54] [55].

Network Layer: Communication Backbone

The Network Layer manages data transmission between devices, gateways, and cloud systems, exposing it to threats targeting communication protocols, traffic flows, and routing mechanisms. MITM attacks intercept and alter data during transmission. Encryption-aware anomaly-based IDS detects unauthorized communication attempts [56] to solve Man-in-the-Middle attacks. The DDoS attacks, e.g., the Mirai botnet, overwhelm network resources using compromised IoT devices, which can be mitigated through distributed IDS for edge device monitoring and traffic filtering. Routing attacks that manipulate routing tables are countered with signature-based IDS to detect known anomalies and anomaly-based IDS for suspicious patterns. Replay attacks, reusing legitimate packets for unauthorized actions, are mitigated through timestamping and IDS to flag duplicate packets. IDS functionality includes real-time ML traffic monitoring, lightweight IDS at edge nodes, and protocol-specific IDS tuned for MQTT, CoAP, and other IoT protocols to enhance detection accuracy [57] [58].

2.3.3 Application Layer: Data Processing and Service Delivery

The Application Layer processes data for services such as predictive maintenance, remote monitoring, and analytics, interacting with users and external systems. This exposes it to threats, e.g., malware and ransomware, and exploits software vulnerabilities to inject malicious code [59]. Different techniques are already in practice to solve these problems. Among them, Signature-based IDS detects known malware, while behavioral IDS identifies unusual application behavior [60]. Data privacy breaches involving sensitive information, e.g., health or financial data, are mitigated through AES-256 encryption [61] and IDS to detect unauthorized access attempts. Privilege escalation, where attackers exploit weak authentication to gain administrative control, is countered using behavioral IDS to monitor and prevent unauthorized access patterns [62]. IDS at this layer also monitors anomalies, integrates with firewalls and patch management systems, and detects API exploitation through behavior analysis and rule-based monitoring [63].

2.3.4 Cross-Layer Threats and IDS Deployment

Some security threats span multiple IoT layers, requiring advanced IDS. Advanced Persistent Threats (APTs) involve prolonged infiltration using reconnaissance and zero-day vulnerabilities. AI-driven IDS detect these subtle, long-term attack patterns. Insider threats, where authorized users misuse privileges, are addressed with behavioral analytics-based IDS to detect deviations in user activity. Supply chain attacks from compromised hardware or software during manufacturing are mitigated by IDS monitoring hardware integrity and firmware updates for abnormal behavior [64] [65] [66].

2.4 Software-Defined Networking (SDN) in IoT Security

The proliferation of IoT devices has introduced significant security challenges due to the vast number of connected endpoints, the heterogeneous nature of IoT networks, and the increasing sophistication of cyber threats [67]. Traditional network architectures, which rely on static configurations and decentralized security management, struggle to address these challenges effectively. SDN provides a transformative solution by decoupling the control and data planes, enabling centralized network management, dynamic security enforcement, and real-time threat mitigation. By leveraging SDN, IoT security is significantly enhanced through automated threat response, network segmentation, traffic visibility, and policy-driven security enforcement [68].

A primary advantage of SDN in IoT security is its ability to provide centralized threat detection and automated response. Traditional networks rely on distributed control mechanisms and manual configurations, making responding to emerging threats in real-time complex. On the other hand, SDN continuously monitors network traffic and detects security anomalies, such as DDoS attacks, malware propagation, and unauthorized access attempts. By integrating with Intrusion Detection and Prevention Systems (IDPS) and AI-driven threat intelligence, SDN controllers can automatically apply countermeasures, including blocking malicious IP addresses, rerouting network traffic, or isolating compromised devices. This automation significantly reduces response time and enhances the overall security posture of IoT networks [69] [70].

Another critical aspect of SDN's role in IoT security is its ability to enforce network segmentation and attack containment. Unlike traditional segmentation methods that rely on static VLANs or firewall rules, SDN enables dynamic micro-segmentation, which isolates devices and applications into logically separated network zones. This approach limits an attacker's ability to move laterally within the network, thereby reducing the potential impact of a security breach. For instance, in industrial IoT (IIoT) networks, SDN can segment operational technology (OT) networks from general IT networks, ensuring that critical infrastructure remains secure even if an IoT endpoint is compromised. Similarly, in healthcare IoT, SDN can isolate medical devices such as infusion pumps and patient monitors from administrative systems, preventing cyber threats from affecting life-critical operations [71].

SDN also improves IoT security by providing enhanced traffic visibility and anomaly detection. Traditional IoT networks often lack comprehensive monitoring capabilities, making

detecting and preventing sophisticated cyber threats difficult. SDN, however, offers a centralized, real-time view of network traffic, allowing security administrators to analyze data flows and identify suspicious behavior [72]. By integrating with big data analytics and machine learning models, SDN can detect unusual network activities, such as unauthorized data exfiltration, unexpected communication patterns, and connections to blacklisted domains. This increased visibility enhances early threat detection and enables proactive security measures to prevent attacks before they escalate.
Another key advantage of SDN in IoT security is its ability to enforce policy-driven security controls. In contrast to traditional networks, where security policies must be manually configured across multiple devices, SDN enables centralized, automated policy enforcement [73]. Security administrators can define access control policies that dynamically adapt to changing network conditions and emerging threats. For example, SDN can enforce context-aware access controls, where device permissions are granted based on their security posture, location, and role within the network. SDN integrates seamlessly with Security Information and event management systems, enabling continuous monitoring and automatic policy updates based on real-time threat intelligence.

2.4.1 Machine Learning in IoT Security

ML has become a powerful tool for securing IoT systems by enabling intelligent threat detection, anomaly identification, and automated response mechanisms. Unlike traditional security approaches that rely on predefined rules and static configurations, ML-driven IoT security systems continuously learn from network behavior, making them adaptable to evolving cyber threats. ML models play a crucial role in IDS by analyzing traffic patterns and distinguishing between normal and malicious activities. Supervised learning algorithms, e.g., DT and RF, classify IoT network traffic to detect anomalies. In contrast, Deep Learning (DL) models, e.g., Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks, identify sophisticated attack patterns, including zero-day threats. Additionally, unsupervised learning techniques, e.g., clustering and autoencoders, enhance security by detecting unknown attack vectors without prior labeled data. RL further strengthens IoT security by enabling adaptive decision-making for automated attack mitigation, optimizing firewall rules, and managing network access dynamically. Despite its advantages, ML in IoT security faces several limitations. One of the significant challenges is the resource constraints of IoT devices, which often lack the computational power and memory to execute complex ML models in real-time. Deploying lightweight, edge-based ML solutions requires optimized architectures and Federated Learning (FL) approaches. Data quality and availability are also critical concerns, as ML models require large, diverse datasets for practical training, yet

IoT networks often generate fragmented and imbalanced security data. The susceptibility to adversarial attacks is another major limitation, as attackers can manipulate ML models by injecting poisoned data, leading to false negatives in threat detection. Moreover, the lack of interpretability in DL models raises concerns in critical IoT applications, where security decisions must be explainable and verifiable. Addressing these challenges requires ongoing research in lightweight ML models, adversarial defense mechanisms, and privacy-preserving learning techniques to enhance the resilience of ML-driven IoT security solutions.

Decision Trees (DT)

In IDS, DT is a key ML method for analyzing network data. They use trees, e.g., models, to break down network features into binary decisions, evaluating network attributes at each node to identify effective splits. This creates a rule-based hierarchy that excels at spotting differences between normal and suspicious network activities. DTs are valued for their clarity and ease of interpretation, playing a vital in improving cybersecurity by identifying unusual or unauthorized actions [74] [75].

Random Forest (RF)

The algorithm is highly valued in IDS for its precision in classifying network data. Utilizing RF, an ML algorithm, it creates a group of DT to assess various network attributes, effectively distinguishing between normal and malicious activities. RF excels in managing large datasets, balancing IDS data disparities, and minimizing overfitting, making IoT and network security crucial. It achieves accurate detection of unusual network behaviors [76] [77].

k-Nearest Neighbors (KNN)

The KNN algorithm is a key IDS tool known for its effective similarity-based classification. It compares network traffic with existing labeled data using distance metrics to classify new instances, with 'k' indicating the number of neighbors considered. This method is crucial for identifying regular versus abnormal network activities, offering a simple yet versatile solution for real-time IDS. KNN excels in both binary and multiclass problems, providing quick, reliable categorizations crucial for responding to threats in dynamic networks [78] [79] [80].

Long Short-Term Memory (LSTM)

LSTM networks, a type of recurrent neural network, are highly effective in analyzing sequential data for IDS. Their unique memory cells excel at identifying complex patterns in network traffic, making them adept at spotting advanced threats that traditional methods may miss. LSTMs are especially valuable for maintaining context over data sequences, which is crucial for distinguishing between normal and malicious network activities. Their application in IDS significantly boosts cybersecurity, especially in dynamic and IoT networks, by adapting to new threats and efficiently handling varying data lengths, offering a robust solution to modern cybersecurity challenges [81] [82].

Convolutional Neural Network (CNN)

CNNs provide a resilient DL methodology for IDS. CNNs are widely recognized for their ability to independently acquire hierarchical features from network traffic. This is achieved through convolutional, pooling, and fully connected layers, which enable the discernment of spatial patterns in the traffic data. This capacity facilitates the recognition of both well-established and new threats. CNN in IDS is crucial in enhancing cybersecurity defenses against a wide range of cyber threats due to their capacity to scale effectively and efficiently handle real-time data [83] [84].

Hybrid model of LSTM and CNN

Integrating LSTM and CNN models into IDS significantly boosts network security by combining the spatial analysis capabilities of CNNs with the temporal pattern recognition of LSTMs. This hybrid approach detects complex cyber threats by analyzing network traffic data in both spatial and temporal dimensions. CNNs effectively identify security breaches through local pattern recognition, while LSTMs track the sequence of network events over time, offering a detailed understanding of potential threats. This fusion results in more accurate and efficient detection of sophisticated, multi-stage attacks, reducing false positives and adapting to new threats, thereby enhancing overall anomaly detection and maintaining network integrity without excessive alerts [85] [86].

Enhanced Intrusion Detection Deep learning Multi-class Classification Model (EIDM)

EIDM is a cutting-edge IDS approach that expertly handles many network events. Its design combines convolutional and dense layers to tackle the challenges of class diversity and data imbalance. The model begins with a 120-node dense layer, followed by an 80-neuron convolutional layer with a kernel size of 20 to better distinguish between similar network activities. It also features a Maxpooling layer for enhanced feature extraction and a dropout layer to avoid overfitting. EIDM can classify 15 network behaviors through six dense layers, using 'relu' activation and SGD and Adam optimizers for optimal accuracy and efficiency. According to [87], EIDM's unique structure and optimization techniques make it a standout solution for improving network IDS.

2.5 DRL for IoT Security

DRL leverages artificial intelligence (AI) to optimize security mechanisms in complex and dynamic IoT networks. By combining DL with RL, DRL-based security systems can continuously adapt to evolving cyber threats, making them highly effective for securing IoT networks. A key application of DRL in IoT security is its role in IDS, where DRL models detect anomalies by learning patterns of normal and malicious traffic.

Traditional IDS solutions rely on static rules and signature-based detection, rendering them ineffective against zero-day attacks and sophisticated threats. DRL-based IDS, however, continuously learns from real-time network traffic, allowing it to identify new and previously unseen attack patterns. By interacting with the network environment and refining its decisions based on feedback, a DRL agent enhances its ability to detect and mitigate threats dynamically. Unlike static rule-based IDS, DRL models generalize across attack vectors and adapt to evolving adversarial strategies.

Beyond detection, DRL enables automated attack mitigation and response. DRL agents can be trained to execute real-time security decisions, such as blocking malicious traffic, isolating compromised devices, or reconfiguring network policies to prevent further exploitation. This autonomous defence mechanism reduces human intervention, minimizes response time, and enhances overall security. In IoT networks, DRL-based security frameworks protect critical infrastructure from cyber-physical attacks targeting industrial control systems (ICS) and OT networks.

DRL enhances access control and authentication by dynamically managing security policies based on real-time contextual data. Unlike traditional access control mechanisms that rely on static credentials, DRL models assess user behavior, device trust levels, and network conditions to grant or restrict access dynamically. This adaptive approach strengthens IoT security by mitigating unauthorized access and insider threats. Additionally, DRL optimizes IoT network traffic management and policy enforcement. Given IoT deployments' large-scale and heterogeneous nature, ensuring seamless network performance while maintaining robust security is challenging. DRL-based models dynamically allocate resources, optimize network configurations, and enforce security policies based on real-time threat analysis. This is particularly beneficial in edge computing environments, where computational resources are limited, and security decisions must be made locally without reliance on centralized cloud processing.

2.5.1 Deep Q-Network (DQN)

In this study, we adopt the DQN architecture to model the IDS framework described in Section 7.4, following the approach introduced by [88]. DQN, the RL model introduced by [89], uses two distinct deep neural networks, the prediction network and the target network, both employed to approximate Q-values. The prediction network estimates $Q(s, a)$, representing the Q-values for all potential actions in the current state. Meanwhile, the target network calculates $Q(s', a')$, corresponding to the next state's Q-values. Using these dual networks helps mitigate the instability often encountered in RL training by decoupling the updating process.

DQN aims to converge to the optimal Q-function, as described in Equation (2.1). The optimal policy π maximizes the expected cumulative future rewards by selecting an action a_t at state s_t. The parameter γ, known as the discount factor, controls the trade-off between immediate and long-term rewards. The Q-values are updated recursively based on the Bellman equation [90], as shown in Equation (2.2), where α is the learning rate.

$$Q^*(s,a) = \max_{\pi} \mathbb{E}\Big[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \dots \quad \mid s_t = s, a_t = a, \pi\Big] \tag{2.1}$$

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[R(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a)\right] \tag{2.2}$$

DQN minimizes a loss function to optimize the learning process, as shown in Equation (2.3). In this equation, θ represents the parameters of the prediction network, while θ^- refers to the parameters of the target network. Instead of updating the prediction network by computing the entire summation, stochastic gradient descent (SGD) is used to adjust its parameters iteratively. Additionally, the target network's parameters are periodically updated from the prediction network's parameters after every T_{target} steps, improving the stability of training.

$$L(\theta) = \sum_{s,a,r,s'} \left(r + \gamma \max_{a'} Q(s',a';\theta^-) - Q(s,a;\theta)\right)^2 \tag{2.3}$$

This loss function ensures that the prediction network minimizes the difference between its estimated and target Q-values, allowing it to approximate the true Q-values accurately. By periodically synchronizing the target network with the parameters from the prediction network, the DQN stabilizes the learning process, reducing oscillations and divergence throughout the training.

2.5.2 Importance of Energy Consumption, CPU Usage, and CPU Load

Energy consumption is a critical factor for IoT devices [91] [92], especially those in edge computing where power is limited. Low energy use is vital for extending the device's lifespan and ensuring consistent performance in energy-constrained situations. Excessive energy use can lead to frequent downtimes and reduced reliability, which is especially problematic in critical IoT applications.

Similarly, CPU usage and CPU load are crucial indicators of how efficiently a device operates. High CPU usage can slow down processes and cause delays, disrupting real-time monitoring and automated systems that depend on quick responses. On the other hand, CPU load shows how well a device manages multiple tasks simultaneously, which is essential for maintaining overall system stability and ensuring that security measures do not interfere with other important functions. By closely monitoring these metrics and analyzing their impact, we can ensure that the DRL-based IDS detects and responds to threats effectively and uses resources efficiently. Achieving this balance is essential for deploying security solutions in the resource-limited hardware typical of many IoT applications.

2.5.3 Proximal Policy Optimization (PPO)

PPO is a policy-based DRL method designed to improve training efficiency and stability in IDS for IoT security. Unlike value-based methods like DQN, which rely on estimating Q-values, PPO optimizes the policy function, making it well-suited for continuous and dynamic IoT security challenges.

PPO enhances real-time threat detection and resource allocation by refining policy updates through a clipped surrogate objective, preventing drastic policy changes that could destabilize training. This ensures more stable convergence and prevents the oscillations commonly encountered in other policy gradient methods. PPO achieves this by limiting the step size in policy updates, which helps balance exploration and exploitation, improving learning efficiency. The PPO objective function is defined as follows:

$$L(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta) A_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon) A_t \right) \right] \tag{2.4}$$

A key advantage of PPO in IDS applications is its ability to dynamically adapt to evolving attack patterns. Since IoT networks are highly heterogeneous and experience frequent changes in traffic patterns, traditional IDS solutions struggle to generalize across different attack types. PPO-based IDS models continuously adjust their policies based on real-time feedback, allowing them to respond more effectively to new and unseen threats. By interacting with the environment and optimizing policy updates iteratively, PPO improves the system's ability to detect sophisticated cyber threats, including zero-day attacks and adversarial intrusions.

Beyond IDS, PPO also plays a crucial role in resource-efficient security enforcement. IoT devices operate under stringent computational and energy constraints, making it essential for IDS solutions to balance security performance with resource utilization. PPO-based models can dynamically allocate system resources—such as CPU cycles, memory, and energy—based on the severity and likelihood of security threats. PPO ensures that IoT security mechanisms remain efficient and lightweight by prioritizing critical threats while minimizing unnecessary computations.

Another key benefit of PPO in IoT security is its scalability in edge and cloud environments. Unlike traditional IDS frameworks that rely on centralized processing, PPO can be deployed in decentralized edge computing environments, where security decisions must be made with minimal latency. PPO's ability to optimize decision-making at the edge ensures faster response times and reduces dependency on cloud-based processing, making it suitable for real-time IoT security applications.

2.5.4 IoT patterns

This section examines the core concepts of six security patterns and their importance in securing the intracoastally and interconnectively IoT-edge gateway. Implementing the Stuttgart patterns [1] was considered essential, as they provide a comprehensive set of IoT security patterns [93]. Moreover, the Stuttgart patterns address critical threats with robust protection and resource efficiency tailored for IoT systems. They simplify complex implementations through abstraction and ensure adaptability in heterogeneous ecosystems with context-specific customization. These patterns integrate seamlessly into secure development methodologies, providing systematic security throughout the lifecycle. Their effectiveness optimizes security for resource-constrained devices, making them practical for diverse IoT systems. This blend of abstraction, adaptability, and efficiency highlights their value in advancing secure IoT systems [1]. Additionally, we integrated the SSN pattern [94] to ensure secure data transmission between the server, user, and the edge gateway, along with its attached sensors.

2.5.5 Personal Zone Hub (PZH)

The growing prevalence of IoT devices has presented users with a concomitant increase in the complexity of managing these devices' permissions, data sharing, and control across various gateways and cloud systems. To address these challenges, PHZs have emerged as a viable

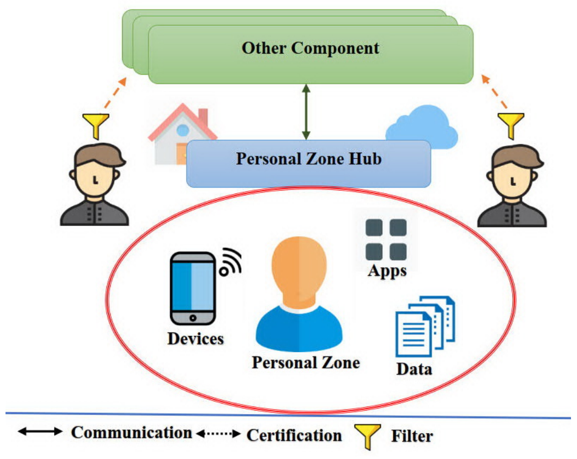

Figure 2.2 PZH pattern solution sketch [1].

solution. PHZs enable users to effectively manage, control, and integrate their devices, services, applications, and workflows. Furthermore, PHZs offer the added advantage of creating a permanently addressable hub, allowing users to selectively share access to different parts of the data and functionality encompassed by the PHZs. As a result, PHZs represent a promising solution to the challenges associated with managing and controlling IoT devices in a network.

2.5.6 Trusted Communication Partner (TCP)

In a dynamic environment, a device may have multiple communication options available. However, not all of these communication partners may be familiar or reliable. Some may even pose a security threat; attackers may exploit these communication media to gain unauthorized access to the device and its network. To mitigate such risks, it is recommended to configure the device with a single trusted communication partner or a list of dependable communication partners. This pattern enables secure incoming and outgoing communication with these trusted partners, restricts other communication attempts, and promptly alerts the designated person in charge for further investigation.

2.5.7 Outbound-Only Connection (OOC)

The proliferation of IoT devices has created a new avenue for malicious actors to access individuals' personal and sensitive information. Cybercriminals often exploit vulnerabilities in these devices by sending unsolicited communication requests that trick the device into connecting to a compromised network. To mitigate such risks, many IoT devices are programmed

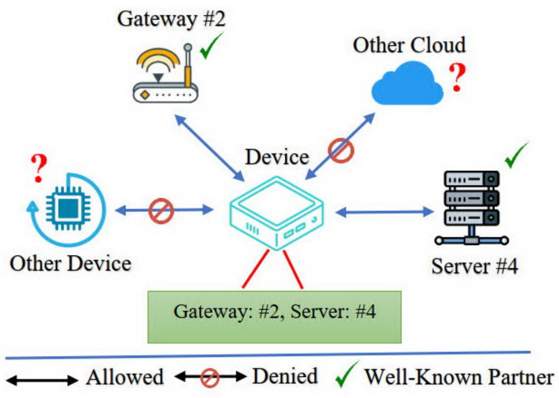

Figure 2.3 Trusted Communication Partner pattern solution sketch [1].

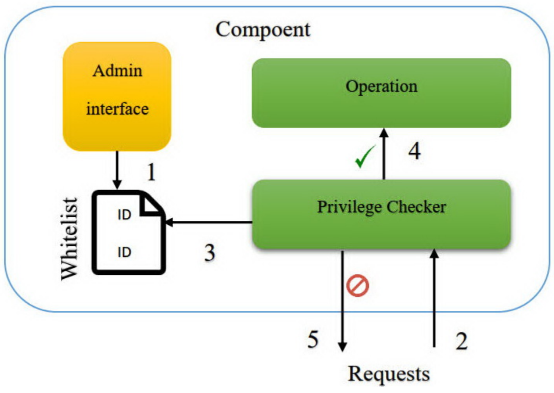

Figure 2.4 Whitelist pattern solution sketch [1].

only to allow authorized connections, thereby preventing unauthorized access attempts. This controlled connection OOC pattern allows the devices to initiate and maintain connections without rejecting all incoming communication requests without proper authorization.

2.5.8 Blacklist (BL)

Privileges, whether they are freely given or explicitly granted, are susceptible to abuse. Establishing measures to prevent ongoing abuse and restrict potential future abuse is imperative. To achieve this, maintainers frequently create a BL containing records of individuals identified as abusive users or who misused their privileges. By maintaining such a list, the maintainers can better manage the privileges and ensure they are only granted to trustworthy individuals.

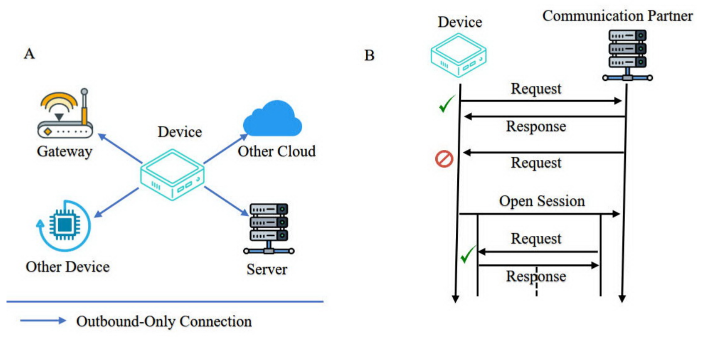

Figure 2.5 Outbound-Only Connection pattern solution sketch [1].

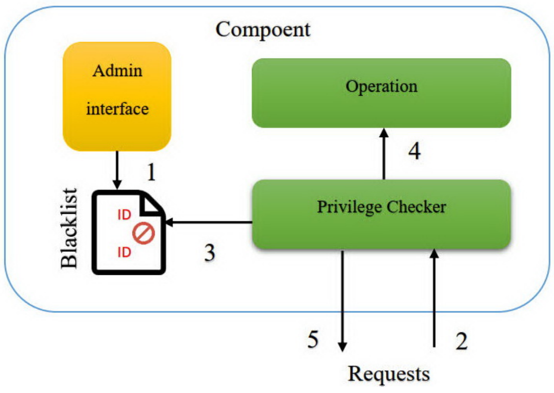

Figure 2.6 Blacklist pattern solution sketch [1].

2.5.9 Whitelist (WL)

The WL functions in contrast to the BL, which blocks known abusive communication partners. However, while the BL is incomplete, the WL limits the number of possible attacks by providing an administrative interface to add communication partner identifiers. This interface enables the checking of privileges that the WL controls. When a communication partner on the WL requests a privilege, the system can proceed. Conversely, the system automatically denies the request if the requester is not on the WL. Thus, the WL serves as an effective means of limiting access to known communication partners.

2.5.10 Secure Sensor Node (SSN)

A sensor node seamlessly performs data collection and transmission tasks in a sensor network. However, suppose the sensor node lacks security mechanisms. In that case, it may be vulnerable to various security issues, such as privacy, integrity, and confidentiality. This can compromise the data stored, transmission, and the environment it interacts with. The

SSN pattern tells a sensor node how to securely send and store data in a WSN, regardless of its topology. A secure sensor node can encrypt data in communications and storage while keeping a record of events for auditing purposes. As the sensor node covers other aspects of the node without special security considerations, it has no mechanisms or design decisions. To deliver security features to a sensor node, it is necessary to apply additional components to encrypt data, log events, and perform authentication and authorization processes with other WSN nodes.

2.6 Resource Efficiency, Response Time, and Sustainability in DRL-based IDS

Several critical factors govern the performance, scalability, and sustainability of DRL-based IDS, including carbon emissions, CPU usage, energy consumption, memory usage, and IDS response time for detecting attacks. These factors play a pivotal role in ensuring the practical viability of IDS implementations, particularly in resource-constrained edge gateways.
Carbon emissions, as a byproduct of energy consumption, provide a quantifiable measure of the environmental footprint associated with the operation of the IDS. Designing energy-efficient systems that adhere to green computing principles is imperative for promoting sustainability in IoT deployments, often characterized by extensive and distributed networks.
CPU usage indicates the computational demand placed on the edge gateway by the IDS. High CPU utilization affects the system's real-time responsiveness and hinders scalability, leading to potential performance bottlenecks and delayed attack detection. Maintaining low computational overhead is essential for ensuring reliable operation in dynamic IoT settings.
Energy consumption is a particularly significant concern for resource-constrained edge gateways, where inefficiencies can rapidly deplete power reserves, increase operational costs, and undermine system reliability. Optimizing energy usage is critical to extending the operational lifespan of IoT devices, especially in scenarios relying on battery-powered gateways.
Memory utilization reflects the ability of the IDS to process data streams and perform real-time threat detection without exceeding the hardware constraints of edge devices. Efficient memory management is crucial for maintaining system stability and ensuring consistent performance, even during high workloads or peak traffic conditions.
Response time and latency are fundamental metrics for assessing the effectiveness of an IDS in detecting and mitigating threats. Response time refers to how quickly the IDS can detect an attack and trigger mitigation actions, while latency represents the delay introduced by the IDS in processing network traffic. Minimizing response time is critical in IoT networks where cyberattacks, e.g., DDoS, can rapidly disrupt services and cause operational failures. A slow IDS may allow attackers to exploit vulnerabilities before appropriate countermeasures

are enacted, leading to severe consequences such as device takeovers, data breaches, and service downtime.
Similarly, high latency negatively impacts real-time IoT applications, such as autonomous vehicles, industrial automation, and healthcare monitoring systems, where even slight delays in decision-making can cause catastrophic failures. Thus, an effective IDS must balance detection accuracy and computational efficiency to ensure real-time threat mitigation while maintaining low processing overhead [95] [67]. This is especially important for resource-constrained edge gateways, where excessive response time or latency may degrade overall system performance. An IDS can provide timely, efficient, and scalable security solutions for protecting IoT networks against evolving cyber threats by optimizing response time and latency.

2.6.1 DDoS Probability in IoT Edge Systems

DDoS probability refers to the likelihood of an IoT system experiencing a DDoS attack. This probability depends on various factors, including network vulnerability, traffic anomalies, and the presence of botnets. Due to their low computational resources and lack of robust security mechanisms, IoT devices are highly likely to be targeted by attackers. Understanding and predicting DDoS probability is crucial for developing proactive defense mechanisms and enhancing the resilience of IoT edge systems.

2.7 Chapter Summary

This chapter provided the foundational concepts for understanding the proposed IDS framework in IoT edge computing. It began by introducing the IoT edge architecture and its core components, emphasizing the need for lightweight, energy-efficient security mechanisms due to the constrained nature of edge gateways. A detailed analysis of IoT security challenges was presented across the perception, network, and application layers, along with the role of IDS in mitigating various threats such as data injection, DDoS, MITM, and insider attacks. The chapter also explored the value of SDN in enhancing IoT security through centralized management, real-time threat mitigation, and policy enforcement. ML techniques,e.g., supervised, unsupervised, and DL methods, were discussed for their effectiveness in intrusion detection. Specific algorithms such as Decision Trees, Random Forest, KNN, CNN, and LSTM were introduced, along with hybrid models and advanced architectures, e.g., EIDM.
DRL was presented as a powerful solution for adaptive, autonomous IoT security. The mechanics of DQN and Proximal PPO were explained in detail, showcasing their capabilities in real-time anomaly detection, policy optimization, and energy-aware threat response. Addi-

tionally, the chapter highlighted six IoT security patterns, e.g., PZH, TCP, OOC, BL, WL, and SSN, that support secure data handling and communication in edge systems. Finally, critical performance considerations, e.g., CPU usage, memory consumption, response time, energy efficiency, and sustainability, were examined in relation to DRL-based IDS. Minimizing latency and carbon footprint in real-time IoT environments was emphasized, reinforcing the chapter's relevance to technical performance and environmental responsibility.

CHAPTER 3 ARTICLE 1: APPLICATION OF DEEP REINFORCEMENT LEARNING FOR INTRUSION DETECTION IN INTERNET OF THINGS: A SYSTEMATIC REVIEW

3.1 Article Metadata

- **Title:** Application of Deep Reinforcement Learning for Intrusion Detection in Internet of Things: A Systematic Review
- **Authors:** Saeid Jamshidi, Amin Nikanjam, Kawser Wazed Nafi, Foutse Khomh, Rasoul Rasta
- **Affiliations:**
  - SWAT Laboratory, Polytechnique Montréal, H3T 1J4, Quebec, Canada
  - Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran
- **Submitted Date:** 6 September 2024
- **Accepted Date:** 4 February 2025
- **DOI:** `https://doi.org/10.1016/j.iot.2025.101531`
- **Journal:** *Internet of Things*

3.2 Chapter Overview

This chapter reviews previous studies on applying DRL to IDS in IoT systems. We analyze the challenges of securing IoT networks, traditional IDS limitations, and DRL's potential for adaptive threat detection. Additionally, we examine benchmarking methods, dataset utilization, and scalability concerns while identifying key research gaps.

3.2.1 Deep Q Network (DQN)

Yucheng Liu et al. [96] proposed a multi-layered defense system based on DQN to adjust the IP blocking time in IoT systems dynamically. This system has the advantage of reducing the false positive rate to a great extent and not disrupting the service due to the wrong threat identification. This study achieved an accuracy of 96% in single-layered attack scenarios and 97% in multi-layered attack scenarios. Underlying such practical utility and potential for broader adoption in IoT security is adherence to IEEE P2668 standard, with consequences for scalable applications in the current emerging Metaverse.

Xing Liu et al. [97] explored the application of DRL to the Industrial Internet of Things (IIoT), presenting a system design that evaluates vulnerabilities to adversarial attacks. The study categorized the attacks into two main types: function-based attacks that disrupt the DRL model during training and performance-based attacks that manipulate system operations post-training. Empirical results from simulations demonstrate that these attacks significantly impair system functionality, highlighting the need for robust security measures in IIoT systems.

T.V. Ramana et al. [98] proposed an advanced IDS framework utilizing the RL-DQN model for IoT systems. This dual-layer security system operates across edge and cloud layers, with the edge network achieving a binary attack classification accuracy of 92.8% and the cloud network performing multi-attack classification with an accuracy of 98.2% on the UNSW-NB-15 dataset. The framework employs Markov Decision Processes (MDP) to dynamically adapt to varying network conditions, thereby enhancing the robustness of IDS. The RL-DQN model outperformed traditional ML models, including Random Forest (86.9% accuracy), Gradient Boosting Machine (88.3% accuracy), and Convolutional Neural Networks (90.9% accuracy) on similar datasets. The RL-DQN model demonstrated a recall rate of 98.7% and a precision rate of 93.2%, highlighting its superiority in detecting and classifying intrusions across various datasets, including the IoTID20 dataset.

Sreekanth Vadigi et al. [99] introduced an innovative IDS that leverages FL combined with DRL, specifically employing DQN across distributed agents in IoT enterprise networks. The system incorporates a dynamic attention mechanism to enhance detection accuracy while preserving data privacy across the network. Experimental evaluation of the proposed system on the ISOT-CID and NSL-KDD datasets demonstrated impressive results. The system achieved an accuracy of 99.66% on the ISOT-CID dataset and 96.7% on the NSL-KDD dataset, coupled with very low false positive rates of 0.0017 and 0.019, respectively. These performance metrics surpass conventional IDS approaches, such as Support Vector Machines (SVMs) and DT, which typically exhibit lower accuracy levels, often between 85% and 90%.

Xiaoxue Ma et al. [100] proposed a decision-making framework for intrusion response in fog computing environments, utilizing a Minimax-DQN variant. This adaptation of DQN optimizes strategies to respond to cyber-attacks by modeling the interactions between IDS and attackers as a Markov game. Their model significantly enhances the strategic decision-making capabilities of fog computing environments, effectively demonstrating the practical

application of DRL in mitigating cyber risks. The experimental results validate the model's efficacy, showing that the proposed algorithm substantially increases the IDS's probability of winning against attackers, achieving a win rate of 94.6% against random strategies and outperforming standard DQN setups by 3.7% Zhang et al. [101] introduced a defense mechanism for IoT edge devices using DRL in IoT, a zero-sum game framework. They enhance the interaction model between attackers and edge devices by integrating a modified KNN algorithm with Dynamic Time Warping (DTW). This approach analyzes network traffic and data interactions, crucial for identifying vulnerabilities. The strategy optimizes defensive responses, adapting to evolving IoT challenges and improving device resilience against cyber threats. Empirical results validate the model's effectiveness, showing decreased attacker payoffs and enhanced security metrics. This leads to a notable reduction in successful attacks, demonstrating the practical utility of their approach in real-world scenarios.
Mohammad Al-Fawa'reh et al. [102] introduced a sophisticated detection system for malware botnets in IoT networks named MalBoT-DRL. The system leverages DRL to manage the evolving patterns of malware dynamically. It incorporates damped incremental statistics, specifically mean, variance, and covariance of network traffic features, along with an attention reward mechanism in IoT. This combination enhances the system's adaptability and generalizability, effectively addressing the issue of model drift. Performance evaluations of MalBoT-DRL, conducted through trace-driven experiments on two representative datasets, demonstrate its efficacy, with the system achieving an average detection accuracy of 99.80% in the early detection phase and 99.40% in the late detection phase.
By adopting traditional DRL frameworks, Manuel Lopez-Martin et al. [103] replace the interactive real-time system with a simulated environment that utilizes pre-recorded datasets of intrusions, specifically NSL-KDD and AWID. This method generates rewards based on detecting errors during training, effectively leveraging supervised learning techniques. The researchers evaluated several DRL models, including DQN (96.87% accuracy), Double Deep Q-Network (DDQN) (97.85% accuracy), Policy Gradient (PG) (94.50% accuracy), and Actor-Critic (AC) (95.12% accuracy), with DDQN emerging as the best-performing model. The study highlighted that DRL can significantly enhance IDS capabilities, offering improvements in both speed and accuracy over conventional ML techniques.

3.2.2 Modified DRL (MDRL)

Almasri et al. [104] suggested a new two-phase security solution to increase the level of safety for IoT devices in smart cities. In the first phase, the system uses a cascaded adaptive neuro-fuzzy inference system to detect anomalies in network traffic from possibly compromised devices, issuing an alarm to system administrators. Subsequently, the MDRL strategy isolates the compromised devices, cutting off their network communication to prevent further damage. This approach achieved a very high accuracy of about 98.7%, with minimal false alarm rates, and hence, guarantees smooth operation and effective delivery of smart city network services.

3.2.3 Deep Q-Learning DQL)

Jiushuang Wang et al. [105] introduced ReLFA, an advanced IDS specifically designed to counteract Link Flooding Attacks (LFA) in SDN for IoT systems. ReLFA utilizes Rényi entropy to detect anomalies in network traffic, effectively identifying the subtle and varied patterns characteristic of LFAs. This IDS leverages DQL to adjust network routing dynamically, ensuring the system maintains optimal performance even under attack conditions. By integrating DRL into the IDS, ReLFA detects and mitigates attacks in real-time, enhancing IoT networks' overall security and resilience. The effectiveness of ReLFA as an IDS is demonstrated through simulation results, showing that ReLFA achieves faster rerouting and mitigation compared to existing methods, such as LFADefender and Woodpecker. Specifically, ReLFA's rerouting process is much quicker, reducing the time required to alleviate the impact of LFAs and restoring normal network operations more efficiently.
Nisha Kandhoul et al. [106] proposed the Deep Q-Learning Security (DQNSec), a novel routing protocol for the Opportunistic Internet of Things (OppIoT) that leverages DQL. This protocol conceptualizes OppIoT as a Markov decision process and employs value-based and policy-based DQL methods to predict and mitigate threats dynamically. Their extensive simulations, using real data traces from Kaggle's Microsoft Malicious dataset, underscore the robustness of DQNSec against traditional ML-based routing protocols. Specifically, the DQNSec protocol showcased a delivery probability of 47.9%, which was significantly superior to other ML-based solutions such as RFCSec (35.4%), RLProph (33.4%), CAML (32%), and MLProph (29.4%). Additionally, the protocol demonstrated lower average latency and packet drop rates, with an average latency of 1874.58 seconds and 2679 packets dropped, establishing its efficiency, accuracy, and enhanced responsiveness to network anomalies.
Jiadai Wang et al. [107] explored the enhancement of security in the SDN-enabled IIoT through DRL. Their study highlighted that Forwarding Nodes (FNs) are vulnerable to a spectrum of cyber-attacks. They proposed a novel attack tolerance scheme that employs DQL to adaptively direct traffic away from compromised FNs, thereby minimizing the impact of these attacks. The scheme's effectiveness is bolstered by using a Generative Adversarial Network (GAN) to generate realistic network traffic data, enhancing both the training and the evaluation of the DRL model. This methodology not only improved the success rate of IIoT traffic reaching its destination and achieving near-optimal performance but also strengthened the overall security framework of IIoT against diverse attack vectors.

Kezhou Ren et al. [108] introduced MAFSIDS, a sophisticated IDS utilizing DQL to enhance the effectiveness of feature selection in network security systems. By incorporating a Graph Convolutional Network (GCN), MAFSIDS selectively processes network features, achieving high accuracies of 96.8% and 99.1% on CSE-CIC-IDS2018 and NSL-KDD datasets, respectively. This approach significantly minimizes feature redundancy, reducing the original feature set by approximately 80%. Integrating DQL with GCN not only refines the feature selection process but also optimizes the overall performance of IDS, streamlining the identification and classification of network threats.
Bin Yang et al. [109] investigated the application of DRL, specifically DQL and Policy Gradient methods, in their advanced network IDS operating at both packet and flow levels. These DRL algorithms enhance the system's ability to detect and manage cyber threats efficiently across these dimensions: packet-level, which involves the analysis of individual data packets, and flow-level, where the traffic flow in IoTthe network is examined. Utilizing the CICDDoS2019 dataset, the research demonstrated significant improvements in detection performance, achieving an accuracy rate of up to 98.78%, with a marked increase in detection speed and adaptability to new, unseen cyber threats.

3.2.4 Soft Actor-Critic (SAC)

Yuming Feng et al. [110] explored deploying the SAC model in IoTDRL to enhance DDoS attack detection in IoT systems. Their approach innovatively adapts the parameters of an unsupervised classifier at IoT edge gateways, which are especially vulnerable to abrupt and extensive variations in network conditions. In this scenario, traditional adaptive methods falter. The effectiveness of the SAC model in this context is underscored by its ability to respond to environmental changes dynamically across various IoT devices. This method expedites detection and optimizes resource utilization by fostering a collaborative network sharing data and detection insights among edge devices. This is crucial for strengthening the overall security infrastructure of IoT networks against sophisticated and large-scale DDoS attacks. Significantly, the implementation of this approach has yielded a detection accuracy exceeding 95% in real-world scenarios, demonstrating its significant potential to detect even stealthy DDoS attacks effectively.

3.2.5 Q-learning

H. Karthikeyan et al. [111] proposed advanced security frameworks of Intelligent Transportation Systems (ITS) by integrating Q-learning with DRL methodologies to effectively manage and mitigate DDoS flooding attacks in IoTdynamic vehicular networks. This integration especially targets the complex network environments of Vehicle-to-Infrastructure (V2I) and Vehicle-to-Vehicle (V2V) communications in IoTITS. The adaptability of Q-learning enables it to effectively respond to low-rate and high-rate DDoS attacks, thus significantly enhancing the resilience of Roadside Units (RSUs) and overall network traffic management. This approach proactively adjusts defense strategies in real-time, aligning them with evolving attack patterns to safeguard critical transportation infrastructure. The dynamic adjustment is facilitated by continuously updating the reward mechanisms and policy networks based on real-time traffic and threat analyses, allowing immediate and effective responses to changing threat dynamics.

Frantzy Mesadieu et al. [112] innovatively applied a DRL framework that integrates a DQN for enhanced IDS in IoTSCADA systems. This pioneering approach exploits the synergies between Q-learning and Deep Learning (DL) through what is designated as the 'Q-network.' The 'Q-network' is a specialized neural architecture that augments Q-learning with the capacity to analyze and learn from complex data patterns in network traffic, patterns that are often overlooked by conventional detection methods. Conventional IDS, primarily signature-based and anomaly-based systems, frequently struggle against novel and sophisticated cyber threats due to their inherently static nature. While these traditional methods are effective against known threats, they lack the flexibility to adapt to new, evolving challenges. In stark contrast, the Q-network dynamically processes and updates from ongoing data streams, markedly enhancing its ability to identify anomalies that indicate potential security breaches in real-time. The deployment of the Q-network represents a substantial shift away from traditional methodologies by equipping the system with a continuous learning mechanism. Moreover, this mechanism utilizes historical data and continuously adapts to new threat patterns. This crucial adaptive capability significantly contributes to the system's outstanding detection accuracy of 99.36%.

Hafiz Husnain Raza Sherazi et al. [113] developed a sophisticated hybrid detection system to combat DDoS attacks in IoV networks, integrating fuzzy logic and Q-learning to enhance security measures. Fuzzy logic effectively processes ambiguous and uncertain data, allowing dynamic responses to diverse attack patterns. Q-learning adapts defensive strategies based on real-time attack data, improving the system's reactivity and adaptiveness. The simulations demonstrate significant improvements in buffer size efficiency, energy consumption, response time, and network throughput, underscoring the system's capability to maintain operational performance even under attack conditions.

3.2.6 Multi-Armed Bandit (MAB)

MAGPIE represents a significant advancement in IDS for smart homes, developed by Ryan Heartfield et al. [114]. This system dynamically adjusts its anomaly classification decisions using a non-stationary-MAB approach and updates its probabilistic cluster-based reward functions based on silhouette scores. These adjustments are crucial as they account for the non-stationary behaviors typical of smart home environments and the evolving interactions of users with their devices. In this context, "valid" refers to the system's ability to operate effectively under realistic conditions. MAGPIE demonstrated high efficiency and accuracy, especially in recognizing novel cyber-physical threats and adapting to new user behavior patterns. Experimental evaluations show that MAGPIE achieves an anomaly detection accuracy rate of up to 93%, underscoring its potential to enhance security in smart homes significantly.
Radoglou-Grammatikis et al. [115] presented a cutting-edge Intrusion Detection and Prevention System (IDPS) designed specifically for industrial healthcare systems. By integrating SDN with RL, this study addressed the critical vulnerabilities inherent in the IEC 60 870-5-104 protocol, a widely used standard in such systems. The DRL component is especially noteworthy, as it models the mitigation strategy as a stationary multi-armed bandit problem, which is optimally solved using Thompson sampling. The detection accuracy and F1 score achieved by the IDPS are 0.831 and 0.8258, respectively, indicating high precision in identifying threats. Additionally, the mitigation accuracy reaches 0.923, showcasing the system's robustness in neutralizing potential cyberattacks.

3.2.7 Deep Deterministic Policy Gradient (DDPG)

Chengming Hu et al. [116] proposed a novel IDS for the smart grid, named RL-Based Adaptive Feature Boosting (AFB). This system leverages multiple AutoEncoders to extract critical features from the multi-sourced, heterogeneous data generated in smart grid environments. These features are then utilized to train a Random Forest (RF) classifier. The RL-AFB system employs the DDPG algorithm to dynamically adjust the feature sampling probabilities based on their contribution to classification accuracy. The application of DDPG significantly improved classification performance, achieving a 97.28% accuracy on the Hardware-In-the-Loop (HIL) security dataset and outperforming other methods on the WUSTIL-IIOT-2021 dataset.
Laisen Nie et al. [117] proposed a novel IDS explicitly designed for green IoT systems, utilizing DDPG-based DRL. This IDS addresses the significant challenges associated with traditional methods, such as slow detection speeds and high false alarm rates. By implementing DDPG, the system efficiently predicts and schedules network traffic flows, enabling the rapid and

accurate detection of DDoS attacks. In the evaluation, the DDPG-based IDS substantially improved detection accuracy and speed. The system achieved a True Positive Rate (TPR) of 99.98% and 100% in two different datasets, significantly outperforming conventional methods such as SRMF and MWM, which showed TPRs of 92.48% and 82.76%, respectively. Additionally, the system maintained a low False Positive Rate (FPR), indicating its ability to minimize false alarms while adapting to dynamic network conditions. The dynamic threshold adjustment capability of DDPG was especially effective, reducing the Time Relative Error (TRE) to 0.2752 and 0.3721 in two datasets, compared to 0.9838 and 0.9964 for traditional methods. This result highlighted the system's ability to adapt in real-time to fluctuating network conditions, thus significantly enhancing the real-time security of IoT networks.
Noora Mohammed et al. [118] proposed a novel approach to enhance the security of FL in IoT systems by utilizing a DDPG-based reputation management mechanism. This method addresses the dynamic nature of worker behavior in FL by continuously optimizing reputation thresholds, which is critical in identifying and mitigating the risks posed by unreliable or malicious participants. Unlike traditional approaches such as the static reputation threshold models, which set a fixed threshold that may not adapt well to varying worker behaviors, or FedAvg, which lacks any built-in mechanism for evaluating the trustworthiness of workers, the DDPG-based approach allows for more nuanced and effective decision-making. Additionally, the study compares the DDPG method with DQN-based reputation models, highlighting that while DQN can handle discrete action spaces, it struggles with the continuous and high-dimensional action spaces often encountered in FL settings. In contrast, DDPG excels in such scenarios, leading to a more robust selection of reliable workers and ultimately improving the overall accuracy and stability of the FL model. The results presented in the study show that the DDPG-based system improves model accuracy by over 30% compared to conventional methods, including FedAvg, DQN-based reputation models, and static reputation threshold methods.

3.2.8 Inverse Reinforcement Learning (IRL)

Juan Parras et al. [119] proposed using Inverse Reinforcement Learning (IRL) to enhance defense mechanisms against intelligent backoff attacks in wireless networks. They introduced two IRL-based strategies that operate under partial observability, detecting attacks by analyzing deviations from normal behavior. Unlike traditional methods, these mechanisms generalize from known attacks to predict and counteract previously unseen strategies, significantly improving network resilience. While IRL is distinct from DRL, which focuses on inferring reward functions rather than optimizing policies, it can be augmented with DL for handling complex scenarios. This study's contribution is crucial as it provides a flexible, adaptive defense framework that requires minimal assumptions about attack types, addressing the challenge of increasingly sophisticated cyber threats.

3.2.9 Proximal Policy Optimization (PPO2)

Sumegh Tharewal et al. [120] proposed IDS is sourced from the U.S. Department of Energy's Oak Ridge National Laboratory. This dataset is designed explicitly for IIoT systems, encompassing various types of network traffic data representative of real-world IIoT scenarios. The dataset includes standard traffic data and a wide range of malicious attack data, ensuring a robust testing ground for the IDS. Specifically, the dataset comprises 26 distinct features, each representing different aspects of network traffic, such as packet size, protocol types, and time-based features. These features are crucial for identifying patterns and anomalies in IoT networks, which indicate normal operations or potential security breaches. The attack types covered in the dataset include DoS, reconnaissance attacks, malicious command injections, and other forms of cyber threats that are common in IIoT systems. In the study, LightGBM filters out the least important features, reducing the feature set to those most relevant for IDS. This process ensures that the IDS operates efficiently without compromising on detection accuracy. The dataset's diversity and its alignment with real-world IIoT conditions make it an ideal choice for training and validating the IDS, allowing for the demonstration of the system's ability to detect and respond to various types of network threats with high precision and speed. The rigorous testing on this dataset highlights the IDS's superior performance, achieving a detection accuracy of 99.09% and demonstrating its practical applicability in securing IIoT systems.

3.2.10 One-Shot Learning Techniques

Nouf Saeed Alotaibi et al. [121] introduced a dynamic IDS tailored for smart city systems, utilizing RL techniques, specifically one-shot learning, enhanced with DRL capabilities. This approach is specially crafted to tackle the inherent dynamic challenges associated with multi-access edge computing architectures, e.g., managing the variability in network behavior and the potential for increased security vulnerabilities. The model efficiently mitigates zero-day attacks and other emergent threats in IoT-based networks. One-shot learning enables the system to adapt to novel scenarios rapidly using minimal training data. In contrast, the DRL component effectively uses sparse experiential data to make informed real-time decisions. This dual functionality significantly boosts the system's resilience, maintaining robust security against the evolving landscape of network threats. Empirical results indicate a detection accuracy rate of approximately 98.8% with low false positive rates, underscoring the model's effectiveness in real-world applications.

3.2.11 Double Deep Q-Network (DDQN)

Shi Dong et al. [122] combine the strengths of supervised and unsupervised learning to detect anomalous traffic in network data effectively. A key element of this approach is the DDQN, a variant of DRL that enhances the precision and robustness of network anomaly detection. In this method, an autoencoder is first used to reconstruct network traffic features, fed into a deep neural network integrated into the DDQN framework for classification in IoT. This layered approach allows the model to dynamically adjust to varying traffic patterns and accurately identify known and unknown network anomalies. The implementation of K-Means clustering further aids in detecting unknown attacks by categorizing traffic features into distinct clusters without prior labeling, thus reducing the reliance on manually labeled data. The improvements brought about by this semi-supervised method are significant, especially when compared to traditional ML models (e.g., DT, SVM, and RF). The SSDDQN method achieved an accuracy of 83% in binary classification and improved the F1-Score by approximately 6% over the traditional DDQN model. In five-class classification tasks, the SSDDQN model reached an F1-Score of 79.43% and demonstrated a 5% improvement in detection rate (DR) compared to DDQN. These numerical results underscore the model's capability to reduce time complexity while enhancing critical performance metrics such as accuracy, F1-Score, and detection rate, making it especially effective in real-time network environments.
Mohammad Alauthman et al. [123] presented a sophisticated approach to detecting peer-to-peer (P2P) botnet traffic, which is difficult to identify due to its decentralized and evasive characteristics. The study developed a system that integrates a refined traffic reduction mechanism with RL techniques, specifically targeting the improvement of botnet detection accuracy. The proposed approach was thoroughly tested on real-world network traffic, yielding a high detection accuracy of 98.3% and maintaining a low false positive rate of 0.012%. This combination of network traffic analysis with an adaptive RL framework marks a significant advancement in the domain, as it effectively identifies bot-related activities with minimal disruption to network performance. The RL-based method employed is carefully tailored to the unique challenges posed by P2P botnets, offering a robust and highly adaptive solution.
Praveena et al. [124] introduced an optimal DRL approach specifically designed for an IDS in Unmanned Aerial Vehicles (UAVs). The approach leverages DDQN combined with Black Widow Optimization (BWO) to enhance the IDS's performance. The model fine-tunes the network parameters by utilizing BWO for hyperparameter tuning, significantly improving the IDS's capability to predict and prevent threats in UAV networks accurately. This advanced method strengthens the learning process through DDQN. It optimizes the detection process

to handle the complexities of high-dimensional data and intricate attack vectors commonly encountered in UAV networks. The efficacy of this IDS was tested using the NSL-KDD dataset, where it achieved remarkable performance metrics: a precision of 0.985, recall of 0.993, F-measure of 0.988, and accuracy of 0.989. These results affirm the model's effectiveness in real-world scenarios, demonstrating its potential as a robust IDS solution for UAV networks. Integrating DDQN with BWO deepens the IDS's learning capability and optimizes its detection process, making it highly effective in addressing the unique security challenges posed by UAV networks.

3.2.12 Policy Gradient-DRL (PG-DRL)

Vikas Juneja et al. [125] tackled the problem of energy-draining vampire attacks in WSNs, which disrupt network efficiency by extending data transmission routes, leading to rapid energy depletion in sensor nodes. The authors introduce a dual-strategy approach combining a cooperative trust calculation mechanism with a PG-DRL algorithm, demonstrating quantifiable improvements in network performance. The cooperative trust mechanism plays a pivotal role in detecting malicious nodes by analyzing deviations in energy consumption patterns. The cooperative trust matrix, generated by leveraging the residual energy of connected nodes, enables the detection of vampire nodes with an accuracy rate that consistently reaches 100% in controlled scenarios. This method is especially effective in networks with varying environmental conditions, where traditional threshold-based methods might fail. Simultaneously, the PG-DRL algorithm optimizes routing paths by dynamically selecting the next-hop node, ensuring minimal energy consumption and maintaining high levels of trust in the IoT network. The PG-DRL approach significantly reduces unnecessary energy expenditure, with simulations indicating a 3% increase in network lifetime compared to the benchmark Dynamic Source Routing (DSR) protocol. Moreover, the methodology also improves the detection ratio by 20% compared to existing state-of-the-art schemes.

3.2.13 State-Action-Reward-State-Action (SARSA)

Abbasgholi Pashaei et al. [126] developed cutting-edge IDS for industrial control networks, focusing on detecting and preventing critical threats such as Man-in-the-Middle (MITM) and DDoS attacks. The system utilizes the SARSA algorithm, a model-free, on-policy RL method. Unlike DRL, SARSA updates its policy based on current action dynamic adaptability in real-time network environments. The IDS is structured with a dual-agent model: one agent simulates the network environment to challenge the classification agent, which enhances its ability to detect complex attacks. This adversarial setup, combined with a honeypot that mimics realistic network behaviors, allows the system to lure and learn from potential attackers effectively. The system demonstrated over 99% accuracy and an F-measure of 0.98 in extensive tests, outperforming traditional IDS models.

3.2.14 Grasshopper Optimization Algorithm (GOA)

Alok Kumar Shukla et al. [127] presented an innovative approach to enhancing IDS through advanced optimization and RL techniques. The study introduced a novel variant of the Grasshopper Optimization Algorithm (GOA), termed the opposition self-adaptive GOA, which incorporates mutation and perceptive strategies to improve the selection of relevant features for anomaly detection. To further bolster the system's detection capabilities, the study integrated the RL method, specifically the Gain Actor-Critic (GAC) approach, into the SVM. This hybrid model was evaluated using standard IDS datasets, including NSL-KDD, AWID, and CICIDS 2017. The results showcased the model's superior performance, achieving a detection accuracy of 99.71% on NSL-KDD, 99.11% on AWID, and 99.61% on CIC-IDS 2017, along with impressively low false-positive rates of 0.009, 0.091, and 0.052, respectively.

3.2.15 Multiagent Deep Deterministic Policy Gradient (MADDPG)

Delali Kwasi Dake et al. [128] introduced a DRL framework for SDN and IoT, focusing on routing optimization and DDoS detection/prevention using the MADDPG algorithm, an extension of DDPG for multi-agent environments. This framework allows multiple RL agents to work cooperatively or competitively to enhance network performance metrics, e.g., delay, jitter, and packet loss rate. These improvements contribute to better IDS capabilities by ensuring stable network conditions, which reduce false positives and negatives, thus allowing more accurate detection of genuine threats. Extensive simulations show that the MADDPG-based framework outperforms traditional DDPG, especially in complex, dynamic network scenarios, highlighting its effectiveness in improving security and performance in SDN-IoT systems.

3.2.16 Dueling Double Deep Q-Network (D3QN)

Xuecai Feng et al. [129] developed the Security Defense Strategy Algorithm (SDSA), a novel approach to enhancing IoT security using DRL. The SDSA leverages the D3QN, which integrates the strengths of Dueling and Double Q-learning methodologies, to optimize the allocation and coordination of defense resources. By simulating adversarial security scenarios involving multiple defenders and attackers, the SDSA enables the adaptive learning of strategic behaviors that enhance defense effectiveness. The efficacy of the SDSA is demonstrated through empirical data, where the method showed an improvement in defense effectiveness by approximately 87% and 85% compared to the MADDPG and OptGradFP models, respectively, in scenarios involving a single attacker. In more complex scenarios with multiple attackers, the SDSA outperformed the MADDPG and OptGradFP models by 65% and 60%, respectively.

3.2.17 Neural Fitted Q-Iteration (NFQ)

Sengathir Janakiraman et al. [130] presented a sophisticated approach to combating DDoS attacks in cloud environments supported by fog computing. The study introduced a novel DDoS mitigation scheme that leverages the integration of DRL and LSTM networks to enhance the detection and mitigation processes. The scheme employs the NFQ algorithm with LSTM to analyze and respond to malicious traffic dynamically. This combination effectively manages high-dimensional state spaces and temporal dependencies, improving the system's ability to classify and filter out infected packets while ensuring service availability. The framework also incorporates SDN technology in IoT, the fog layer, enabling a robust defense mechanism against DDoS attacks by efficiently managing and controlling network traffic across cloud and fog environments. The experimental results demonstrate the proposed scheme's effectiveness, with the RDL-LSTM-2.2 variant achieving a classification accuracy of 98.34%, precision of 98.88%, and recall of 98.76%. Additionally, the proposed scheme reduced packet latency by 10.42% and computational complexity by 12.96% compared to existing approaches. The scheme also achieved a 99.59% training accuracy and a 98.98% testing accuracy, demonstrating its superior performance in mitigating DDoS attacks in a fog-assisted cloud environment.

3.2.18 Distribution of Models in Literature

Figure 3.1 provides a comprehensive view of the distribution of various DRL-based models utilized in IDS in IoT systems. This distribution highlights the prevalence and application-specific suitability of different models, reflecting both their strengths and the unique challenges they address in IoT contexts.

The DQN dominates with a 25.7% share, underscoring its significant role in IoT security. DQN's prominence is attributable to its robustness in handling large state and action spaces, which is typical in IoT systems characterized by numerous interconnected devices and complex interactions. The DQN model leverages experience replay and off-policy learning, allowing it to store past experiences and learn from them effectively. This capability is crucial

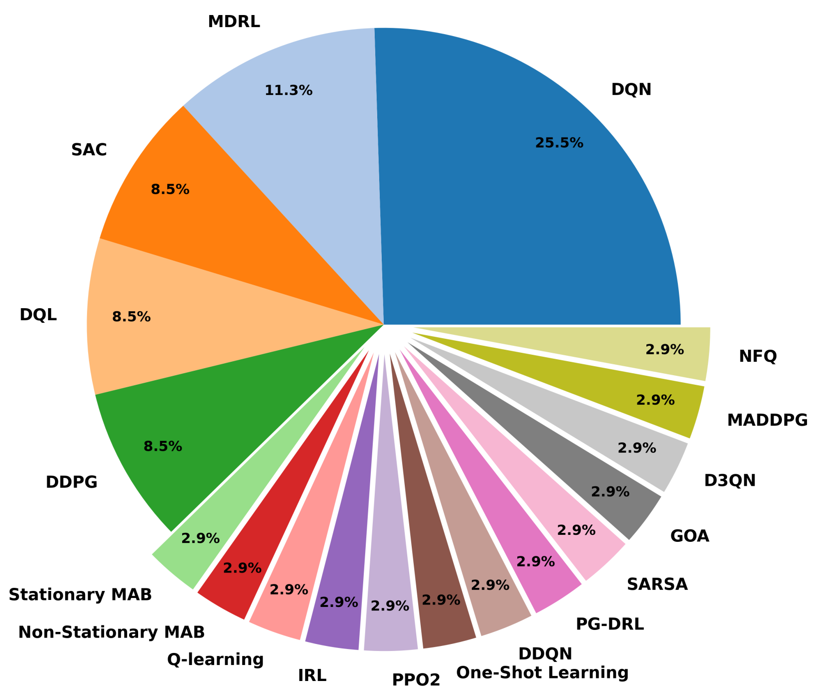

Figure 3.1 Usage Frequency of Different Models in DRL-based IDS in IoT.

in IoT scenarios requiring real-time decision-making and adaptability to dynamic conditions. The model's ability to optimize policies by evaluating the Q-values of actions in given states makes it especially effective in environments where exploration and exploitation must be balanced.

The MDRL model accounts for 11.4% of the usage, highlighting its adaptability in tailored IoT applications. MDRL models often involve modifications to standard DRL architectures to handle better IoT systems' specific constraints and requirements, such as limited computational resources, energy efficiency, and rapid response to threats. The modifications typically aim to enhance the model's ability to operate in highly variable and unpredictable environments, which is common in IoT networks.

DQL and SAC each represent 8.6% of the applications, indicating their critical roles in specific IoT scenarios.

DQL is an extension of Q-learning that integrates DL to manage continuous action spaces more effectively. This makes DQL highly suitable for IoT systems where decisions are not binary but fall on a spectrum of possible actions, such as adjusting the intensity of an IoT device's response to an external threat. DQL's application in high-dimensional sensory input processing, especially in sensor networks and robotics in IoT, ensures that systems can

respond more precisely and effectively to nuanced threats.
SAC is especially favored for its capacity to maintain stability and reliability in learning processes even under dynamically changing policy conditions. This attribute is essential in IoT applications that require long-term strategic planning (e.g., smart grids or autonomous systems), where the environment and network conditions can vary significantly over time. SAC's use of stochastic policies, combined with entropy regularization, enables it to balance exploration and exploitation more effectively, making it robust against overfitting specific scenarios and ensuring broader applicability across varying IoT contexts.
Other models, each contributing 2.9% to the distribution, are chosen for specialized applications, reflecting their tailored utility in specific IoT systems:
PG-DRL and DDPG are especially effective in configurations with continuous action spaces. They are employed in scenarios requiring high-dimensional decision-making processes, where traditional discrete action models like DQN might struggle. PG-DRL, for instance, directly optimizes the policy by following the gradient of expected rewards, making it suitable for complex tasks where the policy space is large and continuous. DDPG extends this by incorporating deterministic policies, which improve efficiency in environments where exploration noise needs to be minimized, such as in highly regulated or safety-critical IoT systems.
Q-learning and SARSA are valued for their simplicity and convergence properties, making them reliable for applications where the action space is discrete and well-defined. Q-learning is especially robust in environments where the agent needs to learn from an off-policy perspective, adapting to environmental changes even if they are not directly related to the actions taken. SARSA, being an on-policy algorithm, is more suitable in environments where the policy being evaluated is the one being improved, providing a more stable learning process in such scenarios.
IRL is employed when the objective is to infer the underlying reward structure from observed behavior rather than optimizing a predefined reward. This is especially useful in surveillance and anomaly detection systems in IoT, where the goal is to understand and replicate the decision-making processes of expert systems or humans. IRL's ability to generalize from observed data makes it a powerful tool for environments where the reward structure is not explicitly known or is difficult to define.
One-shot learning techniques are critical when data is scarce, or the system must rapidly adapt to new and unseen scenarios. In IoT systems, where gathering large amounts of labeled data can be challenging due to devices' distributed and heterogeneous nature, one-shot learning allows systems to generalize from minimal examples, ensuring quicker adaptation and response times.
Stationary and MAB models are utilized in scenarios where the reward distributions change

over time, which is common in IoT systems subject to fluctuating network conditions and varying threat levels. These models are instrumental in dynamic resource allocation problems, where the system must continuously adapt its strategy to maximize the long-term rewards.

DDQN and D3QN improve upon standard DQN by addressing the overestimation bias inherent in Q-learning, which can lead to suboptimal policies. DDQN uses separate networks to select and evaluate actions, reducing this bias. At the same time, D3QN further refines this by decoupling the value and advantage functions, allowing the model to better distinguish between valuable actions in similar states. These improvements make them highly effective in complex IoT systems where fine-tuned action selection is critical.

MADDPG and NFQ are applied in more complex IoT systems that involve multiple agents or require a nuanced understanding of time-sequential data. MADDPG is especially useful in environments where multiple IoT devices must collaborate or compete, such as in networked control systems or cooperative sensor networks. At the same time, NFQ excels in scenarios where the system must learn from sequences of decisions, making it ideal for temporal analysis in IoT security.

The diversity of models employed in DRL-based IDS for IoT reflects the multifaceted challenges these systems face, from handling continuous and high-dimensional action spaces to ensuring real-time responsiveness and adapting to evolving threats. The unique demands of the application often dictate the selection of a specific model, the nature of the IoT environment, and the specific objectives of the IDS, whether that is optimizing resource usage, minimizing response times, or improving detection accuracy.

3.2.19 The existing DRL-based IDS

The application of DRL-based IDS spans various IoT contexts, each addressing distinct security challenges, as summarized in Table 3.1. In smart homes, DRL's adaptive threat detection and real-time response capabilities are vital for securing complex networks of diverse devices. In IIoT, DRL-based IDS protects critical infrastructure by optimizing decision-making and mitigating sophisticated threats that could disrupt operations. In healthcare IoT, DRL ensures accuracy in threat detection and protecting sensitive patient data, a vital necessity given the high stakes involved. Transportation systems, especially vehicular networks, benefit from DRL's ability to secure dynamic environments against advanced cyberattacks. Smart cities, with their intricate IoT ecosystems, leverage DRL to safeguard essential services and maintain operational efficiency. The versatility and adaptive learning capabilities of DRL make it especially beneficial in IoT applications, where the ability to continuously learn from evolving threats and improve detection accuracy is critical. This positions DRL as a cor-

Table 3.1 DRL-based IDS applications in various IoT contexts.

IoT Sector	Paper Reference	Brief Analysis
Smart Home Security	[96], [104], [105], [106], [97]	Focuses on enhancing security measures in smart homes through adaptive threat detection and real-time response to diverse threats like DDoS and cyber-attacks.
Industrial IoT (IIoT)	[110], [107], [98], [111], [130], [112], [131]	Discusses complex decision-making and operational optimization in industrial systems, addressing high-stakes security challenges.
Healthcare IoT Systems	[115], [116], [119], [120]	Applies DRL to ensure critical accuracy and privacy, adapting to the sensitive nature of healthcare data and operations.
Transportation Systems	[111], [123], [124], [103], [101], [118]	Enhances security in transportation by managing dynamic environments and securing high-speed vehicular networks against sophisticated attacks.
Smart Cities	[99], [121], [109], [117], [122], [100], [129], [128]	Utilizes DRL to manage complex urban IoT systems, optimizing security across dense city networks and enhancing city-wide communication protocols.
General IoT Security	[108], [125], [126], [127], [102]	Explores the broader application of DRL across various IoT domains, emphasizing adaptive learning and security enhancements in diverse operational contexts.

nerstone technology for enhancing security in the increasingly complex landscape of IoT, ensuring robust protection across diverse sectors.

3.3 The datasets are used for DRL-based IDS

This section thoroughly examines the datasets used in DRL-based IDS for IoT, detailed in Table 3.2. We classify these datasets to show their specific role and impact on IoT security, pointing out how each contributes to the enhancement of the IDS capability in IoT.

3.3.1 Real-World Network Traffic Simulation

Datasets such as ISCXIDS2012 [132], UNSW-NB 15 [133], and CICIDS2017 [134] are crucial for simulating realistic network environments for IDS training and evaluation. These datasets include labeled data representing various attack scenarios commonly observed in real-world networks, such as DDoS, brute force attacks, and infiltration attempts.

3.3.2 Specialized IoT systems Attacks

Specialized datasets such as WUSTL-IIoT-2018 [135], WUSTL-IIoT-2021 [136], and N-BaIoT [137] focus specifically on industrial and home IoT systems, offering scenarios including advanced persistent threats (APTs) and zero-day exploits. For example, the WUSTL-IIoT datasets capture traffic data from industrial IoT devices under various attack scenarios,

Table 3.2 Overview of types of attacks covered by each dataset

Dataset Name	Types of Attack Covered
ISCX 2012 IDS	Brute Force SSH, Brute Force FTP, Infiltration, HTTP DoS, DDoS, Botnet
Microsoft Malware	Malware (Viruses, Worms, Trojans, Spyware, Adware, etc.)
N-BaIoT	DDoS, DoS, Reconnaissance, MITM
UNSW-NB-15	Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worms
CICIDS2017	DoS, DDoS, Heartbleed, Bot, Infiltration, Web Attacks, PortScan, FTP-Patator, SSH-Patator, etc.
IoTID20	Backdoor, Analysis, Fuzzing, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worms
NSL-KDD	DoS, Probe, U2R, R2L, etc.
AWID	Flooding, Injection, Impersonation, Miscellaneous
CSE-CIC-IDS2018	Brute Force, DoS, DDoS, SQL Injection, Heartbleed, etc.
WUSTL-IIoT-2021	APTs, MITM, Side-channel Attacks, Zero-day Exploits, etc.
WUSTL-IIoT-2018	Advanced Persistent Threats, Evasion Techniques, Side-channel Attacks
MedBIoT	Early-stage malware activities (e.g., scanning, reconnaissance)
TON-IoT Dataset	Botnet, Command Injection, Malware Propagation
DS20S Trace	Robust Feature Extraction in Noisy and Heterogeneous IoT Systems
Hogzilla	CTU-13 Botnet, Illegitimate Packets
SCADA-based Power System	The famous Stuxnet worm that damaged nuclear machinery in Iran.
CIC-DDOS2019	DDoS Attacks (e.g., UDP Flood, HTTP Flood, SYN Flood)
BoT-IoT	DDoS, DoS, OS and Service Scan, Keylogging, Data exfiltration attacks
ISOT-CID	Various IoT-specific attacks and vulnerabilities
BoTNeTIoT-L01	Automated IoT botnet attack scenarios, traffic anomalies
IEC 60 870-5-104	Protocol-specific attacks targeting energy sector communications

including APTs, which are long-term targeted attacks aiming to steal data or disrupt operations. These datasets are critical for developing IDS that target the unique vulnerabilities of IoT devices and networks, such as botnet attacks (e.g., Mirai botnet variants), which exploit common IoT weaknesses like default credentials and outdated firmware.

3.3.3 Emerging and Evolving Threats

The Hogzilla [138] and IoTID20 [139] datasets are instrumental in reflecting the evolving nature of cyber threats, especially those involving sophisticated malware and botnet configurations targeting IoT devices. For example, Hogzilla includes scenarios simulating advanced malware behaviors, such as polymorphic and metamorphic techniques, which allow malware to change its code to evade detection. IoTID20 captures the traffic patterns associated with emerging threats like cryptojacking and data exfiltration through botnets.

3.3.4 Protocol-Specific Security

The IEC 60870-5-104 [140] dataset tests communication protocols used in critical sectors such as the energy industry, especially SCADA systems. This protocol is widely used for data exchange between control centers and substations. Vulnerabilities in this protocol can allow attackers to manipulate control signals, leading to catastrophic outcomes such as power outages.

3.3.5 Wireless Network Security

The AWID dataset [141] focuses on attacks in wireless IoT networks (e.g., flooding, injection, and impersonation). This dataset is especially relevant for training IDS to protect wireless communications in IoT systems, where security risks are heightened due to the inherent vulnerabilities of wireless protocols like Wi-Fi. For instance, the dataset includes scenarios of Wi-Fi flooding attacks, where an attacker overwhelms a network with traffic to cause a denial of service.

3.3.6 Botnet and DDoS Focus

BoT-IoT [142] and CIC-DDoS2019 [143] datasets offer extensive volume data on botnet and DDoS attacks, predominantly in the disruption of IoT operations. Therefore, these datasets facilitate the establishment of IDS, which can identify and counter the effects of the interruption in case of large-scale coordinated attacks started from multiple compromised devices.

3.3.7 Sector-Specific and Critical Infrastructure

Sector-specific datasets such as TON-IoT [143], MedBIoT [144], and SCADA-based power systems [145] address the unique security challenges in specialized IoT systems. For example, TON-IoT focuses on teleoperated networks and includes data related to malware propagation and command injection attacks, which are common in industrial IoT. MedBIoT is especially important for securing medical IoT devices, where the integrity and confidentiality of patient data are paramount. Also, the SCADA-based power system dataset simulates attacks on critical infrastructure, including scenarios similar to the Stuxnet attack.

3.3.8 Dataset Utilization for Robust DRL-based IDS

IDS have been tested using various datasets (e.g., ISCXIDS2012 [132], UNSW-NB15 [133], and CICIDS2017 [134]). These datasets emulate real-world attack scenarios, including DDoS, brute force, and infiltration attacks, enabling IDS to demonstrate robust performance across diverse operational conditions.

Specialized datasets, such as WUSTL-IIoT, N-BaIoT, and Hogzilla, are especially valuable for enhancing IDS accuracy. They allow IDS to focus on specific threats(e.g., vulnerabilities in industrial IoT or modern malware, enabling finely tuned defenses tailored to these contexts). For instance, using the WUSTL-IIoT dataset, IDS can be trained to detect advanced persistent threats in industrial settings. In contrast, the N-BaIoT dataset helps identify and mitigate botnet activities specific to IoT devices. The proactive integration of new datasets, such as IoTID20, which captures emerging threats such as cryptojacking, and protocol-specific datasets such as IEC 60870-5-104, further strengthens IDS capabilities. By incorporating these datasets, IDS can adapt to evolving threat landscapes and better protect critical infrastructure from specialized attacks. Using sector-specific datasets, such as those for SCADA systems, medical IoT devices, and teleoperated networks, is crucial for refining IDS solutions. These datasets ensure that IDS are generally effective and capable of addressing the unique challenges of specific environments.

3.4 Discussion

By integrating DRL into IDS, these systems gain the capability to dynamically adapt and respond to new and evolving threats, a critical advancement given the complexity and continuous expansion of IoT networks. We present a detailed analysis of states, actions, and rewards across selected works in our SLR to demonstrate the transformative potential of DRL in enhancing IDS for the IoT. An overview of the specific functionalities, state descriptions, actions implemented, and reward mechanisms developed in these studies is presented in Table 3.3.

DRL empowers IDS with sophisticated decision-making and adaptability, essential for navigating the intricacies of IoT systems. This adaptability is showcased through diverse state definitions, ranging from simple network traffic patterns to complex, multi-dimensional data vectors, as highlighted in studies, e.g., [96] and [104]. Such dynamic adaptability ensures robust protection against a wide array of cyber threats, thus maintaining the integrity and functionality of critical IoT infrastructure.

DRL's flexibility in action selection allows for a broad spectrum of responses tailored to specific network conditions and threats. This is evident from strategies ranging from basic traffic management (e.g., blocking or allowing traffic) to more complex ones such as adjusting classifier parameters [97] and managing traffic [105]. This strategic application of varied actions helps create a more proactive and intelligent security posture, essential for promptly pre-empting breaches and mitigating threats.

The deployment of DRL-based IDS also brings forth challenges, primarily in designing effective reward mechanisms and managing the computational demands of DRL models. Effective reward mechanisms are crucial as they must encourage desirable behaviors that lead to successful threat mitigation, as shown across various studies [102], [112]. However, these mechanisms must also reflect the complex realities of IoT security, where diverse factors such as device heterogeneity, network dynamics, and threat diversity need to be considered without overwhelming IoT system resources [115]. Moreover, the computational intensity of DRL models poses significant challenges for deployment in resource-constrained IoT systems [116], necessitating further research for optimizing these models for practical applications.

In addition, deploying DRL-based IDS in real-world IoT networks presents additional challenges, including computational constraints, energy efficiency, scalability, and sustainability. IoT devices often have limited processing capabilities, making it challenging to implement resource-intensive DRL models [146]. These devices typically operate on constrained power sources, and continuous IDS operations can lead to rapid energy depletion [147]. The scalability of DRL models is also a concern, as IoT networks can consist of thousands of interconnected devices, complicating the deployment of IDS across large-scale environments [147], [37]. Sustainability is another critical factor, as the energy consumption associated with training and deploying DRL models must align with green computing initiatives. Several strategies have been proposed to address these challenges. Leveraging edge computing and FL can help distribute computational workloads, reducing the burden on individual IoT devices. Implementing energy-efficient DRL techniques, such as model compression and adaptive activation functions, can minimize power consumption [148]. Integrating renewable energy sources into IoT devices can enhance sustainability. Scalable approaches, including distributed and multi-agent DRL frameworks, can effectively manage large networks [149]. Developing lightweight DRL architectures tailored to the specific constraints of IoT devices ensures that IDS deployments are both efficient and sustainable.

Table 3.3 Synthesized overview of DRL contributions to IDS in IoT

Functionality	**State Description**	**Action Implemented**	**Reward Mechanism**
Traffic Management	Metrics such as frequency of packets, port utilization, traffic distribution [105, 110]	Routing decisions, traffic shaping [105, 110]	Maximizes efficiency, minimizes congestion [105]
Threat Detection	Network traffic patterns, anomaly scores from classifiers [96, 104]	Classify traffic as normal or malicious, adjust detection parameters [97, 107]	Positive for accurate detection, negative for errors [97]
System Control	System status such as valve states, tank levels [106]	Control actions such as opening/closing valves [106]	Rewards based on system performance and safety criteria [106]
Security Policy Enforcement	Detected threats, policy compliance levels [108, 121]	Enforcement actions (e.g., block, allow, quarantine) [121]	Rewards for policy adherence, penalties for breaches [108]
Adaptive Security	Varying network conditions, attack signatures [125, 126]	Dynamic adaptation of security measures [126]	Rewards for reducing breaches, adapting to new threats [125]

3.5 The future research opportunities in DRL-based IDS

This section identifies the current limitations and gaps in DRL-based IDS, especially in IoT systems. By highlighting these challenges, we emphasize the need for future research to develop innovative solutions that enhance the security and effectiveness of IDS in increasingly complex IoT systems.

3.5.1 Dataset

Traditional datasets (e.g., ISCXIDS2012citeunb2012ids, UNSW-NB15citeunsw2015nb15, and CICIDS2017citeunb2017ids) bring valuable insights. However, the rapid evolution of IoT technologies and threat vectors challenges their relevance. There is a pressing need for datasets that evolve alongside these technologies and threats, enabling DRL models to recognize and mitigate novel and sophisticated attacks adeptly. The heterogeneity inherent in IoT systems, spanning diverse devices, protocols, and applications, presents additional layers of complexity. While datasets such as N-BaIoT [137], BoT-IoT [142], and IoTID20 [139] make significant strides toward embracing this diversity, they may fall short of fully encapsulating the intricate dynamics of real-world IoT networks. This shortfall potentially restricts the generalization capabilities of DRL-based IDS. Another pervasive issue in cybersecurity datasets is class imbalance, where instances of normal traffic significantly outnumber attack instances. This imbalance can skew model training, leading to a propensity for overlooking genuine threats. However, class imbalance is a well-known issue in ML, and there are established solutions, such as oversampling, undersampling, and synthetic data generation techniques such as SMOTE, which can be employed to mitigate this problem and ensure the development of models proficient in accurately identifying attacks without a predisposition towards the majority class.

Integrating real-world data into the training and testing of DRL-based IDS models is crucial for accurately encapsulating the complexities of network traffic and cyber-attack patterns. For instance, the KDDCup 99 [150] dataset offers a blend of diverse network interactions and simulated attacks, setting a benchmark for evaluating the efficacy of IDS solutions. Precise and consistent data labeling is pivotal, especially in the diverse ecosystems of IoT. The accuracy of DRL-based IDS relies heavily on the quality of data labeling. Inaccurate or inconsistent labels can result in models that either flag benign activities as threats due to oversensitivity or fail to detect actual cyber-attacks due to under-sensitivity. Moreover, the considerable computational resources required to process large and complex datasets (e.g., CICDDoS2019 [143]) pose substantial challenges where computational capabilities are limited. Addressing these demands is vital for harnessing the potential of DRL in IDS solutions, necessitating innovative strategies such as model optimization and the adoption of computationally efficient techniques.

Addressing existing datasets' limitations necessitates exploring and including broader data sources. Emerging datasets (e.g., TON-IoT and Edge-IoTset [151]) bring to the fore fresh perspectives and invaluable data for refining IDS mechanisms. These datasets, with their detailed annotations and comprehensive coverage of benign and malicious traffic flows, underscore the importance of a structured approach to tackling IoT security threats. By incorporating such expansive and detailed datasets, the field can move towards developing DRL-based IDS solutions that are theoretically robust and pragmatically effective in the ever-evolving landscape of IoT security.

3.5.2 Enhancing IoT Datasets for Effective IDS

Current IoT datasets often lack the diversity and dynamism necessary to capture the evolving threat landscape, posing challenges for developing robust IDS. For instance, the CICIoT2023 dataset addresses these limitations by providing a comprehensive collection of IoT network traffic, including various attack scenarios, thereby enhancing the representativeness of training data [152].

Synthetic data generation techniques, such as generative Adversarial Networks (GANs), have been employed to create realistic and varied data samples to address the gaps in dataset diversity [153]. Moreover, simulation tools, e.g., NS3 [154], Cooja [155], and OMNeT++ [156], are increasingly used to generate synthetic network data for IoT networks. These tools enable researchers to emulate IoT scenarios with diverse devices, protocols, and attack vectors, offering controlled environments to enrich datasets. Furthermore, integrating FL frameworks facilitates collaborative model training across distributed IoT devices without compromising privacy, enabling more robust IDS development [157]. These advancements, coupled with simulation and synthetic data generation, empower IDS to adapt effectively to the complexities of modern IoT ecosystems.

3.5.3 Difficult reproducibility of papers

This gap hinders the scientific community's ability to conduct rapid and effective experimental evaluations of proposed solutions, especially in real-world IoT settings, and limits the reproducibility of research fundamental principles for scientific progress and validation of results.

Table 3.4 Summary of limitations, open challenges, and future work

Category	Limitations	Open Challenges	Future Work
Distributed IDS	Limited focus on distributed models in IoT.	Develop distributed IDS for resource-limited environments.	Explore lightweight, efficient IDS for IoT using edge computing.
IoT Application Domains	Gaps in coverage of emerging IoT domains.	Expand research to under-explored IoT areas.	Broaden DRL-based IDS for comprehensive security in new domains.
SDN-IoT and DRL	Limited research on SDN in DRL-based IDS.	Improve SDN-IoT integration for better security decisions.	Develop scalable DRL models optimized for SDN-IoT.
Reward Mechanisms	Simplistic rewards in DRL models.	Create adaptive rewards that reflect IoT security complexity.	Design detailed rewards prioritizing threats based on impact.
Resource Optimization	High computational demands of DRL-based IDS.	Balance energy and computational demands without losing accuracy.	Innovate energy-efficient DRL models for IoT constraints.
FL for Privacy	Privacy concerns with centralized data in DRL-IDS.	Use FL to enhance privacy while improving models collaboratively.	Develop federated DRL models for localized, privacy-focused processing.
Dynamic Threat Detection	Challenges adapting to sophisticated attacks with traditional DRL.	Enhance DRL to adjust to new threats dynamically.	Use advanced DRL to learn patterns indicating emerging threats.
Real-World Deployment	Theoretical models not accounting for real IoT constraints.	Address real-world deployment challenges like hardware limits.	Optimize DRL for efficiency in real IoT systems.
Ethical and Legal Considerations	Potential biases in ML algorithms.	Ensure transparent, fair, and legally compliant IDS operations.	Implement strategies for ethical compliance and oversight.
Topology Awareness	Using graph models in complex IoT networks.	Leverage network topology for better threat detection.	Apply graph learning to analyze and protect complex IoT networks.
Live Environment Testing	Lack of performance evaluation in live scenarios.	Conduct real-world testing of DRL-based IDS.	Test IDS in live IoT systems to validate effectiveness.
Security Policy Post-Detection	Gap in intelligent response after threat detection.	Enhance IDS with autonomous policy implementation.	Design IDS to execute context-aware security responses.

3.5.4 The Predilection for Value-Based DRL Approaches

Our investigation into recent advancements in DRL-based IDS for IoT highlights a significant trend in the predominant use of value-based DRL methodologies, especially DQN frameworks, as seen in studies, e.g., [107] and [120]. The popularity of DQN is primarily due to its simplicity and effectiveness in environments where decisions involve discrete choices (e.g., categorizing network traffic as benign or malicious). This approach aligns well with the structured nature of many IoT systems, making it a favored choice in IoT-IDS research. However, our analysis reveals a gap in exploring more advanced DRL techniques, especially actor-critic and policy learning frameworks. While DQN is well-suited for scenarios with discrete actions, there are instances in IoT networks where decisions require a more nuanced approach that cannot be easily categorized into discrete options. In these cases, methods like proximal policy optimization or actor-critic techniques could significantly enhance the flexibility and accuracy of IDS solutions. By leveraging the strengths of both policy-based and value-based approaches, IDS can become more adaptable and effective in detecting and mitigating sophisticated cyber threats in IoT networks.

3.5.5 Transitioning from Centralized to Distributed IDS in IoT

Our investigation reveals a significant shift from traditional centralized IDS to more adaptable, distributed models in the IoT domain. This evolution, driven by the integration of edge computing and a focus on resource efficiency, acknowledges the pressing need for IDS solutions that are lightweight yet capable of operating in the stringent confines of IoT systems. This trend highlights the growing demand for IDS frameworks that integrate seamlessly into the diverse and resource-limited IoT landscape, ensuring comprehensive security without sacrificing device efficiency or autonomy. Some studies, such as the implementation of the DQN framework optimized for edge deployment [120] and applying of FL in a distributed IDS architecture [107], exemplify strategic approaches to minimizing IoT devices' computational demands. These approaches address power and computational constraints and enhance data privacy through localized processing, demonstrating the shift towards resource-aware and privacy-preserving distributed learning and IDS strategies. Despite these advancements, distributed IDS approaches remain underrepresented, indicating significant room for innovation in distributed IDS for IoT. The need for IDS solutions that are acutely aware of and designed to overcome the challenges of energy consumption, CPU load, CPU usage, and memory limitations is more critical than ever.

3.5.6 Imitations in Coverage of IoT Application Domains

The chapters' analysis demonstrates significant gaps in the coverage of IoT application domains, especially in emerging areas such as Agricultural IoT, Environmental Monitoring, and Retail and Supply Chain IoT. Current research has predominantly focused on more established sectors like Industrial IoT, healthcare, and smart cities, leaving these newer domains underexplored. Addressing these gaps is crucial for providing a comprehensive understanding of DRL-based IDS applications across the full spectrum of IoT.

3.5.7 SDN-IoT and DRL-based IDS Integration

Despite the recognized potential of integrating SDN with IoT through DRL to enhance IDS, researchers show a considerable gap in fully leveraging SDN's capabilities in DRL-based IDS frameworks [107]. This underscores the need for more focused efforts to maximize the benefits of SDN in improving the efficiency of IDS in IoT settings. Challenges in this integration include establishing efficient communication channels between DRL-based IDS agents and SDN controllers for seamless data exchange and decision-making. IoT networks' diverse and expansive nature adds complexity, requiring scalable learning algorithms that remain effective without compromising performance. Additionally, it is crucial to balance the use of SDN-IoT's network management features with the preservation of the decision-making autonomy of DRL-based IDS agents to ensure effective and informed security actions.

3.5.8 Complex Reward Mechanisms

The complexity of IoT security, especially in scenarios (e.g., safeguarding smart city infrastructures), necessitates advanced DRL-based IDS equipped with adaptive reward mechanisms. These mechanisms must accurately reflect cyber threats' diverse and sophisticated nature. For example, in protecting a smart city from a DDoS attack, a DRL-based IDS benefits from a reward mechanism that considers the criticality of services, potential public safety impacts, and the system's resilience. Such detailed reward structures enable DRL-based IDS to prioritize threats effectively, ensuring critical infrastructure remains protected. This approach enhances the model's decision-making capabilities and ensures continuous adaptation to new threats, maintaining the integrity and safety of smart city IoT ecosystems.

3.5.9 Optimizing Resource Utilization

Integrating DRL into IDS in the IoT ecosystem highlights significant resource management challenges, including CPU load, CPU usage, and energy consumption. This underscores the need for energy-efficient IDS solutions that manage computational demands. Innovative approaches are needed to reduce energy and computational burdens without compromising threat detection accuracy to enhance the scalability and efficiency of DRL-based IDS. Future advancements may include energy-efficient deployment strategies tailored to the power constraints of IoT devices and the implementation of data-efficient learning mechanisms. These efforts are crucial for ensuring robust security measures that comply with the operational limitations of edge computing and the expansive nature of IoT networks.

3.5.10 FL for Privacy

Integrating FL into DRL-based IDS presents a strategic approach to bolstering privacy and security in IoT networks. This method facilitates a collaborative learning environment, enabling devices across the network to contribute to a collective ML model without centralizing sensitive information. Such a decentralized model is beneficial in IoT settings, addressing data privacy issues and managing the vast amounts of data produced. FL ensures data remains localized, reducing risks linked to data transmission and aligning with privacy regulations. For effective deployment of DRL-based IDS, navigating challenges of data diversity, communication efficiency, and synthesizing learning from disparate sources is essential. This technique enhances the capability of edge devices to autonomously detect threats based on local data, contributing to a more intelligent and privacy-conscious IDS framework. In a recent study, [118] emphasized the importance of incorporating privacy-preserving mechanisms, such as differential privacy, within FL frameworks to ensure secure data exchange while minimizing risks of exposure. Their work highlights how these mechanisms, combined with DRL, can enhance sensitive IoT data's reliability and ethical handling.

3.5.11 Dynamic Threat Detection

The rapidly evolving landscape of IoT security, marked by adversaries devising new attack methods, presents a significant challenge for DRL-based IDS. These systems must recognize known threats and adapt to detect novel, unseen attack vectors. Addressing this challenge requires leveraging transfer learning, enhancing learning algorithms, and employing multi-agent systems with advanced exploration strategies. Transfer learning enables DRL models to apply knowledge from previous tasks to new problems, reducing the training time and data needed for novel threats. Combined with sophisticated learning algorithms, this adaptability enables the processing of complex data structures and identifying patterns indicative of emerging threats with minimal human intervention. Furthermore, multi-agent systems foster a collaborative learning environment where agents collaborate, enhancing the detection capabilities and robustness of the IDS against evolving threats in the IoT ecosystem.

3.5.12 Real-World Deployment

Transitioning DRL-based IDS from theoretical models and simulations to practical, real-world applications encompasses a range of challenges. Real-world deployment must contend with hardware limitations, network constraints, and the need for real-time processing capabilities. DRL models that perform well in simulations may encounter issues in actual IoT systems, where resource constraints and unpredictable network conditions can impact performance. Addressing these challenges involves optimizing DRL algorithms for efficiency and ensuring they operate in real IoT devices' hardware and network limitations. This includes developing robust models capable of processing and responding to threats in real-time.

3.5.13 Ethical and Legal Considerations

The potential for biases in threat detection, rooted in the training data of ML algorithms, raises serious ethical concerns. Such biases could lead to unfair or incorrect threat assessments, mainly impacting healthcare applications in IoT [158]. Moreover, the automated decision-making capability of DRL-based IDS necessitates a thorough evaluation of its implications. In healthcare, where decisions can directly impact patient well-being and data privacy, the accuracy of these systems is paramount. Errors, such as false positives or negatives, could lead to unnecessary alarms or overlooked threats, each with significant implications [159].
The complexity of DRL-based IDS systems raises concerns about fairness, transparency, and the ethical handling of sensitive data. The work by [112] highlights the necessity of integrating privacy-preserving mechanisms to protect sensitive information during the training and deployment of DRL models. Furthermore, the study emphasizes the critical role of fairness-aware algorithms in mitigating biases inherent in datasets, which can otherwise lead to inequality. By incorporating these approaches, DRL-based IDS enhance their trustworthiness, ensure ethical data handling, and address societal demands for equity and accountability in threat mitigation.

3.5.14 Topology Awareness and Graph Modelization

Graph representation learning and graph RL can significantly enhance IDS capabilities in complex IoT networks, such as those in smart cities. By modeling these networks as graphs, it is possible to leverage the inherent structure to detect anomalies and patterns that indicate potential cyber threats. For instance, identifying unusual communication patterns or critical nodes in the IoT network can help preemptively mitigate risks. Graph-based models enable more efficient allocation of monitoring resources, thereby improving IDS performance in complex environments. While the potential of topology awareness and graph modeling in IDS is evident, it is essential to recognize that this area is still underexplored. Future research should focus on integrating these techniques into DRL-based IDS to develop more sophisticated cybersecurity measures, especially in smart city contexts and other intricate IoT systems.

3.5.15 Evaluating DRL-Based IDS in Live Threat Environments

The lack of studies evaluating DRL-based IDS under real-time cyber threat scenarios presents a significant gap in current research. Testing in live environments is crucial for accurately assessing the system's responsiveness and adaptability to dynamic and unpredictable threats. However, several challenges make such testing difficult. One major challenge is real-world cyber threats' inherent unpredictability and complexity, which vary in form, intensity, and timing. This unpredictability makes it challenging to create controlled yet realistic testing scenarios. Additionally, testing in live environments may expose critical systems to potential harm, raising concerns about the safety and integrity of the IoT ecosystem during the evaluation process. Addressing these challenges is essential for practically implementing DRL-based IDS and ensuring that these systems are robust, reliable, and effective in real-world applications.

3.5.16 Implementing Security Policies

Assume that in a smart city scenario, a DRL-based IDS detects a sophisticated DDoS attack aimed at disrupting traffic management during peak hours. While the IDS identifies the threat, the challenge extends to the post-detection actions to mitigate the attack and preserve the city's infrastructure integrity. Ideally, the IDS should activate a predefined security policy involving rerouting network traffic, isolating affected systems, or temporarily disabling certain functions to contain the attack. In a recent study, [96] proposed a Power-Law-Based Blocking Time Management (PL-BTM) mechanism that dynamically adjusts the blocking time of IP addresses based on their cumulative threat levels. This approach helps defense strategies for specific threat scenarios, ensuring minimal disruption to essential services such as traffic lights and emergency responses while effectively containing malicious activity. However, the lack of research on implementing these automated post-detection responses highlights a significant gap in current IDS capabilities, emphasizing the need for comprehensive security frameworks that include threat detection and effective countermeasures to protect IoT systems against complex cyber threats.

3.6 Chapter Summary

This chapter highlights the significant progress in applying DRL-based IDS in IoT systems from 2014 to 2024. DRL methodologies have proven adaptable and effective in enhancing the security of IoT networks, offering improved accuracy and flexibility in detecting and mitigating evolving cyber threats. However, challenges remain, including reliance on outdated datasets, reproducibility issues, and the need for more scalable solutions. Addressing these challenges is essential for further advancing the field. Future research should focus on exploring federated learning, policy learning methods, and the integration of high-level threat intelligence into DRL models, which hold promise for improving the effectiveness and efficiency of IDS solutions.

CHAPTER 4 ARTICLE 2: EVALUATING MACHINE LEARNING-DRIVEN INTRUSION DETECTION SYSTEMS IN IOT: PERFORMANCE AND ENERGY CONSUMPTION

4.1 Article Metadata

- **Title:** Evaluating Machine Learning-Driven Intrusion Detection Systems in IoT: Performance and Energy Consumption
- **Authors:** Saeid Jamshidi, Kawser Wazed Nafi, Amin Nikanjam, Foutse Khomh
- **Affiliations:**
  - SWAT, Polytechnique Montréal, H3T 1J4, Quebec, Canada
  - Huawei Distributed Scheduling and Data Engine Lab, Montréal, Quebec, Canada
- **Submitted Date:** 29 September 2024
- **Accepted Date:** 31 March 2025
- **DOI:** `https://doi.org/10.1016/j.cie.2025.111103`
- **Journal:** *Computers & Industrial Engineering*

4.2 Chapter Overview

In the network security landscape, the integration of ML-based IDS represents a significant leap forward, especially in the domains of the IoT and SDN. Such ML-based IDSs are crucial for improving security infrastructures, and their importance is increasingly pronounced in IoT systems. However, despite the rapid advancement of ML-based IDS, a gap remains in understanding their impact on critical performance metrics (e.g., CPU load, energy consumption, and CPU usage) in resource-constrained IoT devices. This becomes especially crucial in scenarios involving real-time cyber threats that challenge IoT devices in both public and private networks.
To address this gap, this chapter presents an empirical study that evaluates the impact of state-of-the-art ML-based IDSs on performance metrics, including CPU usage, energy consumption, and CPU load, in both the absence and presence of real-time cyber threats, with a specific focus on their deployment at the edge of IoT infrastructures. We also incorporate

SDN to evaluate the comparative performance of ML-based IDSs with and without SDN. To do so, we focus on the impact of both SDN's centralized control and dynamic resource management on the performance metrics of an IoT system. Finally, we analyze our findings using statistical analysis, using the Analysis of Variance (ANOVA). Our findings demonstrate that traditional ML-based IDS, when implemented at the edge gateway with and without SDN architecture, significantly affects performance metrics against cyber threats compared to DL-based ones. Also, we observed substantial increases in energy consumption, CPU usage, and CPU load during real-time cyber threat scenarios at the edge, underscoring the resource-intensive nature of these systems. This chapter fills the existing knowledge void and delivers essential insights into the operational dynamics of ML-based IDS at edge gateway in IoT systems.

4.3 Study Design

The rapid expansion of the IoT has ushered in an era where data flows seamlessly across various sectors, driving profound changes in how devices interact [160] [58]. This intricate IoT ecosystem, composed of countless devices, sensors, and intelligent nodes, has fundamentally reshaped how we think about device communication, significantly minimizing the need for human involvement [161]. The integration of SDN within the IoT landscape represents a significant step forward, creating a unified IoT-SDN framework that offers centralized control, improved network management, and stronger security measures [162] [55].

The rapid expansion of IoT, driven by the interconnection of millions of devices via WSNs, presents significant challenges [163]. These challenges stem mainly from these devices' limited memory, power, and battery life, highlighting the need for optimized computing and advanced data analysis techniques [164]. Deploying SDN within this framework aims to overcome these obstacles by offering a streamlined, secure network infrastructure that facilitates effective resource allocation and enhanced threat management.

Given the widespread security vulnerabilities in IoT networks, such as service disruptions and unauthorized access, the importance of ML-based IDS has grown [41]. ML-based IDS are crucial for protecting network integrity due to their ability to adapt dynamically and effectively identify threats [165] [166] [167].

However, despite advancements in developing ML-based IDS for IoT, several critical gaps remain, as highlighted by Tekin et al. [168]. While previous research has examined ML-based IDS's performance in controlled, static testbed environments, there is a significant gap in understanding how these systems operate under the dynamic conditions of real-time cyber threats, especially when IoT is integrated with SDN. Moreover, while the potential of SDN to significantly enhance resource management in IoT systems is widely acknowledged [169] [170] [171], there is a lack of empirical evidence on how SDN interacts with ML-based IDS during cyber threats.

In this study, we set two primary objectives designed to deepen our understanding of network performance metrics in IoT. Firstly, we assess the impact of deploying ML-based IDS at the edge gateway, mainly focusing on ML-based IDS performance metrics under real-time cyber threats. Secondly, we explore the impact of integrating SDN with our testbed, again at edge gateway, to evaluate its influence on performance metrics under similar cyber threats. The rationale behind incorporating SDN into our testbed is its potential to improve resource management in IoT systems significantly [172] [173]. We conduct a comparative analysis of the performance of seven state-of-the-art ML-based IDSs in two distinct setups: firstly, at the edge gateway, and secondly, in a similar setup augmented with SDN integration at the edge gateway, all under real-time cyber threats. This analysis is designed to elucidate the impact of SDN on performance metrics and resource management in IoT systems, especially highlighting how SDN integration can optimize the operational efficiency and resilience of IoT networks against the backdrop of evolving cyber threats. To summarize, this chapter makes the following contributions:

- Assessing performance metrics of ML-based IDS in IoT systems under real-time cyber threats: Our investigation revealed the significant impact of seven ML-based IDS on the performance at the edge, specifically measuring CPU usage, CPU load, and energy consumption amidst cyber threats. Utilizing ANOVA, we clarify the operational consequences of deploying these sophisticated IDSs on the edge.
- Evaluating the impact of ML-based IDS at edge integrated with SDN: We evaluated seven ML-based IDS performance metrics at the edge gateway system integrated with SDN. Utilizing ANOVA, we clarify the impact of the integrated SDN with IoT on deploying these sophisticated IDS under real-time cyber threats.
- Proposing a plugin-based ML-based IDS test suite: This test suite comes with a group of available datasets and available ML-based IDSs and allows the users to define their own IoT and SDN applications and test their ML-based IDSs and models in terms of detection accuracy and performance metrics. Researchers can efficiently perform comparative analyses for their algorithms and models with other available algorithms and models. The test suite is publicly available (section 4.8) for researchers and practitioners to reuse.

4.4 Designing an Empirical Study for ML-Driven Intrusion Detection in IoT Systems

This section describes our methodology to evaluate the impact of specific ML-based IDSs using selected performance metrics. We first mention our Research Questions (RQs), followed by an explanation of the experimental design and the metrics used to evaluate the impact of the ML-based IDS.

4.4.1 Research questions(RQs)

Our research aims to address the following RQs:

- **RQ1: How do ML-based IDSs impact CPU usage, CPU load, and energy consumption at the edge gateway without SDN during real-time cyber threats?**
  This RQ examines the impact of ML-based IDSs on crucial performance metrics, specifically CPU usage, CPU load, and energy consumption, at edge gateway not integrated with SDN. It focuses on analyzing the performance of seven state-of-the-art ML-based IDSs and their impacts on these key metrics in the face of diverse cyber threats.

- **RQ2: What are the differences in CPU usage, CPU load, and energy consumption impacts of ML-based IDS at the edge gateway with SDN integration during real-time cyber threats?**
  This RQ explores how ML-based IDSs influence CPU usage, CPU load, and energy consumption at the edge gateway integrated with SDN. It involves analyzing the impacts of various ML-based IDSs on these essential performance metrics under various cyber threats.

4.4.2 DataSet

In our study, we used the CICIDS2017 data set [174], a highly regarded resource organized by the Canadian Institute for Cybersecurity. This dataset is recognized as one of the gold standards in cybersecurity research, capturing a broad spectrum of benign network activities and the latest cyberattacks [175]. CICIDS2017 is designed to simulate real-world network environments, making it an essential resource for researchers to thoroughly test and validate advanced IDS. The breadth and diversity of the asset highlight its importance, making it necessary for those aiming to strengthen network security paradigms.

4.4.3 The ML-based IDS

Numerous ML-based IDS have been developed by researchers [168] [176] [177] [178]. However, we had a significant challenge in reviewing these publications and selecting some for our study. Most did not make their solutions' applications or source code publicly available. This lack of transparency hinders the ability to experiment with these works in real IoT devices. This omission complicates, and may even prevent, the objective comparison of the proposed solutions. Consequently, to initiate our study, it became necessary to independently implement all ML-based IDS that have been previously utilized, except the ML-based IDS proposed by [87], which shared their code ML-based IDS available to researchers. In this section, we explore the implementation process of seven ML-based IDSs that we have developed: DT, KNN, RF, LSTM, CNN, and a hybrid model of LSTM and CNN. Table 4.3 presents a comparative analysis of the performance metrics of ML-based IDS.

Table 4.1 Distribution of labeled IoT-SDN attacks in the dataset

IoT Attack Labels	**No of labeled entries**
BENIGN	2271320
DoS Hulk	230124
Port Scan	158804
DDoS	128025
DoS GoldenEye	10293
FTP-Patator	7935
SSH-Patator	5897
DoS slowloris	5796
DoS Slowhttptest	5499
Bot	1956
Web Attack & Brute Force	1507
Web Attack & XSS	652
Infiltration	36
Web Attack & SQL Injection	21
Heartbleed	11

4.4.4 DT, KNN, RF

We have developed and deployed DT-based IDS, RF-based IDS, and KNN-based IDS [179], each specifically designed to improve security policy. The foundation of these models is a preprocessing technique applied to the selected CICIDS 2017 dataset. The dataset features various simulated cyber-attack scenarios alongside standard traffic data. It encompasses multiple numerical attributes, including but not limited to packet sizes, flow durations, and bytes per flow, which are critical for analyzing network behavior and detecting anomalies. We applied min-max normalization as our initial preprocessing step to ensure uniformity across these diverse numerical attributes and mitigate scale discrepancies. Missing values were imputed to preserve the integrity of the data. The LabelEncoder [180] was utilized to convert labels into a format suitable for ML techniques. An essential aspect of our methodology is to divide the selected dataset into training and testing subsets. For the first RQ, we adopted 80% training and 20% testing, aligning with standard practices in ML model development. This adjustment was made to accommodate the different requirements of each research phase. As shown in Table 4.1, the dataset has five classes (Benign, DDoS, DoS, Brute force, and Port scan) with significantly more entries than the remaining ten classes, which contain fewer samples. SMOTE [181] with auto-sampling was employed to address the class imbalance issue in the dataset. This technique effectively augmented the representation of underrepresented classes, leading to a more balanced dataset for training purposes.

Table 4.2 Comparison of structure and accuracy of different Neural Network models in IDS for IoT-SDN network

Dataset	CICIDS2017	CICIDS2017	CICIDS2017	CICIDS2017
Categories	15	15	15	15
Model	LSTM	LSTM+CNN	CNN	EIDM
Layers	10	11	8	12
Parameters	56386	12795	3497	48735
Structure details	Dense (64) Dense (128) LSTM (128) LSTM (256) Dense (128) Dense (48) Dense (15)	Dense (64) Conv1D (64, 10) Conv1D (64, 10) MaxPooling1D (2) LSTM (128) LSTM (64) Dense (64) Dense (15)	Conv1D (16,30) Conv1D (16,30) MaxPooling1D (2) Flatten() Dense (32) Dense (15)	Dense (120) Conv1D(80, 20) MaxPooling1D (2) Dense (120) Dense (100) Dense (80) Dense (60) Dense (60) Dense (40) Dense (15)
Training Accuracy (%)	97.72%	98.77%	97.92%	99.57%
Testing Accuracy (%)	93.86%	95.75%	94.74%	99.56%

4.4.5 CNN

In our research, we deployed a CNN-based IDS tailored for our experimental testbed. The configuration details of the CNN model, including its layers, parameters, and architecture specifics, are outlined in Table 4.2.

4.4.6 LSTM

In our investigation, we implemented an LSTM-based IDS specifically for our testbeds. The detailed architecture and parameters of the LSTM model, crucial for its operation in our IDS, are thoroughly presented in Table 4.2.

4.4.7 Hybrid model of LSTM and CNN

In our exploration, we implemented a hybrid LSTM and CNN architectures model to create an advanced IDS tailored to our experimental setup. This architecture has already been tested in various scenarios [182] [183] [86]. The intricate configuration of this hybrid LSTM and CNN model, which leverages the strengths of both LSTM and CNN to enhance detection capabilities, is detailed in Table 4.2.
The goal of using the hybridization of LSTM and CNN is twofold. First, CNN can drop the non-impactful features and select only the impactful ones (feature engineering). At the same time, it helps to learn the features in a Spatial Hierarchical manner [184]. Second, from our dataset, we got 77 features. As it is unknown which features are impactful from the given features, we applied a 2 1-dimensional CNN layer followed by a max-pooling layer to find the impactful features by learning the 10 nearby features together (kernel size 10). This helps us to create new feature representations where the impactful ones are sustained. Later, we fed these newly derived features directly to 2 LSTM layers. This step helps to learn the spatial and temporal features from CNN, resulting in feature representations presented in context and awarded. Finally, we applied 2 Dense layers to regress the feature representations generated from previous CNN and LSTM layers into 15 classes. This process helps us learn the input features more deeply and increase the classification accuracy.

4.4.8 Experimental Design

To address RQ1, we designed a testbed incorporating two Raspberry Pi 4 Model B units as edge gateways. Each unit is equipped with 8GB of RAM and a 1.5GHz 64-bit quad-core CPU, providing a realistic environment for evaluating the computational impact of ML-based IDS at the edge gateway. Our study evaluates the performance of seven ML-based IDS models: DT, KNN, RF, LSTM, CNN, EIDM, and a hybrid of LSTM and CNN model, selected for their established effectiveness in cybersecurity. We conducted controlled experiments in IoT-edge networks to assess these IDS models, simulating a range of cyber threats(e.g., BENIGN, DDoS, DoS, Brute force attacks, and the Port scan) using Kali Linux [185]. These experiments enabled us to analyze the IDS models' impact on critical performance metrics, specifically CPU usage, CPU load, and energy consumption.
To address RQ2, we extended our testbed by integrating the edge gateway with the Ryu controller, establishing an SDN-based environment. Ryu, an open-source Python-based SDN controller [186], provides centralized traffic management, enhancing resource allocation and security analysis. We further utilized Mininet [187] to simulate a realistic SDN infrastructure consisting of eighteen hosts, six switches, and a Ryu controller, mirroring real-world network

Table 4.3 Performance Comparison of ML-based IDS

	DT	**KNN**	**RF**	**LSTM**	**LSTM+CNN**	**CNN**
Accuracy	0.9985	0.9967	0.9981	0.9386	0.9575	0.9474
Precision	0.9985	0.9966	0.9980	0.9771	0.9877	0.9792
Recall	0.9985	0.9967	0.9981	0.9524	0.9645	0.9611
F1-Score	0.9985	0.9966	0.9980	0.9646	0.9760	0.9701

conditions.

4.4.9 Metrics

We evaluated CPU usage, CPU load, and energy consumption in our test beds in the context of ML-based IDS during cyber threat scenarios. We employed the ANOVA [188] to ensure an objective assessment of the performance of various ML-based IDS.

4.4.10 CPU Load CPU Usage

IDS, especially at the edge and SDN environments. CPU usage measures the percentage of the CPU's current capacity, reflecting how much processing power is dedicated to task execution. High CPU usage in an IDS can signal extensive computational demands, potentially impacting the performance of other tasks and system responsiveness, a concern in resource-limited IoT settings. Efficient IDS, especially those utilizing ML techniques, must manage CPU usage carefully to balance detection accuracy with minimal resource use. Excessive CPU usage can slow IDS's real-time network traffic processing, leading to delays or missed attack detection. On the other hand, CPU load indicates the number of processes waiting to be executed, providing an understanding of the CPU's workload. An increase in CPU load might suggest heavy network traffic or numerous attack attempts, highlighting the risk of system overload. Monitoring CPU load allows for early identification of potential bottlenecks, ensuring that IDS operations do not adversely impact system performance. In SDN-enabled IoT edge systems, adept CPU load management is vital to distribute tasks between IDS and other network-efficient functions, ensuring optimal resource allocation and system performance. Both CPU usage and load are pivotal metrics for assessing IDS efficacy in environments where resources are constrained, e.g., at the edge gateway [189] [190] [191].

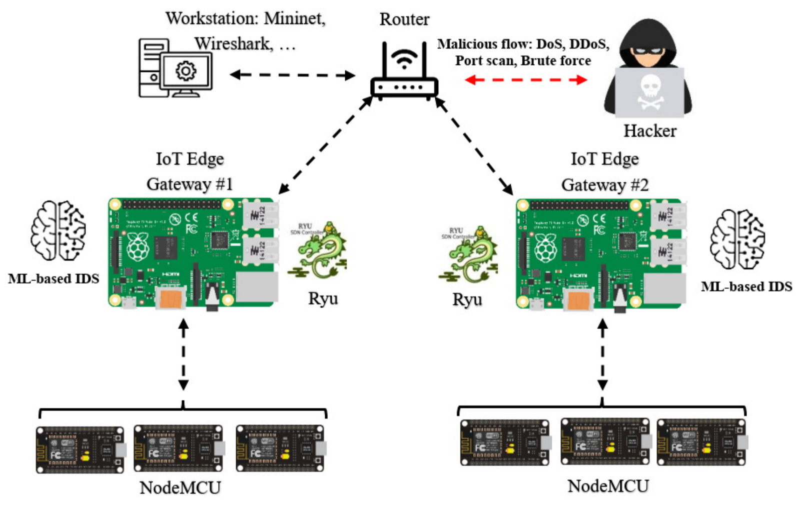

Figure 4.1 IoT-edge testbed topology, illustrating non-SDN and SDN-enabled setups.

4.4.11 CPU Performance Metrics

To assess the computational impact of ML-based IDS, we analyze both CPU load and CPU usage, as these metrics provide complementary insights into system performance. CPU usage is typically expressed as a percentage, indicating the proportion of processing power utilized at a given moment. In contrast, CPU load is presented as a numerical value, representing the average number of active processes waiting for CPU execution over a specific time interval. Moreover, while CPU load can be converted into a percentage, it provides a more detailed view of system stress, especially in multi-core environments. In a multi-core processor, a load value of 1.0 on a single-core system indicates full utilization. In contrast, on a quad-core system, a load of 1.0 suggests that only 25% of the total available processing capacity is used. This distinction is crucial when interpreting our results, as high CPU load does not always imply that the system is at risk of overutilization—it depends on the number of available processing cores and the workload distribution.

4.4.12 Energy Consumption

Energy consumption, often measured in watt-hours or joules, quantifies the amount of energy a device or system expended during its operation. In IoT hardware, where many devices are battery-powered or operate in energy-constrained environments, efficient energy consumption is desirable and necessary. Devices (e.g., sensors and actuators) and even more complex IoT nodes must be designed to perform their tasks while consuming minimal energy, ensuring longevity, and reducing the need for frequent battery replacements or recharges. Moreover,

IoT devices integrated with SDN bring a new dimension to the energy conversation; SDN centralizes network control, dynamically optimizing network resources based on real-time demands. Although this centralization offers enhanced flexibility and scalability, it also means that the network's core components must be energy efficient. In IoT systems, where potentially thousands or even millions of devices communicate and exchange data, even minor inefficiencies in energy consumption can accumulate, leading to significant energy drains. Integrating ML-based IDS into the edge gateway emphasizes the need to consider energy metrics critically. ML-based IDS are inherently data-intensive, requiring substantial computational resources to process large datasets for detecting and mitigating security threats. Although these systems offer invaluable security enhancements, their operation can be energy-intensive. Therefore, measuring and optimizing the energy consumption of ML-based IDS is crucial to ensure they deliver effective security measures without unduly burdening the system's energy resources. This balance is essential for maintaining the sustainability and efficiency of the edge gateway, where energy efficiency is often a key concern.
We employed PowerTop [192], a robust tool, to precisely gauge and examine the energy consumption in two separate testbed configurations: the edge gateway integrated with SDN and without SDN. PowerTop's sophisticated monitoring capabilities allowed us to gain insights into these testbeds' energy consumption patterns and processor activity.

4.4.13 Designed cyber threats

For our research, we focused on analyzing DDoS, DoS, brute force attacks, and the port scan. We chose these specific types of attacks since they were already categorized in the employed dataset. These cyber threats are prevalent and pose substantial risks in the field of cybersecurity. Below, a concise summary of each is presented:

- **A Denial-of-Service (DoS):** At the edge, DoS attacks are critical cybersecurity threats that disrupt device and service operations by flooding systems with excessive requests and consuming vital resources (e.g., bandwidth, processing power, and memory). This overload prevents the system from serving legitimate users, blocking access to essential operations. The distributed, resource-constrained nature of the edge makes them especially susceptible to DoS attacks. The vulnerability of these devices, coupled with their interconnectedness, means that an attack on a single device can significantly compromise the entire network's functionality and security [193].

- **A distributed denial-of-service (DDoS):** A DDoS attack is a coordinated effort where multiple attackers from different locations flood a specific target, such as a server or network at the edge, with excessive traffic. The goal is to deplete the target's resources, causing severe service disruptions or a complete shutdown. Unlike traditional DoS attacks, which come from a single source, DDoS attacks are distributed across numerous sources, making them harder to defend against. This distributed nature makes DDoS attacks especially dangerous at the edge, where the interconnected and resource-constrained devices can exacerbate the attack's impact, potentially crippling the entire network [194].

- **Brute Force:** A brute-force attack involves an attacker systematically attempting to gain unauthorized access to a system by trying every possible combination, such as trying every key until one works. With its many interconnected devices and varying security levels, the edge is especially vulnerable to such attacks. Attackers exploit these weaknesses by repeatedly guessing passwords, encryption keys, or access codes, which seriously threatens the integrity and confidentiality of data at the edge gateway [195].
- **Port Scan:**A port scan aims to identify a target system's open ports. By identifying open ports and the services running on them at the edge, attackers can uncover and exploit vulnerabilities, posing a serious threat to the security and integrity of the edge gateway [196].

4.4.14 Analysis method for energy consumption, CPU usage, CPU load

We used ANOVA to assess our observed results. ANOVA is an indispensable statistical tool for testing the null hypothesis that posits the equivalence of group means. Our study specifically employed one-way ANOVA to examine the impact of a singular independent variable on the evaluated systems. This method relies on several crucial assumptions, including the necessity for the data to exhibit a normal distribution, the variances between groups being equal (homogeneity of variance), and all observations being independent.
In addition, we conducted 15 separate tests on ML-based IDS to measure CPU load, CPU usage, and energy consumption under various cyber threats. This rigorous approach allowed us to leverage the F statistic, which quantifies the variance ratio between the means of different groups to the variance in the groups. A significant F-statistic, together with a p-value of ≤ 0.05, denotes statistically significant differences between group means, underscoring the efficacy of our testing methodology. By implementing this robust statistical framework, we have thoroughly evaluated the performance of various ML-based IDS models in response to different cyber threats. This analysis has allowed us to identify specific models that demonstrate resilience or efficiency against multiple attacks and require increased computational resources or energy consumption. While CPU load is a key performance metric for IDS

evaluation, it is also crucial to consider its impact on IoT device availability and reliability. Excessive CPU consumption by an IDS can degrade the device's primary functions, leading to slow response times or system failures. This is especially critical in real-time applications such as healthcare, industrial automation, and smart home security, where device downtime can have serious consequences. An IDS must enhance security without inadvertently causing an attack, such as a DDoS condition due to resource exhaustion. In addition, through these fifteen iterations of testing, ANOVA has enabled us to validate significant differences in IDS performance metrics (e.g., detection accuracy, false positive rates), CPU load, CPU usage, and energy consumption across diverse scenarios. This methodological approach provides a detailed examination of how different IDS models respond to varied threats, establishing a solid statistical foundation for assessing the efficacy of each model in a controlled environment. By distinguishing between performance differences attributable to the models' inherent capabilities and those due to random variation, our use of ANOVA has proven to be critical. It aids in identifying the most resource-efficient and reliable IDS, thereby guiding the selection process for optimal cybersecurity defenses and enhancing our management and understanding of IDS performance under cyber threat conditions [197] [198].

4.4.15 TestSuite

To initiate the research work presented in this chapter and to facilitate the environment for further research and testing, we introduce a versatile test suite designed to experiment with and evaluate ML-based IDS in SDN environments. Unlike conventional experimental testbeds, our test suite is an extensible framework equipped with predefined APIs and a selection of pre-integrated algorithms, facilitating the seamless integration and testing of novel IDS models. Another good contribution to our test suite is that users can execute their experiments on it without Raspberry Pi or any other hardware support. As discussed in the previous paragraph, the test suite is developed following the plug-in architecture feature. This ensures that the user can easily integrate their algorithm into the test suite and test the accuracy, energy consumption, and CPU usage with or without security threats. Users can create their own IoT-SDN network and complexity in the network and generate any number of security breaching attacks. This approach not only simplifies the validation process of IDS models in a realistic network scenario but also encourages the exploration of innovative IDS methodologies by providing a solid foundation of tools and benchmarks. We have made the test suite available with the same configuration discussed in Section 4.4.8. We integrated the same tools for creating an IoT-SDN network, generating security attacks, and measuring IDS accuracy, energy consumption, CPU usage, etc. Through its design, the test suite aims to advance the development and thorough evaluation of cutting-edge IDS solutions, significantly enhancing network security in the era of SDN.

4.5 Experimental Results and Analysis

This section discusses our experimental results and findings. After presenting our results, we conducted an in-depth statistical analysis using ANOVA. This analysis aims to illuminate the implications and insights that emerge from the experimental results, providing an understanding of the efficacy and nuances of each IDS under study.

4.5.1 Experimental finding for RQ1

CPU Load:

We tested ML-based IDSs under various cyberattack scenarios to assess their impact and strain on our testbed. The types of cyberattacks we considered include DDoS, DoS, brute force attacks, and the port scan. Moreover, we conducted the ANOVA, focusing on CPU load variations in our testbed. Figure 4.2 illustrates a comparative analysis of the average CPU load among different ML-based IDS models in the presence of various types of cyberattacks. The DL-based IDS (CNN, LSTM, combined model of LSTM and CNN, and EIDM) consistently maintain lower CPU loads across all attack types, demonstrating their efficiency in resource utilization during inference. In contrast, traditional ML-based IDS, such as KNN, DT, and RF, exhibit significantly higher CPU loads, especially under brute force and DDoS attacks, with KNN and DT being the most resource-intensive. This is because DL models, such as CNN and LSTM, efficiently handle computations in parallel and are optimized for inference. In contrast, traditional models (e.g., KNN and DT) require more repeated, resource-heavy calculations, such as distance computations in KNN or recursive splitting in DTs, especially under large-scale attacks.

Statistical Findings:

Table 4.4 ANOVA results: CPU Load for ML-based IDS under DDoS.

Source	Degrees of Freedom	Sum of Squares	Mean Square	F Statistic	P-value
Between groups	6	21609.87	3601.64	60.40	< 0.05
In groups	91	5426.49	59.63		
Total	97	27036.36	278.73		

We conducted an ANOVA, and the results presented in Table 4.4 illuminate significant differences in CPU load among diverse ML-based IDS under DDoS, underscored by F-statistic of 60.40 and a p-value < 0.05. This F-statistic delineates the contrast in CPU load variance

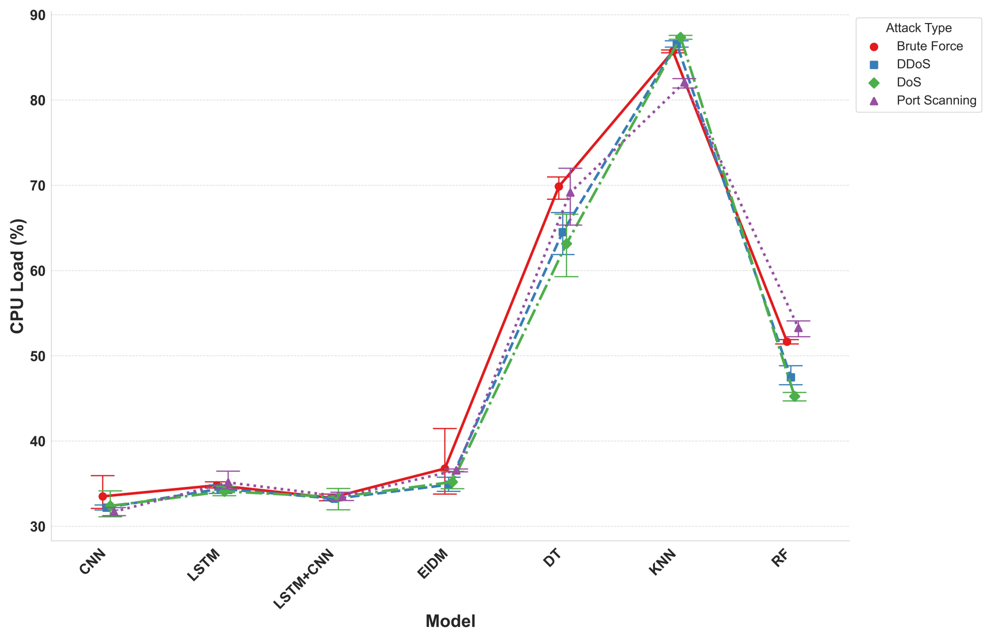

Figure 4.2 The Average CPU load of ML-based IDS under cyber threats.

across ML-based IDSs against the variance in, highlighting a significant influence of IDS selection on CPU load. The remarkably low p-value corroborates this finding, conclusively demonstrating the substantial differences in CPU load among the IDSs. Furthermore, we observed similar p-values (< 0.05) across other attacks, including brute force, DoS, and the port scan, so we do not report them. This reinforces the presence of marked differences in CPU load among diverse ML-based IDS under different cyber threats.

Finding

DL-based IDS, such as CNN, LSTM, and hybrids, perform more efficiently in managing computational demands across diverse types of cyber threats than traditional ML-based IDS, such as KNN, DT, and RF, as they exhibit higher CPU loads at the edge. This pattern suggests that DL-based IDS' intrinsic efficiency is not attack-specific but rooted in their architecture, making them especially suited for real-time applications at edge gateway. These results are expected, as traditional ML-based IDS (e.g., KNN, DT, RF) perform computationally expensive operations during inference, unlike DL-based IDS, which optimizes processing through parallelization and learned feature extraction.

CPU Usage:

Figure 4.3 compares the average CPU usage of various ML-based IDS models under different cyberattacks. The KNN model consistently exhibits the highest CPU usage across all attack

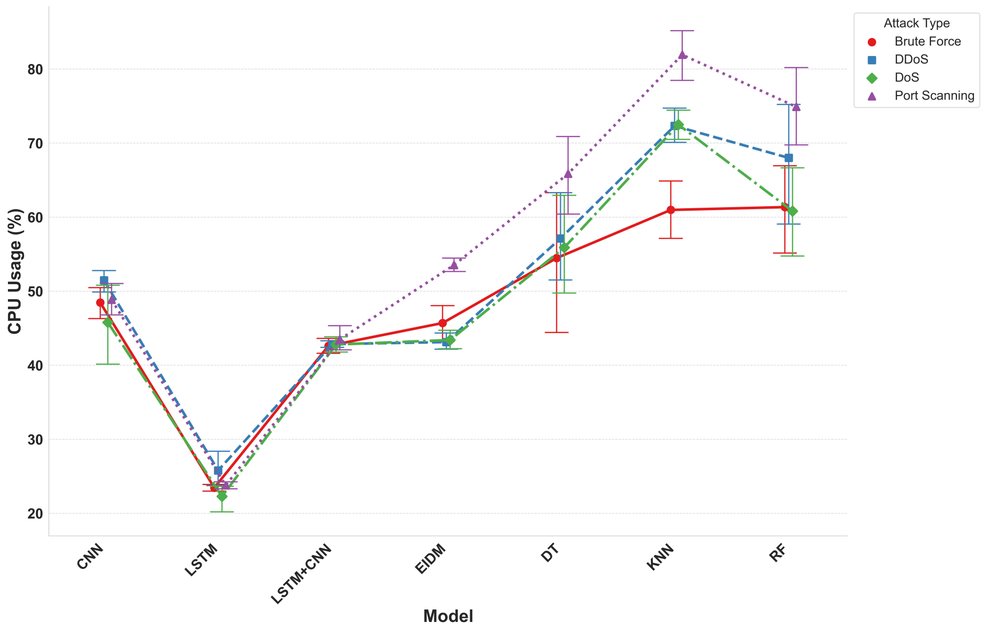

Figure 4.3 The Average CPU usage of ML-based IDS under cyber threats.

types, indicating its high computational demand, which limits its use in resource-constrained environments. The RF and DT models are also CPU-bound, though they are less intensive than KNN. In contrast, the LSTM model demonstrates the lowest CPU usage, making it the most efficient option for scenarios where minimizing resource consumption is critical. The hybrid of the LSTM and CNN model, along with the CNN and EIDM models, offers a balance between inference accuracy and computational efficiency, making them viable choices for environments with moderate resource availability.

Statistical Findings:

Table 4.5 presents our ANOVA results. Our results reveal significant differences in CPU

Table 4.5 ANOVA results: CPU Usage for ML-based IDS under DDoS.

Source	Degrees of Freedom	Sum of Squares	Mean Square	F Statistic	P-value
Between groups	6	21609.86	3601.64	60.39	< 0.05
In groups	91	5426.49	59.62		
Total	97	27036.36	278.73		

load among diverse ML-based IDS under DDoS, as evidenced by a compelling F-statistic of 60.39 and a p-value < 0.05. This F-statistic highlights the variance in CPU load across IDS groups compared to the variance in, underscoring a significant impact of IDS selection on CPU load. The exceedingly small p-value further supports this conclusion. Moreover, we observed similar p-values (below 0.05) across various cyber threats, such as brute force, DoS, and the port scan, so we do not report those results.

Finding

Our analysis reveals that traditional ML-based IDS, such as KNN, DT, and RF, exhibit increased CPU usage under various cyber threats, thus posing challenges for the edge. Also, LSTM and other DL-based IDS exhibit lower CPU demands. This consistent efficiency across various attacks highlights the benefit of adopting DL-based IDS at the edge gateway. The increased CPU usage of KNN, DT, and RF reflects their reliance on instance-based and tree-splitting operations, which require repeated evaluations. In contrast, DL models efficiently process data in structured layers, reducing computational strain.

Energy consumption:

Figure 4.4 shows that the LSTM and DT models are the most energy-efficient across different types of cyberattacks, consistently exhibiting the lowest energy consumption. The CNN model also performs efficiently, with slightly higher energy usage. The LSTM, CNN model hybrid, and EIDM have moderate energy consumption, balancing complexity and efficiency. In contrast, the KNN model has the highest energy consumption across all scenarios, making it less suitable for energy-constrained environments. The RF model falls in between, with moderate energy demands.

Statistical Findings:

We conducted the ANOVA, and the results presented in Table 4.6 reveal significant differences in energy consumption among diverse ML-based IDS under DDoS, underscored by a F-statistic of 57.44 and a p-value of < 0.05. This F-statistic delineates the contrast in energy consumption variance across the group of IDSs against the variance in, highlighting a significant influence of IDS selection on energy consumption. The extremely low p-value further supports this conclusion, conclusively demonstrating the substantial differences in energy consumption among the IDSs. In addition, we observed similar p-values (< 0.05) for other cyber threats, such as brute force, DoS, and the port scan, so we do not report the results. This observation demonstrates significant differences in energy consumed among various ML-based IDS when faced with differing cyber threats.

Table 4.6 ANOVA results: energy consumption for ML-based IDS under DDoS.

Source	Degrees of Freedom	Sum of Squares	Mean Square	F Statistic	P-value
Between groups	6	47732.07	7955.34	57.44	< 0.05
In groups	98	13571.72	138.48		
Total	104	61303.80	589.45		

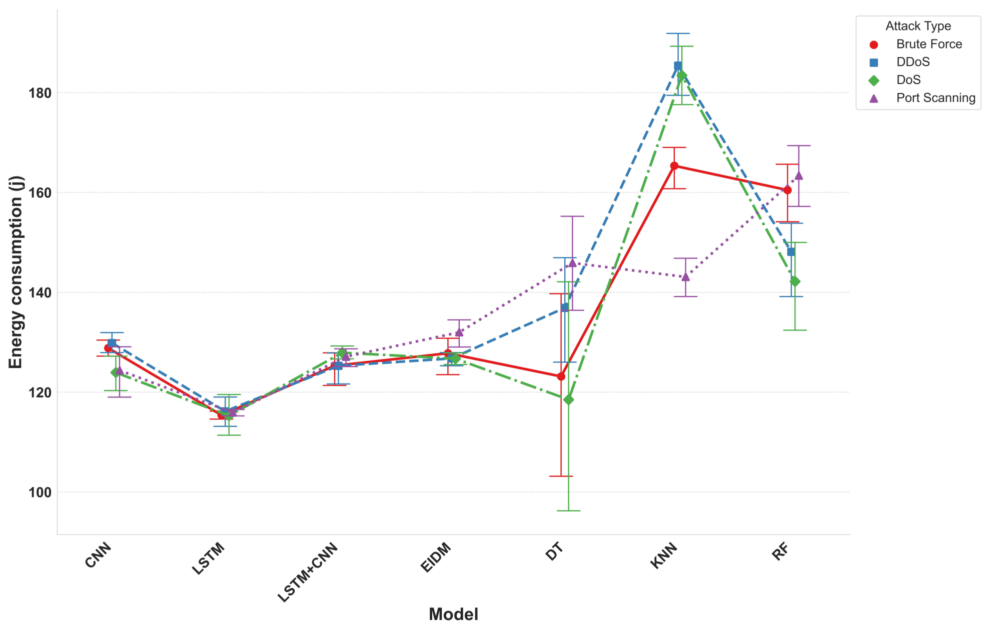

Figure 4.4 The Average Energy consumption of ML-based IDS under cyber threats.

Finding

Our analysis concludes a marked discrepancy in energy consumption, with traditional ML-based IDS such as KNN, RF, and DT exhibiting significantly higher energy consumption under cyber threats such as DDoS and brute force, a drawback for energy-constrained at the edge. In contrast, DL-based IDS models, LSTM, CNN, EIDM, and their hybrids excel in energy efficiency, making them the preferable choice for the edge. Traditional ML models' higher energy consumption results from their iterative computations and lack of optimized inference paths, making them less viable for real-time IoT applications where power efficiency is crucial.

4.5.2 Experimental finding for RQ2

This section presents our experimental results for IoT-edge devices with SDN integration during real-time cyber threats.

CPU Load:

In Figure 4.5, we illustrate the CPU load of various ML-based IDS models under different cyberattacks in an SDN-enabled at the edge gateway. The analysis shows that KNN and DT models have the highest CPU load, especially during DDoS and DoS, indicating significant resource demands at the edge. Conversely, the LSTM model demonstrates the lowest CPU load, highlighting its efficiency in resource management. The CNN model also performs

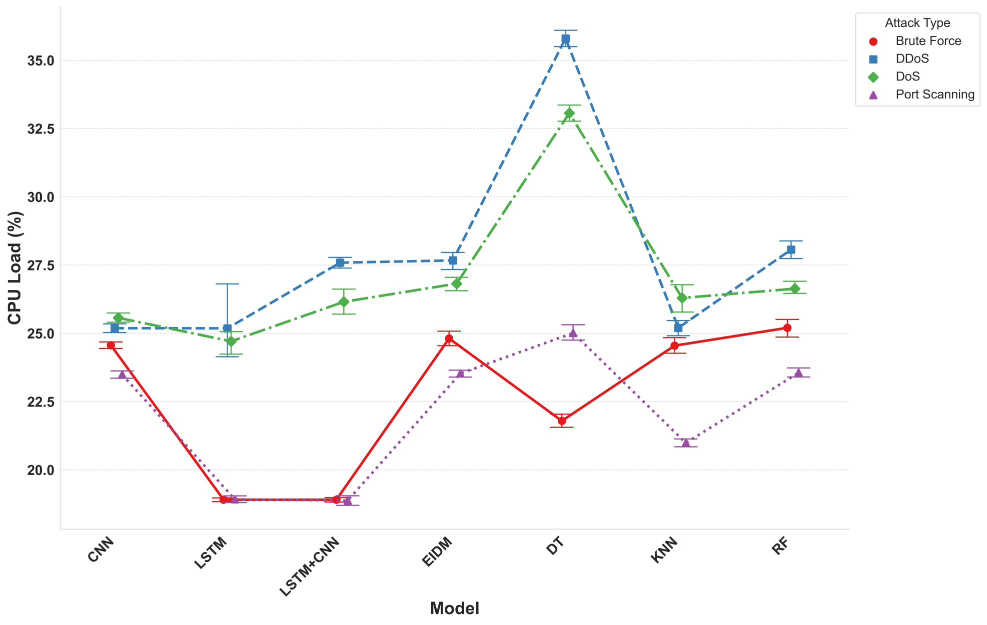

Figure 4.5 The Average CPU load of ML-based IDS under cyber threats.

efficiently but not as well as LSTM. The LSTM and CNN model hybrid, similar to EIDM, offers balanced performance, making it suitable for scenarios where moderate CPU efficiency is required at the edge.

Statistical Findings:

We conducted an ANOVA for the case of the DDoS attack, and the results are presented

Table 4.7 ANOVA results: CPU load for ML-based IDS in SDN under DDoS.

Source	**Degrees of Freedom**	**Sum of Squares**	**Mean Square**	**F Statistic**	**P-value**
Between groups	6	1184.21	197.36	142.57	< 0.05
In groups	91	125.97	1.38		
Total	97	1310.18	13.50		

in Table 4.7. The results reveal significant differences in CPU load among diverse ML-based IDS under DDoS attack, underscored by an impressive F-statistic of 142.57 and a p-value of < 0.05. This F-statistic highlights the variance in CPU load across IDSs compared to the variance in them, indicating a significant impact of IDS selection on CPU load. In addition, consistent p-values (< 0.05) were observed across other cyber threats, including brute force, DoS, and the port scan, and we do not report the result. This reinforces the presence of marked differences in CPU load among diverse ML-based IDS when subjected to different cyber threats.

Finding

The findings demonstrate that traditional ML-based IDS, e.g., DT, exhibit elevated loads under DDoS and DoS. In contrast, DL-based IDSs, including EIDM, LSTM, CNN, and their hybrids, demonstrate superior energy efficiency, making them suitable for SDN-enabled at the edge gateway. The integration of SDN helps balance network resource allocation. Yet, traditional ML-based IDS still exhibit higher CPU load due to their design, reinforcing the efficiency advantage of DL-based models in dynamic network environments.

CPU Usage:

Figure 4.6 shows that CPU usage across various ML-based IDS models in an SDN-enabled edge gateway is fairly consistent across different attack scenarios. Only minor variations are observed, as CNN, LSTM, and hybrid versions demonstrate relatively lower CPU usage, indicating efficient resource management. The DT, KNN, and RF models also show consistent CPU usage across attacks. The EIDM model balances efficiency and performance well.

Statistical Findings:

We conducted an ANOVA for the results we got for ML-based IDS in SDN under the DDoS

Table 4.8 ANOVA results: CPU usage for ML-based IDS in SDN under DDoS.

Source	Degrees of Freedom	Sum of Squares	Mean Square	F Statistic	P-value
Between groups	6	27.97	4.66	5.94	< 0.05
In groups	91	71.32	0.78		
Total	97	99.30	1.02		

attack. The results presented in Table 4.8 reveal significant differences in CPU usage among diverse ML-based IDS under DDoS attack, underscored by an impressive F-statistic of 5.94 and a p-value of < 0.05. This F-statistic highlights the variance in CPU usage across the group of IDSs compared to the variance in, indicating a significant impact of IDS selection on CPU usage. In addition, we observed a consistently low p-value (< 0.05) for other examined cyber threats (not reported in the chapter), including brute force, DoS, and port scan, reinforcing the presence of marked differences in CPU usage among diverse ML-based IDS when subjected to different cyber threats.

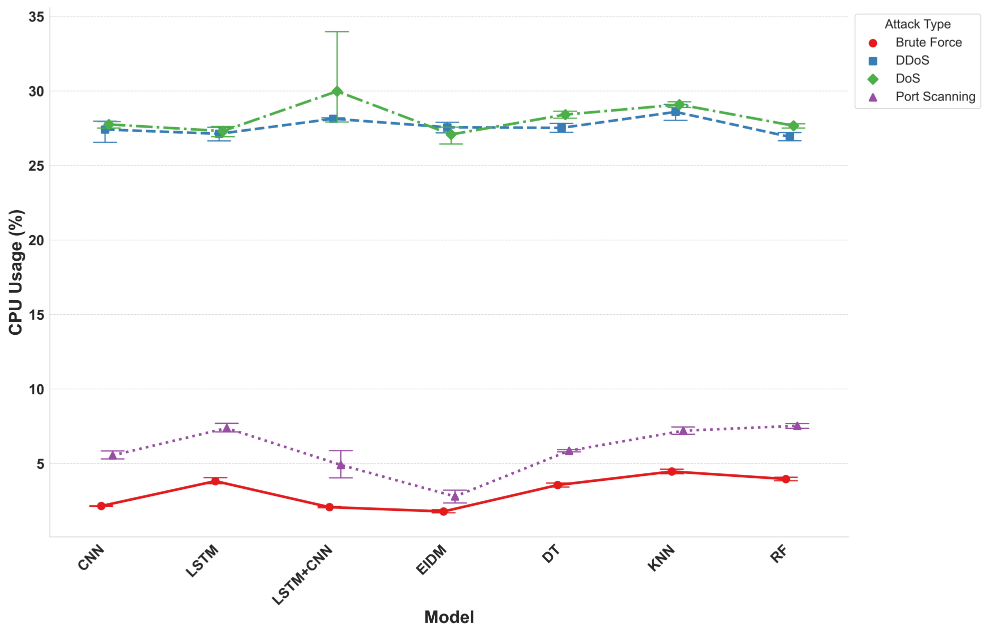

Figure 4.6 The Average CPU usage of ML-based IDS under cyber threats.

Finding

In the context of SDN-enhanced IoT, deploying DL-based IDS with advanced models such as CNN, LSTM, EIDM, and their hybrids demonstrates efficient energy consumption. These models achieve reduced CPU usage against brute force and port scan, benefiting from the centralized resource optimization afforded by SDN. Nonetheless, the complexity of DDoS and DoS presents a significant challenge, necessitating increased computational resources. Although SDN optimizes network operations, IDS models such as KNN and RF remain resource-intensive due to their frequent computational overhead. At the same time, DL-based IDS maintains efficiency through batch processing and learned representations.

Energy consumption:

Figure 4.7 depicts the average energy consumption of ML-based IDS models under different attacks in an SDN environment. The results indicate that traditional ML models consume more energy, especially during port scans, e.g., DT, KNN, and RF. In contrast, the EIDM model consistently shows lower energy consumption across all attack types, highlighting its efficiency. The LSTM and CNN models, including their hybrid version, display moderate energy consumption. Compared to non-SDN environments, the increased energy consumption in the SDN setup is attributed to the SDN controller's active role in traffic management and threat response, which demands more energy resources.

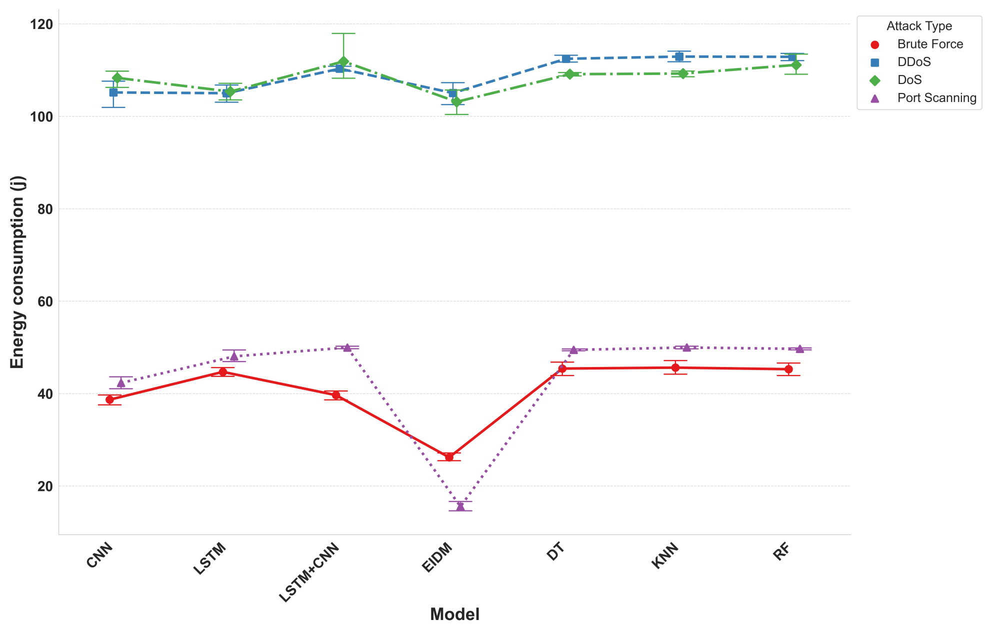

Figure 4.7 The Average Energy consumption of ML-based IDS under cyber threats.

Statistical Findings:

We applied ANOVA on energy consumption data across ML-based IDSs in SDN under DDoS. The results, presented in Table 4.9, reveal significant differences in energy consumption among diverse ML-based IDS under DDoS, underscored by an impressive F-statistic of 18.27 and a p-value of < 0.05. This F-statistic highlights the variance in energy consumption across a group of IDSs compared to the variance in, indicating a significant impact of IDS selection on energy consumption. Moreover, a consistently low p-value (< 0.05) was observed across other cyber threats, including brute force, DoS, and port scan, so we do not report the results here. This highlights marked differences in CPU usage among diverse ML-based IDS when subjected to examined cyber threats.

Table 4.9 ANOVA results: Energy consumption for ML-based IDS in SDN under DDoS.

Source	Degrees of Freedom	Sum of Squares	Mean Square	F Statistic	P-value
Between groups	6	1263.26	210.54	18.27	< 0.05
In groups	91	1048.21	11.51		
Total	97	2311.48	23.82		

Finding

The findings accentuate the distinct energy efficiency profiles of ML-based IDSs when exposed to various cyber threat scenarios. During brute force and the port scan, traditional ML-based IDS, such as DT, KNN, and RF, are observed to have higher energy consumption. This indicates that these models are not energy-efficient under the examined conditions due to their complex computational frameworks. On the other hand, DL-based IDS and the EIDM show markedly superior energy efficiency. The reduced energy footprint of DL-based IDS is especially advantageous in the context of the SDN-enabled at the edge, where low energy consumption is crucial due to device constraints and the need for long-term, autonomous operation. The reduction in energy consumption observed in DL-based IDS when integrated with SDN highlights the benefits of centralized network control and optimized workload distribution, making them a more sustainable choice for IoT security.

4.5.3 Analyzing the Impact of SDN on CPU Usage, Load, and Energy Efficiency in ML-Based IDS

Figure 4.8 demonstrates that integrating SDN with ML-based IDS in the edge gateway significantly improves resource efficiency, reducing energy consumption, CPU usage, and CPU load. The most substantial improvement is in CPU usage, where DL-based IDS, e.g., LSTM and CNN, outperform traditional ML models by efficiently handling complex computations through parallel processing. Additionally, SDN integration reduces CPU load by balancing workloads, essential for real-time threat detection in edge gateways. The observed reduction in energy consumption further highlights the approach's suitability for battery-powered edge gateway, confirming its scalability and practicality for real-world applications.

4.6 ML-Based IDS vs. Signature-Based IDS (Snort)

This section compares our ML-based IDS models and the signature-based Snort IDS to evaluate the performance improvements achieved by leveraging ML-based IDS over traditional

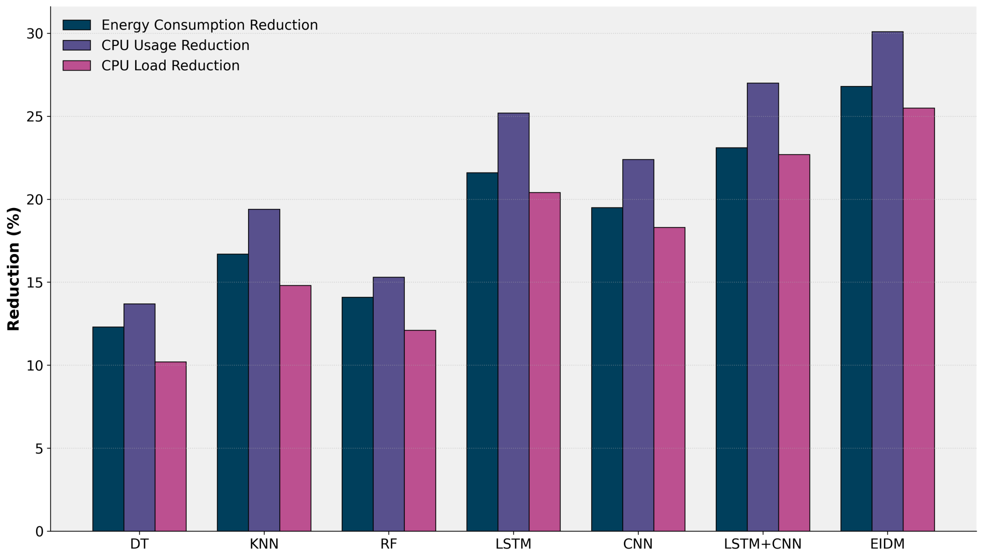

Figure 4.8 Reduction in energy consumption, CPU usage, and CPU load for ML-based IDS models with SDN integration in edge gateway.

detection systems. This comparison is essential to highlight the advantages of ML-based approaches regarding resource efficiency, scalability, and adaptability, especially in the edge gateway.

The results presented in Table 4.10 provide a comparative analysis of our ML-based IDS models against the signature-based Snort IDS discussed in other research.

Regarding CPU usage, Snort IDS shows high utilization under heavy traffic due to its reliance on predefined rules and signature matching. In contrast, the ML-based IDS models demonstrate better CPU efficiency. While traditional ML models, e.g., DT and KNN, have higher CPU usage because of iterative computations, DL-based IDS, e.g., LSTM, CNN, and a hybrid of LSTM and CNN, EIDM exhibits lower CPU usage. This is primarily due to the DL-based IDS's ability to process data in batches and leverage parallel processing for real-time threat detection. For energy consumption, Table 4.10 shows that Snort IDS consumes more energy, especially in IoT networks requiring multiple containers. However, our ML-based IDS models, especially DL architectures, e.g., LSTM and EIDM, demonstrate superior energy efficiency. These models optimize resource usage and process data efficiently, making them suitable for resource-constrained edge gateway and highlighting their scalability advantages. Finally, in terms of CPU load, Table 4.10 indicates that earlier versions of Snort IDS suffer from high CPU load on a single core because of their single-threaded architecture. Although newer versions introduce multi-threading, they still encounter processing bottlenecks under heavy traffic. Conversely, the ML-based IDS models distribute the CPU load more effectively across multiple cores. DL-based IDS, especially LSTM and hybrid architectures, achieve the lowest CPU load levels due to their parallel execution capabilities and efficient handling of sequential data.

Table 4.10 Comparative Resource Utilization of ML-Based IDS and Snort IDS Based

Metric	**Snort IDS**	**ML-Based IDS (Our Findings)**
CPU Usage	- **High Traffic Conditions:** CPU usage can reach its maximum during initialization with many active rules [199]. - **Multi-Core Systems:** Snort 3.0 utilizes a significant portion of CPU resources on a multi-core processor [200] [201].	- **Traditional ML Models (DT, KNN, RF):** Tend to exhibit higher CPU usage during real-time cyber threats, especially those requiring intensive computations. - **DL-Based Models (CNN, LSTM, Hybrid of LSTM and CNN and EIDM):** Show lower CPU usage compared to traditional ML models, with LSTM models demonstrating the most efficient utilization due to sequential data processing and parallelization.
Energy Consumption	- **IoT Deployment:** Deployment of Snort on IoT gateways results in considerable energy consumption [202].	- **Traditional ML-based IDS:** Generally consume more energy during inference cycles due to repetitive computations. - **DL-Based Models:** Exhibit better energy efficiency, especially models that combine convolutional and sequential layers, benefiting from optimized processing structures.
CPU Load	- **Single-Core Utilization:** Older Snort versions (pre-3.0) lead to high load on a single core under heavy traffic [203]. - **Multi-Core Systems:** Updated versions distribute the load but still face processing bottlenecks under extensive traffic [203].	- **Traditional ML-based IDS:** Often show higher CPU load during complex attack scenarios. - **DL-Based Models:** Maintain a lower CPU load, benefiting from parallel processing capabilities, with hybrid models showing the most balanced load distribution.

4.7 Discussion

Our investigations explored the performance metrics of ML-based IDS with various models, especially in IoT-edge devices with and without SDN integration. Our study primarily evaluated the impact of these models on CPU load, CPU usage, and energy consumption amidst diverse cyberattack scenarios. The empirical findings revealed significant disparities in resource utilization across different ML-based IDS, shedding light on crucial aspects of their deployment in IoT devices integrated with SDN. The KNN, DT, and RF significantly exhibited higher CPU load, CPU usage, and energy consumption, especially under specific types of cyberattacks. While these models are adept at identifying threats, their resource-intensive nature could pose challenges in the IoT context, where computational resources are often limited. This could lead to diminished performance or instability in environments with constrained resources. Specifically, KNN's higher variance in CPU load and energy consumption, as observed in Tables 4.4 and 4.5, stems from its lazy learning approach. Unlike other models, KNN does not build a generalized model during training but instead stores the entire dataset and computes distances at query time. This results in increased processing demands, leading to fluctuations in resource utilization. Such behavior makes KNN less suitable for real-time IDS applications in resource-constrained IoT networks [204] [205]. While CPU load significantly impacts energy consumption, it is not the sole factor. Memory operations, network activity, peripheral devices, and thermal management also contribute to power usage in IoT devices. High data transmission rates and active sensors can increase energy demands, while sustained CPU load may trigger additional energy consumption for cooling mechanisms. Although a strong correlation between CPU load and energy consumption is expected, these factors introduce variations across IDS models. Optimizing IDS efficiency can help balance security and resource constraints in IoT networks. Conversely, the CNN and LSTM models demonstrated greater efficiency in resource utilization. While their architectures are sophisticated and adept at processing complex data structures, they appear to optimize the computational load during inference when employed in IDS. This makes them more suitable for scenarios where resource conservation is critical. However, the complexity of these models introduces its own set of challenges, especially in terms of training and ongoing maintenance in the dynamic landscape of IoT devices integrated with SDN.

The balance between detection efficiency and resource consumption is especially critical at the edge gateway, where devices often have limited processing power and energy reserves. This balance is closely tied to several United Nations Sustainable Development Goals (SDGs), especially SDG 9 (Industry, Innovation, and Infrastructure), SDG 11 (Sustainable Cities and Communities), and SDG 13 (Climate Action). Optimizing IDS deployment in smart cities strengthens cybersecurity infrastructure, directly supporting SDG 9 while fostering resilient, sustainable urban environments in line with SDG 11. Furthermore, by prioritizing energy-efficient IDS solutions, this research contributes to SDG 13, promoting responsible resource consumption and mitigating the environmental impact of growing IoT networks [44].

To aid IoT developers in selecting appropriate IDS solutions, we provide detailed guidelines in Table 4.11 and Table 4.12, outlining the performance trade-offs of seven different ML-based IDS models for IoT devices examined in this chapter, both with and without SDN integration. These insights enable developers to make informed decisions, ensuring the optimal balance between security and resource efficiency during application development. We use graphical indicators (smiley faces) instead of numerical values to provide an intuitive, high-level comparison of IDS performance. This visual approach simplifies decision-making for IoT developers, aligning with similar methodologies used in prior work [206]. Moreover, all corresponding numerical values related to CPU usage, CPU load, and energy consumption are presented in the Figures and Tables in Section 5.

On the other hand, to the best of our knowledge, only Tekin et al. [168] have explored a similar direction in evaluating the performance of ML-based IDS in IoT systems. However, our study takes a fundamentally different approach, especially in how computational resources are classified and utilized, which plays a critical role in the effectiveness and scalability of IoT systems. While Tekin et al. focus on energy consumption and inference times using Raspberry Pi as an IoT device, our study emphasizes the advantages of processing data at the edge, especially regarding energy efficiency, CPU load, and usage. We show how models such as DT and RF benefit from edge processing, reducing latency, and improving responsiveness, especially when combined with SDN, which optimizes network traffic and resource allocation. Our findings underscore the importance of balancing computational tasks across the network using SDN to maintain performance, unlike Tekin et al. [168], who do not explore the impact of edge computing or SDN integration.

4.8 Threat and validity

Empirical research inevitably encounters issues related to the validity of findings. In light of this, the present section seeks to identify and discuss possible threats to our research's validity, per the recommendations of Wohlin et al. [207].

4.8.1 Internal Threats

During our empirical study on ML-based IDS in the context of IoT devices with IoT devices integrated with SDN, we recognized the existence of internal obstacles that impact the credibility of our findings. The precision of our performance measures is of utmost importance, namely the measurement of CPU load, CPU usage, and energy consumption in these intricate network settings. The complex characteristics of IoT devices and the adaptable structure of SDN provide significant difficulties in guaranteeing accurate and dependable performance evaluations. To address these concerns, we performed fifteen experiments on our testbeds. To improve the trustworthiness of our results in the context of SDN and IoT, we utilized average values to reduce the impact of network or hardware differences and ambient factors. In addition, the cyber threat simulations were conducted using highly practiced cybersecurity testing mechanisms in academic research and industries in IoT-edge devices integrated with SDN. This work aims to tackle internal risks associated with the setup and precision of ML-based IDS, improving their usefulness and significance in these fast-advancing technical fields.

Table 4.11 Guideline for selecting seven ML-based IDS in edge gateway.

Metric	**DT**	**KNN**	**RF**	**CNN**	**LSTM**	**CNN**	**LSTM+CNN**	**EIDM**
CPU load	☹️	☹️	😐	😊	😊	😊	😊	😊
CPU usage	☹️	☹️	☹️	😊	😊	😊	😊	😊
Energy consumption	😐	☹️	☹️	😊	😊	😊	😊	😊

Table 4.12 Guideline for selecting seven ML-based IDS in SDN-edge gateway.

Metric	**DT**	**KNN**	**RF**	**CNN**	**LSTM**	**CNN**	**LSTM+CNN**	**EIDM**
CPU load	☹️	☹️	☹️	😊	😊	😊	😊	😊
CPU usage	😐	😐	😐	😐	😐	😐	😐	😐
Energy consumption	😐	😐	😐	😐	😐	😐	😐	😐

The energy consumption and CPU usage in all ML-based IDS lowered during the brute force attack and port scan.

4.8.2 External Threats:

The landscape of network security, especially in IoT-edge devices and IoT-edge devices integrated with SDN realms, is increasingly challenged by external threats. These range from sophisticated cyberattacks such as DoS, DDoS, and brute force attacks to more subtle, yet equally harmful, reconnaissance methods such as a port scan. These threats highlight the urgent need for robust and adaptable IDS solutions. Integrating ML into IDS presents promising advancements in threat detection and mitigation. However, this integration faces challenges due to the complexity of IoT-edge devices, which are marked by numerous interconnected devices, and the dynamic nature of SDN architectures. IDS solutions must be precise in threat detection while also being resource-efficient. Our research evaluates ML-based IDS based on CPU usage, CPU load, and energy consumption, especially under real-time cyber threats. These metrics are vital to ensure that ML-based IDS are effective in protecting networks against external threats and sustainable in their operation. They help maintain a crucial balance between security and performance in the complex ecosystems of IoT devices and IoT devices integrated with SDN. Additionally, to ensure the transparency and reproducibility of our study, we have provided detailed information about the experimental setup and made our testbed and results publicly available for further research [208]. By adopting these measures, we have attempted to provide robust validation and increase the inability to reject our findings among practitioners and researchers.

4.9 Chapter Summary

This chapter presents a comparative analysis of the ML-based IDS in IoT-edge devices and IoT-edge devices integrated with SDN under different cyberattack scenarios, resulting in comprehension. In IoT systems, conventional ML models (e.g., KNN and DT) often experience increased CPU load and CPU usage, especially when subjected to DoS and DDoS cyber threats. This suggests that these models have limits in resource-limited situations. In contrast, DL-based IDS (e.g., CNN and LSTM) exhibit reduced CPU usage, indicating improved efficiency and compatibility with IoT security. A consistent energy consumption pattern was identified across attack types in both scenarios, encompassing advanced neural networks and conventional methods. The consistent energy efficiency of these models, independent of their computing complexity, highlights their efficacy and long-term viability for use in different network environments. The findings emphasize the significance of choosing ML-based IDS according to their computational efficiency and energy consumption to achieve optimal performance in networks with limited resources. It is imperative to thoroughly evaluate the scalability and robustness of ML-based IDS in future research, especially in more significant and complex network environments. This assessment will explain their ability to adjust to changing cyber threats. Furthermore, it is crucial to evaluate the influence of new technologies, e.g., 5G and edge computing, on the efficacy and suitability of ML-based IDS in advanced network infrastructures.

CHAPTER 5 ARTICLE 3: UNDERSTANDING THE IMPACT OF IOT SECURITY PATTERNS ON CPU USAGE AND ENERGY CONSUMPTION: A DYNAMIC APPROACH FOR SELECTING PATTERNS WITH DEEP REINFORCEMENT LEARNING

5.1 Article Metadata

- **Title:** Understanding the Impact of IoT Security Patterns on CPU Usage and Energy Consumption: A Dynamic Approach for Selecting Patterns with Deep Reinforcement Learning
- **Authors:** Saeid Jamshidi, Amin Nikanjam, Kawser Wazed Nafi, Foutse Khomh
- **Affiliation:** SWAT, Polytechnique Montréal, H3T 1J4, Quebec, Canada
- **Submitted Date:** 10 February 2025
- **Accepted Date:** 23 February 2025
- **DOI:** `https://doi.org/10.1007/s10207-025-01011-5`
- **Journal:** *International Journal of Information Security*

5.2 Chapter Overview

IoT introduces numerous security challenges that require effective solutions. IoT security patterns provide a practical approach to address recurring security issues, yet their impact on edge gateway metrics (e.g., energy consumption, CPU usage, and load) remains largely unexplored. This study empirically evaluates six IoT security patterns (PZH, TCP, OOC, BL, WL, and SSN) in three IoT-edge applications: smart home, smart city, and healthcare. We evaluated the patterns individually and in combination, subjecting them to cyber threats. Subsequently, we analyzed their impact on energy consumption, CPU usage, and load.
We propose a DRL-based IDS that dynamically selects security patterns based on real-time conditions to address observed resource trade-offs. Tested against threats (for example, DoS Hulk, Slowloris, DDoS, and GoldenEye), this adaptive approach optimizes security and resource efficiency, selecting the most suitable patterns for each scenario. The findings show that pattern selection significantly impacts resource metrics and that the DRL-based system maintains robust security while minimizing energy and CPU overheads. Based on these

findings, we provide guidelines for developers to improve IoT-edge security by optimizing resource consumption.

5.3 Study Design

IoT is a network of interconnected physical objects embedded with hardware and software components. IoT has found wide-ranging applications in various domains, including commerce, healthcare, industry, and transportation [209]. IoT devices facilitate data collection, analysis, and communication among connected devices, enhancing automation and efficiency. To support these processes, edge devices such as sensors, gateways, routers, and edge servers are integral in collecting and processing data in IoT applications. Communication among these devices relies on wired and wireless networks in IoT [210], which are inherently vulnerable to various attacks. The interdependence of components within IoT architectures, as depicted in Fig. 5.1, emphasizes the susceptibility of the entire system to a single security breach. Therefore, ensuring the safety and reliability of IoT-based applications has become essential for their seamless operation in critical domains.

Researchers have proposed various security patterns to encapsulate solutions to recurring challenges in IoT security. These patterns consider the secure design of IoT-based applications, accounting for their heterogeneity and complexity [93]. Examples include the PZH, TCP, OOC, BL, WL, and SSN [1], and Secure Sensor Node patterns [94]. Despite their availability, to the best of our knowledge, no prior study has evaluated the energy consumption and CPU usage of IoT devices configured with different security patterns. Consequently, determining a trade-off between security effectiveness and resource constraints—such as energy consumption and CPU usage—remains a pressing challenge.

However, IoT comprises low-powered devices (e.g., Raspberry Pi) with limited computational and memory resources. Implementing security mechanisms typically demands additional processing, communication, and computational resources, which can impact the performance of IoT devices and networks [211]. Despite the performance overhead introduced by security mechanisms, they remain indispensable for protecting IoT networks against cyber threats. Striking a balance between achieving a robust level of security and accommodating the inherent physical limitations of IoT devices, such as low computing power and restricted memory capacity in battery-powered devices, is crucial.

On the other hand, while effective in mitigating some threats, existing static security mechanisms often lead to increased energy consumption and CPU load, degrading overall system performance [212]. Additionally, these static configurations cannot adapt to IoT networks' dynamic and evolving nature, reducing their effectiveness against real-time threats.

To address these limitations, this chapter empirically investigates the impact of six IoT security patterns on CPU usage, energy consumption, and security resilience in three IoT applications: smart home, smart city, and healthcare. Our findings show that these patterns improve security and introduce computational overhead, affecting performance. To mitigate this, we propose a DRL-based IDS that dynamically selects security patterns based on real-time threats, optimizing resource efficiency while maintaining strong security defenses. To summarize, this study makes the following contributions:

- **Evaluation of IoT Security Patterns:** We evaluate six IoT security patterns (PZH, TCP, OOC, BL, WL, and SSN) against critical attacks at the edge gateways scenarios, such as DDoS, MITM, and brute-force.
- **Analysis of Resource Trade-offs:** For each pattern, we analyze trade-offs between energy consumption and CPU usage.
- **A novel DRL-based IDS:** We propose a DRL-based IDS that dynamically selects and adapts security patterns, optimizing resource efficiency and ensuring real-time threat mitigation.

5.4 Evaluation Framework for IoT Security Patterns

This section outlines our methodology for evaluating the impact of specific IoT security patterns at the edge gateway. We first define our RQs and subsequently provide details on the experimental design and the metrics used to assess the impact of the patterns.

5.4.1 Research Questions (RQs):

Our research aims to address the following RQs:

- **RQ1: To what extent does utilizing IoT security patterns improve the security at the edge gateway?**
  We analyze both in-pair and combined patterns (i.e., all patterns) to assess the security at the edge gateway. The penetration test evaluates the security, which involves launching simulated attacks against the edge. By adopting this approach, we aim to evaluate the effectiveness of the security patterns and identify any vulnerabilities that need to be addressed to enhance the security at the edge for implementing applications.

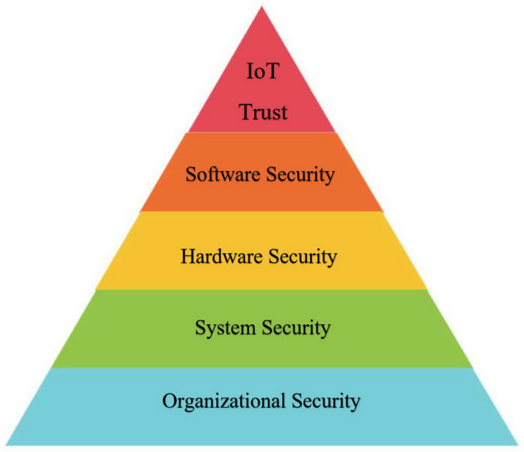

Figure 5.1 The figure illustrates the block diagram of the IoT pyramid.

- **RQ2: How effective is the DRL-based IDS in dynamically selecting security patterns for real-time threat mitigation?** We assess the DRL-based IDS's ability to adaptively choose security patterns based on evolving threat conditions and resource constraints. This involves evaluating its responsiveness, resource efficiency, and accuracy in detecting and mitigating specific cyber threats, such as DoS Hulk, DoS Slowloris, DDoS, and DoS GoldenEye attacks, compared to static pattern configurations.

- **RQ3: How does using security patterns at the edge gateway impact energy consumption, CPU usage, and CPU load?** We conduct an empirical study at three edge gateways (IoT application) to investigate the impact of in-pair and combined patterns on energy consumption, CPU usage, and CPU load. We measure these metrics with/without patterns and under attacks.

5.4.2 Experimental Design

In this subsection, we present the experimental setup of our study. We empirically investigated three IoT-edge-based applications to assess six IoT security patterns: PZH, TCP, OOC, BL, WL, and SSN. To address RQ1, we examined these patterns using multiple communication frameworks and IoT devices. A testbed, consisting of three Raspberry Pis (edge gateways) and various sensors, was created. Each device was configured to establish an IoT-edge-based application setup alongside its respective sensor nodes. Our methodology involved implementing the six distinct security patterns across the three applications and conducting penetration tests using Kali Linux to evaluate their effectiveness in enhancing security [1, 93, 94].

For RQ3, we measure energy consumption, CPU usage, and CPU load with/without patterns under the attacks simulated by the penetration tests. We created three IoT hubs with the sensors. To work with each sensor's data in each IoT hub, we developed individual applications and implemented related security patterns.
In this chapter, we targeted three IoT-edge-based applications:

- **Smart Home:** The first experimental setup focuses on a smart home automation system using a Raspberry Pi as the edge gateway [213]. The system includes a motion sensor, an AC supply, a DHT22 sensor, an MQ-9 gas sensor module, and an MCP3008 converter. The edge gateway processes data from the sensors to control internet-connected home devices. It measures temperature and humidity, manages lighting and gas sensors, and detects intruders. When motion is detected, the edge processes the data and notifies the user.

- **Smart City:** The second experimental setup involves the deployment of a smart city infrastructure [214] for real-time pollution monitoring. In this setup, the Raspberry Pi acts as the edge gateway, processing data from various sensors at the local level. The system integrates sensors such as the DHT22, MQ-9, MQ-135, a pressure air sensor, and an MCP3008 converter to monitor air quality. The edge gateway processes data on temperature, atmospheric pressure, humidity, carbon dioxide, carbon monoxide, methane, and ammonium levels.

- **Healthcare:** The third experimental setup is the healthcare application [215, 216] that incorporates ECG and heartbeat monitoring. The system allows remote patient monitoring using an AD8382 ECG sensor and an Arduino ESP8266 Wi-Fi module. The edge gateway processes data from the ECG and heartbeat sensors, and vital information is stored for further analysis. Continuous monitoring provides real-time data to doctors, nurses, or relatives, allowing them to assess the patient's condition remotely.

Fig. 5.2 illustrates the topology of our configured testbed. We coded all the data collection and interaction applications in Python and C++. We developed the server side using Flask [217] and Nginx [218], a lightweight web server. We stored sensor data in an SQLite database and used the Google Chart API [219] to create visual representations. In contrast, Plotly [220] handled the graphical analysis of the sensor data. The application sends the data to the edge gateway, where only the authorized server maintainer controls it, and users can view or manipulate it as needed. Additionally, we performed experiments on three Raspberry Pi 4 Model B devices to test security patterns under real-time cyber threats, including

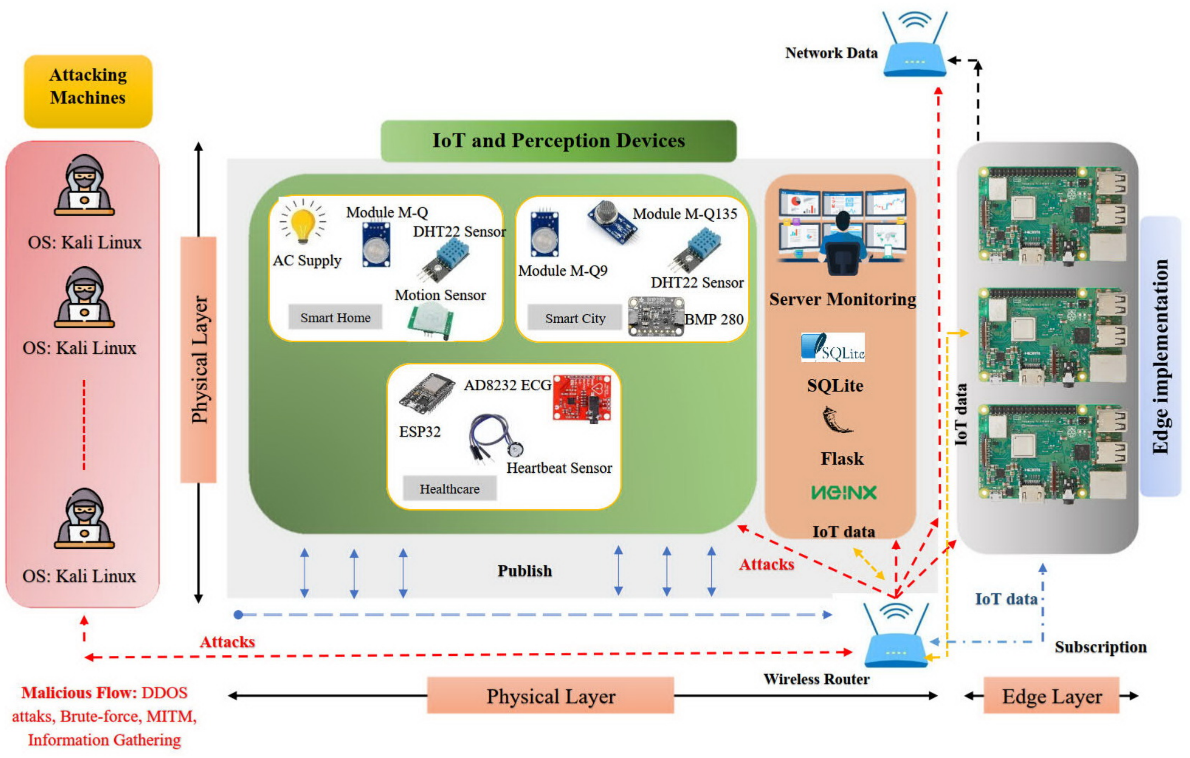

Figure 5.2 Testbed topology.

Table 5.1 List of equipment for the testbed.

TestBed	Name	Description
Smart Home	Edge device	Raspberry Pi
	Node 1	DHT22 sensor
	Node 2	Module M-Q9
	Node 3	AC supply
	Node 4	MCP3008 converter
	Node 5	Motion sensor
	User	Connecting using Any device
	Router	Wireless Hub/ Connection Point
	Hacker	Intruder/ Cyber Attacker
Smart City	Edge device	Raspberry Pi
	Node 1	DHT22 sensor
	Node 2	Module M-Q9
	Node 3	BMP 280 (Air Pressure)
	Node 4	MCP3008 converter
	Node 5	Module MQ135 Sensor
	User	Connecting using Any device
	Router	Wireless Hub/ Connection Point
	Hacker	Intruder/ Cyber Attacker
Health Care	Edge device	Raspberry Pi
	Node 1	ESP32
	Node 2	AD8232 ECG Sensor
	Node 3	Heartbeat Sensor
	Node 4	DHT22 Sensor
	Node 5	Microcontroller ESP32 Module
	User	Connecting using Any device
	Router	Wireless Hub/ Connection Point
	Hacker	Intruder/ Cyber Attacker

DDoS, MITM, and brute force at the edge gateway. Each Raspberry Pi featured a quad-core

Cortex-A72 CPU running at 1.5 GHz, 4 GB of LPDDR4 RAM, and a 512GB microSD card for storage, powered by a 5000mAh power bank. Moreover, we measured CPU usage, CPU load, and energy consumption both with/without patterns and under cyber threats.
Also, we conducted experiments with the DRL-based IDS on a Raspberry Pi 4 Model B to dynamically manage security patterns under real-time DDoS and DoS (DoS GoldenEye, DoS Hulk and DoS Slowloris) cyber threats. The Raspberry Pi featured a quad-core Cortex-A72 CPU running at 1.5 GHz, 4 GB of LPDDR4 RAM, and a 512GB microSD card for storage, powered by a 5000mAh power bank to replicate typical IoT conditions. In addition, we measured CPU usage, memory usage, and energy consumption on this constrained edge gateway for adaptive pattern selection. The tests include regular operation and simulated attacks, with performance evaluated based on energy efficiency, resource utilization, and attack detection.
Additionally, the experimental testbed initially consisted of two Raspberry Pi devices inspired by the setup described in [110]. To further evaluate scalability, we expanded the testbed to include additional Raspberry Pi devices configured to simulate diverse roles and communication protocols. This setup allowed us to test the patterns across edge gateway scales, from small-scale environments, e.g., smart home, to medium-scale setups, e.g., smart city, and larger-scale deployments, e.g., interconnected healthcare networks. By incorporating these scenarios, the testbed evaluated the patterns' adaptability and effectiveness in diverse and increasingly complex edge gateways.

5.4.3 Metrics

In this research, we focused on evaluating energy consumption, CPU usage, and CPU load as critical metrics for assessing the feasibility of IoT security implementations in resource-constrained edge gateways (Raspberry Pi devices) [202] [221] [222]. These metrics provide a direct reflection of the trade-offs between security and resource utilization, aligning with the primary objective of our study. Metrics, such as network latency, data throughput, or user experience, were not included because external factors influence them, e.g., network architecture, application-layer protocols, and end-user interactions. By excluding these, we ensure our analysis focuses on resource utilization metrics, providing a clear and in-depth understanding of the trade-offs in securing edge gateways. Monitoring CPU load, CPU usage, and energy consumption is crucial at the edge gateway, which has limited processing power and cooling capabilities. High CPU load can degrade performance, increase energy consumption, and generate excess heat, impacting system stability and energy efficiency. These metrics (energy consumption, CPU usage, and CPU load) [40] allow us to directly evaluate the practicality of IoT security patterns in real-world edge scenarios, where maintaining resource efficiency is as essential as ensuring robust security. On the security side, monitoring CPU usage can help detect threats (e.g., DDoS) attacks by identifying unusual traffic or processing spikes. We measure energy consumption and CPU usage using PowerTOP [192], which precisely monitors both metrics.

Energy Consumption and CPU Usage Evaluation

As we mentioned, we measured energy consumption using *PowerTOP* on a Raspberry Pi. *PowerTOP* provides power usage data over specified intervals. The total energy (E) was calculated using the formula:

$$E = \sum_{i=1}^{n} P_i \cdot \Delta t_i \tag{5.1}$$

Where P_i is the power usage during interval i, Δt_i is the duration of interval i, and n is the total number of intervals. This calculation allowed us to assess the energy impact of different security patterns and attack scenarios on the Raspberry Pi. Moreover, we measured CPU usage (%) on the Raspberry Pi using metrics from the `/proc/stat` file. CPU usage was calculated as follows:

$$\text{CPU Usage (\%)} = \frac{\text{Active CPU Time}}{\text{Total CPU Time}} \times 100 \tag{5.2}$$

Active CPU Time includes user and system times, while Total CPU Time comprises Active and Idle times. This approach provided a detailed understanding of the computational overhead that introduced security patterns and attack conditions.

5.4.4 Analysis method for energy consumption and CPU usage

In this study, we employ the Mann-Whitney U test, also known as the Wilcoxon rank-sum test, as it was used in similar studies Khomh and Abtahizadeh [206]. It is used to test whether there is a significant difference in the distribution of a continuous variable between two independent groups. Since the findings indicate differences between several independent groups, we use the Mann-Whitney U test [223] to examine $H_x^1, H_{1,3}^2, H_x^3, H_{1,3}^4$, etc. Moreover, we used Cliff's δ effect size to determine the importance of the differences between metric values. Cliff's δ is a non-parametric effect-size measure that indicates the extent to which two sampling distributions overlap. All of our tests are performed at a 95% confidence level (i.e., the p-value is less than 0.05). Because we conduct multiple tests of the null hypothesis, we use a Bonferroni correction [224] to deal with the issue of multiple comparisons, which involves dividing the threshold p-value by the number of tests.

Table 5.2 Patterns selected for experimentation.

Pattern	**Abbreviation**	**Code**
Personal Zone Hub	PZH	P_1
Trusted Communication Partner	TCP	P_2
Outbound-Only Connection	OOC	P_3
Blacklist	BL	P_4
Whitelist	WL	P_5
Secure Sensors	SSN	P_6
Without Pattern	-	P_0

5.4.5 Studied Attacks

In this section, we analyze the attacks we select to launch in the testbeds to assess the efficiency and resilience of the security patterns employed. The selection of cyber threats, including DDoS [110], brute-force [225] [195] [226], and MITM [227] attacks, was guided by their prominence in the literature and relevance to IoT scenarios [228] [229]. DDoS attacks are a significant concern in smart cities due to their highly interconnected systems, where service disruptions can have widespread consequences. Brute-force and MITM attacks are critical for protecting sensitive patient data in healthcare settings.
The penetration testing simulated realistic conditions to ensure the practical relevance of the selected threats. While no selection can encompass all IoT risks [230] [231], these attacks provide essential benchmarks for evaluating the efficiency and resilience of the security patterns across smart homes, smart cities, and healthcare systems [232] [233] [234].
- **DDoS:** attacks exploit vulnerable edge gateways with limited processing power and security. In distributed environments such as smart cities, attackers can flood the gateway with traffic, overwhelming it and potentially disrupting the entire network [235] [11] [10] [236].
- **SSH-MITM:** attacks allow attackers to intercept and manipulate communication between the edge gateway and the cloud, especially over wireless networks. This poses serious risks, particularly in critical infrastructure systems, as attackers gain unauthorized access to sensitive data [227] [237].
- **Brute-Force:** attacks target edge gateways with weak security by repeatedly attempting to guess credentials. Once an attacker gains access, they can use the compromised device as an entry point to the broader network, potentially exposing sensitive data or compromising critical infrastructure [238] [239].

5.5 Implementation of patterns

To address RQ1, we implemented various security patterns, including OOC, PZH, WL, BL, TCP, and SSN. Each pattern serves a distinct purpose in enhancing the security of edge

Algorithm 1: OOC Pattern Implementation with EdgeGuard

```
Input: Trusted Servers ($S_{trusted}$), Allowed Ports ($P_{allowed}$)
Output: Outbound traffic is allowed, inbound traffic is restricted
Initialization:
Set $S_{trusted} \leftarrow \{IP_1, IP_2, \ldots, IP_n\}$ // Initial list of trusted server IPs
Set $P_{allowed} \leftarrow \{Port_1, Port_2, \ldots, Port_m\}$ // Initial list of allowed ports for outbound traffic
Step 1: Monitor outgoing connections ;
foreach Device in Network do
    $Device_{IP} \leftarrow$ Get_IP(Device);
    $Device_{Port} \leftarrow$ Get_Port(Device);
    Step 2: Validate outgoing traffic ;
    if $Device_{IP} \in S_{trusted}$ and $Device_{Port} \in P_{allowed}$ then
        Allow Outgoing Connection;
    else
        Block Outgoing Connection;
Step 3: Block unsolicited inbound traffic ;
foreach Incoming Connection do
    if Connection not initiated by internal device then
        Block Incoming Connection;
Step 4: Dynamic Updates for $S_{trusted}$ and $P_{allowed}$ ;
while Network Active do
    Update $S_{trusted}$ using threat intelligence and network monitoring;
    Adjust $P_{allowed}$ based on application behavior;
```

gateways under different threat scenarios.

5.5.1 Implementation of OOC and PZH security patterns

We implemented the OOC and PZH patterns using our custom-designed EdgeGuard system. According to their distinct concepts, we implemented these two patterns for various security purposes.

For the OOC pattern, the "trusted servers" ($S_{trusted}$) and "allowed ports" ($P_{allowed}$) are dynamically managed through automated updates, as detailed in Algorithm 1. Trusted servers are updated when new safe servers are identified or existing ones are flagged as compromised based on real-time threat intelligence. Similarly, allowed ports are adjusted dynamically by monitoring application behavior and detecting vulnerabilities in unused or unsafe ports. These updates occur during the dynamic update phase of the algorithm, ensuring the edge gateway enforces strict outbound-only communication and effectively blocks unauthorized inbound traffic.

Similarly, the PZH pattern, described in Algorithm 2, dynamically adjusts the whitelist of recognized devices ($D_{recognized}$) and the allowed IP range (IP_{range}). During the dynamic update phase, real-time anomaly detection identifies unauthorized devices or network changes, such as adding new devices or subnets. Devices that do not match the whitelist or fall outside the allowed IP range are flagged and isolated immediately. These dynamic updates enhance the system's ability to adapt to changing network conditions, providing robust protection against unauthorized access.

5.5.2 Implementation of WL, BL, and TCP

We implement WL, BL, and TCP security patterns to enhance edge security by managing access control through user authentication and privilege management. The goal is to restrict access to authorized users and devices while blocking unauthorized ones. We evaluated these patterns through penetration tests, including brute-force attacks, to measure the system's resilience. The WL pattern allows pre-approved users and devices access, while the BL blocks unauthorized users who attempt to violate access protocols. A dynamic *access_level* attribute is introduced to control user privileges. If access rules are violated (e.g., after exceeding login attempts), users and devices will be moved from the WL to the BL. This setup helps mitigate brute-force attacks. Algorithm 3 details the logic for authenticating users, managing login attempts, and monitoring network activity for unauthorized edge.

Algorithm 2: PZH Pattern Implementation with Dynamic Updates

```
Input: Recognized Devices ($D_{recognized}$), IP Range ($IP_{range}$)
Output: Only recognized devices are allowed to connect
Initialization:
Set $IP_{range} \leftarrow$ [Defined IoT network IP range] // Initial allowed IP range
Set $D_{recognized} \leftarrow \{MAC_1, MAC_2, \ldots, MAC_n\}$ // Initial whitelist of recognized MAC addresses
Step 1: Scan the network for active devices ;
foreach Device in Network do
    $Device_{IP} \leftarrow$ Get_IP(Device);
    $Device_{MAC} \leftarrow$ Get_MAC(Device);
    Step 2: Validate IP address ;
    if $Device_{IP} \notin IP_{range}$ then
        Flag Device as Unrecognized;
        Isolate Device;
    Step 3: Validate MAC address ;
    if $Device_{MAC} \notin D_{recognized}$ then
        Flag Device as Unrecognized;
        Isolate Device;
    else
        Allow Device Access;
Step 4: Monitor network for new devices ;
while Network Active do
    Capture Network Traffic;
    foreach New Device detected do
        if New Device MAC $\notin D_{recognized}$ then
            Isolate Device;
        else
            Allow Device Access;
    Step 5: Dynamic Updates for $D_{recognized}$ and $IP_{range}$ ;
    Update $D_{recognized}$ based on network behavior and device activity;
    Adjust $IP_{range}$ to accommodate new subnets or devices;
```

5.5.3 Implementation of WL, BL and SSN security patterns

We first implemented the SSN pattern to secure data transmission over an unsecured network. The SSN pattern employs RSA-2048 and AES-256 encryption to protect sensitive data during transfer, ensuring data integrity and confidentiality. We tested the system against brute-force and SSH-MITM attacks to evaluate its effectiveness. The WL pattern grants access only to authorized users, while the BL pattern blocks unauthorized users. The WL, BL, and SSN patterns created a robust security framework that mitigated these attacks by enforcing strict access control and securing data transmission. The implementation details are presented in Algorithm 4.

5.5.4 The Proposed Approach for Dynamic Pattern Selection

To address RQ2, a dynamic approach is essential for selecting security patterns based on evolving edge conditions and resource constraints. We propose a DRL-based IDS for dynamic pattern selection, leveraging PPO [240] [241]. PPO leverages the system's state, represented as S_t, which includes metrics, i.e., specific threat indicators(DDoS, DoS GoldenEye, DoS Hulk, and DoS Slowloris). Based on this state, PPO calculates the most suitable action $a_t \in \{\mathrm{OOC}, \mathrm{PZH}\}$ to address the detected conditions:

$$\pi_\theta(a_t|S_t) = \frac{\exp(Q(a_t, S_t))}{\sum_{a' \in \{\mathrm{OOC,PZH}\}} \exp(Q(a', S_t))}, \tag{5.3}$$

Where $Q(a_t, S_t)$ estimates the effectiveness of applying action a_t in state S_t. The selected action then triggers a predefined security pattern (e.g., OOC for external threats or PZH for internal anomalies) and its associated policy (e.g., IP blocking or honeypot redirection) to mitigate the threat effectively. This approach ensures that the DRL-based IDS dynamically adapts to real-time conditions, balancing threat mitigation and resource efficiency.

We trained the model using the CICIDS2017 [242], a well-regarded benchmark in intrusion detection research. This dataset includes a range of attacks, such as **DoS Hulk**, **DoS Slowloris**, **DDoS**, and **DoS GoldenEye**, which allows our model to learn to detect diverse attack types effectively.

While we have already mentioned our test bed (Fig. 5.2), Figure 5.3 illustrates the setup of a small-scale testbed to understand how the DRL-based IDS selects patterns against threats. The testbed consists of the edge gateway (Raspberry Pi), where the DRL-based IDS is deployed and connected to a router that facilitates network traffic flow. On the right, an attacker node generates various types of malicious traffic, including **DDoS, DoS GoldenEye, DoS Hulk, and DoS Slowloris** attacks sent through the router toward the

Algorithm 3: WL, BL, and TCP Implementation

```
Input: User Credentials ($U_{credentials}$), Device MAC ($D_{MAC}$)
Output: Allow or Deny Access based on WL or BL status
Initialization:
Set $WL \leftarrow \{U_1, U_2, \ldots, U_n\}$ // List of Whitelisted Users and Devices
Set $BL \leftarrow \{U_1, U_2, \ldots, U_n\}$ // List of Blacklisted Users and Devices
Set Login Attempt Limit $N_{limit}$ // Limit number of login attempts
Step 1: User Authentication Request:
foreach Login Attempt by User do
    $U_{credentials} \leftarrow$ Get_Credentials(User)
    $D_{MAC} \leftarrow$ Get_MAC(Device)
    Step 2: Validate against Whitelist:
    if $U_{credentials} \in WL$ and $D_{MAC} \in WL$ then
        Allow Access;
    Step 3: Validate against Blacklist:
    if $U_{credentials} \in BL$ or $D_{MAC} \in BL$ then
        Deny Access;
        Exit
    Step 4: Failed Login Attempts:
    if Login attempts $> N_{limit}$ then
        Move User to Blacklist;
        Deny Access;
    Step 5: Monitor Network:
    while System Active do
        Capture Network Traffic;
        Detect Unrecognized Devices;
        if Device not in $WL$ then
            Flag and Isolate Device;
```

Algorithm 4: WL, BL, and SSN Implementation

```
Input: Credentials ($U_{credentials}$), Device MAC ($D_{MAC}$), Sensor Data ($S_{data}$)
Output: Access Control and Secure Data Transmission
Initialization:
$WL \leftarrow \{Authorized\ Users/Devices\}$, $BL \leftarrow \{Blocked\ Users/Devices\}$ ;
$SSL_{keys} \leftarrow RSA\_2048()$ // Generate keys
;
Step 1: Authentication:
if $U_{credentials}, D_{MAC} \in WL$ then
    Allow Access;
else if $U_{credentials}, D_{MAC} \in BL$ then
    Deny Access;
if Login attempts $> N_{limit}$ then
    Move to BL;
Step 2: SSN Data Transmission:
foreach $S_{data}$ do
    Encrypt with AES-256, Transmit Securely;
```

DRL-enabled edge gateway.

Pattern Selection Strategy

This section outlines the DRL-based security pattern selection based on dynamic real-time edge conditions (e.g., the type of attacks). Moreover, the proposed approach adaptively chooses between **OOC** and **PZH**, deploying targeted policies to address cyber threats efficiently.

- Dynamic Action Selection for Security Patterns:
Algorithm 5 details a DRL-based IDS to dynamically select security patterns (e.g., OOC and PZH) based on the edge gateway conditions. The process begins with the PPO agent detecting threats (DoS Hulk, DoS Slowloris, DDoS, or DoS GoldenEye) based on features extracted from the current state (S_t). The state space (S_t) is dynamically adjusted depending on the type of attack scenario [243]. For example:

- **DDoS Attacks**: S_t emphasizes features such as high packet arrival rates and traffic anomalies.
- **DoS Slowloris**: S_t prioritizes metrics related to prolonged connection durations and irregular headers.

These adjustments ensure the PPO agent's decisions are optimized for specific threat contexts. After an attack is detected, the DRL-based IDS selects a pattern (e.g., OOC for external threats or PZH for internal anomalies) and applies a corresponding policy to mitigate the threat effectively. The reward function r_t is tailored to the application or scenario, emphasizing relevant objectives [243]. For example:

- **Healthcare Applications**: Rewards prioritize high detection accuracy to ensure critical services remain unaffected.

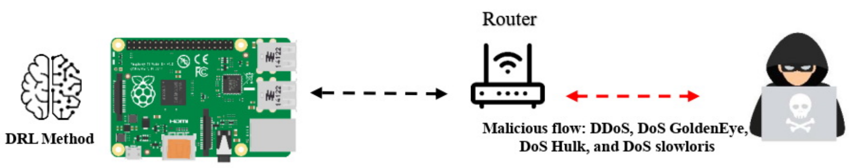
DRL Method
Router
Malicious flow: DDoS, DoS GoldenEye, DoS Hulk, and DoS slowloris

Figure 5.3 The small-scale testbed demonstrates the implementation of our DRL-based IDS model.

Positive rewards incentivize effective threat mitigation, while penalties discourage resource inefficiencies or inadequate responses. This structured approach enables the DRL-based IDS to adapt dynamically to real-time conditions and optimize pattern selection and policy application to achieve robust and efficient security.

For effective adaptive security, at each time step t, the system's state $S_t \in \mathcal{S}$ provides input to the PPO agent, which selects an action $a_t \in \{\text{OOC}, \text{PZH}\}$ based on the policy $\pi_\theta(a|S_t)$, parameterized by θ. The selection is defined as:

$$a_t = \arg \max_{a \in \{\text{OOC}, \text{PZH}\}} \pi_\theta(a|S_t) \tag{5.4}$$

Where **OOC** is selected for outbound protection, implementing restrictions on external connections for high-risk attack conditions, and **PZH** is chosen for internal control, enforcing communication limits in trusted zones against internal threats.

After selecting a pattern a_t, the system applies a corresponding policy $P_{a_t} \in \{P_1, P_2, P_3, P_4, P_5\}$, each chosen to maximize network protection and resilience against specific threats:

- **Policy** P_1: IP Blocking, paired with **OOC**, restricts access from malicious IPs in high-risk conditions (e.g., DDoS), directly reducing the attack surface and resource strain.
- **Policy** P_2: Firewall Activation, compatible with both **OOC** and **PZH**, filters malicious traffic, adding a general protective layer to safeguard against diverse threat types.
- **Policy** P_3: Honeypot Redirection, applied with **PZH**, isolates and monitors attacker activity, providing insights that bolster future defenses.
- **Policy** P_4: Network Throttling, used with **OOC** to moderate traffic during DoS attacks, prevents system overload by controlling incoming flow.
- **Policy** P_5: SYN Flood Protection, deployed with **PZH**, protects critical handshake processes, securing the network against SYN-based attacks.

The PPO agent's objective is to maximize the expected cumulative reward R, given by:

$$R = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t r_t\right] \tag{5.5}$$

where r_t reflects the effectiveness of the selected pattern-policy combination in mitigating threats. Rewards are structured to reinforce the successful detection and mitigation of attacks.The PPO objective function $L^{\text{PPO}}(\theta)$ stabilizes policy updates:

$$L^{\text{PPO}}(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right] \tag{5.6}$$

where $\hat{A}_t$ represents the advantage function, showing the value of the selected pattern-policy combination over alternatives.

Algorithm 5: PPO-Based Dynamic Pattern and Policy Selection for Threat Mitigation

```
Input: PPO agent with policy $\pi_\theta(a|S_t)$, value function $V_\theta(S_t)$, learning rate $\gamma$, and clipping parameter $\epsilon$
Output: Optimal action $a_t$ based on the highest probability
1  Set OOC and PZH as baseline patterns for normal network conditions;
2  foreach episode do
3      Initialize state $S_t$;
4      while episode is active do
5          Calculate action probabilities $\pi_\theta(a|S_t)$;
6          Select action $a_t$ (OOC or PZH) with the highest probability;
7          if threat detected then
8              if $a_t$ is OOC then
9                  Apply OOC-relevant policies:;
                       • Policy 1: IP Blocking
                       • Policy 2: Firewall Activation
                       • Policy 4: Network Throttling
10             else
11                 Apply PZH-relevant policies:;
                       • Policy 3: Honeypot Redirection
                       • Policy 5: SYN Flood Protection
12         else
13             Maintain OOC as the default pattern;
14         Observe reward $r_t$, update state $S_{t+1}$, and compute advantage $\hat{A}_t$;
15         if end of training period then
16             Update policy $\pi_\theta$ with PPO objective function $L^{\text{PPO}}(\theta)$;
17         Set $S_t = S_{t+1}$;
18     Log performance metrics and frequency of selected patterns;
```

5.6 Experimental Results

This section presents the findings of the experiments, followed by the analysis of the findings by applying the six selected patterns discussed in Section 2.5.4.

5.6.1 Impact of OOC and PZH Security Patterns on CPU Load, Security, CPU Usage, and Energy Consumption

To address RQ1, the experimental findings indicate that while the tested edge gateways were resilient to DDoS attacks, they experienced a significant increase in CPU load. Despite the slowdown in performance, the applications continued to function due to the OOC and PZH

security patterns. However, the experiments uncovered a vulnerability to MITM attacks, where attackers successfully spoofed the edge gateway's identity to intercept data, bypassing the PZH pattern's protection. This suggests that the PZH pattern is ineffective against MITM attacks, as it cannot prevent the forging of the gateway's MAC address.
Figure 5.4, 5.5, and 5.6 show the impact of OOC and PZH patterns under DDoS and MITM attacks. Without the OOC and PZH patterns, CPU load ranged from 0.5% to 2.75%. However, with these patterns, CPU load increased significantly, ranging from 1% to 16%, representing a six-fold increase. During a DDoS attack, the load reached 16.5% to 19%, slowing application performance as resources were redirected to manage the increased demand. Overall, the OOC and PZH patterns substantially increased CPU load, even under normal conditions, and the impact was even more pronounced during attacks.

To address RQ3, the following set of null hypotheses are defined to answer RQ3: H^x, $x \in \{1....6\}$. Hypotheses are defined as follows where P_0 corresponds to the version of application that does not use patterns, $P_{1,3}$ indicates the applied patterns, namely PZH and OOC (see Table 5.2), $P_{1,3} - DDoS$ and $P_{1,3} - MITM$ represent the occurrence or presence of the DDoS and the MITM attacks in $P_{1,3}$, respectively:
$H^1_{1,3}$: There exists no difference in the average energy consumption between $P_{1,3}$ and P_0.
$H^2_{1,3}$: There is no difference in the average energy consumption between $P_{1,3}$ and $P_{1,3} - DDoS$.
$H^3_{1,3}$: The average amount of energy consumed by $P_{1,3}$ is not different from the energy consumed by $P_{1,3} - MITM$.
$H^4_{1,3}$: There is no difference between the average amount of CPU usage of $P_{1,3}$ and P_0.
$H^5_{1,3}$: There is no difference between the average CPU usage $P_{1,3}$ and the CPU usage observed for $P_{1,3} - DDoS$.
$H^6_{1,3}$: The average amount of CPU usage in $P_{1,3}$ is not different from $P_{1,3} - MITM$. Table 5.3 reports the P-values of the Mann-Whitney U test and Cliff's δ effect size for energy consumption evaluation. The table compares the average energy consumption observed during attack scenarios (MITM and DDoS) with/without security patterns.
According to the P-value in Table 5.3, we reject $H^1_{1,3}$ for all applications. Based on the data, the evidence suggests that a statistically significant disparity exists in the average energy consumption levels across three distinct applications when comparing the utilization of OOC and PZH patterns without them.
According to the P-value for the MITM attack, we reject $H^3_{1,3}$ for the smart home application since the data analysis reveals a statistically significant difference in the energy consumption by $P_{1,3} - MITM$. However, we cannot reject $H^3_{1,3}$ for the smart city and healthcare cases. The data analysis reveals no statistically significant difference in the energy consumption by $P_{1,3} - MITM$.

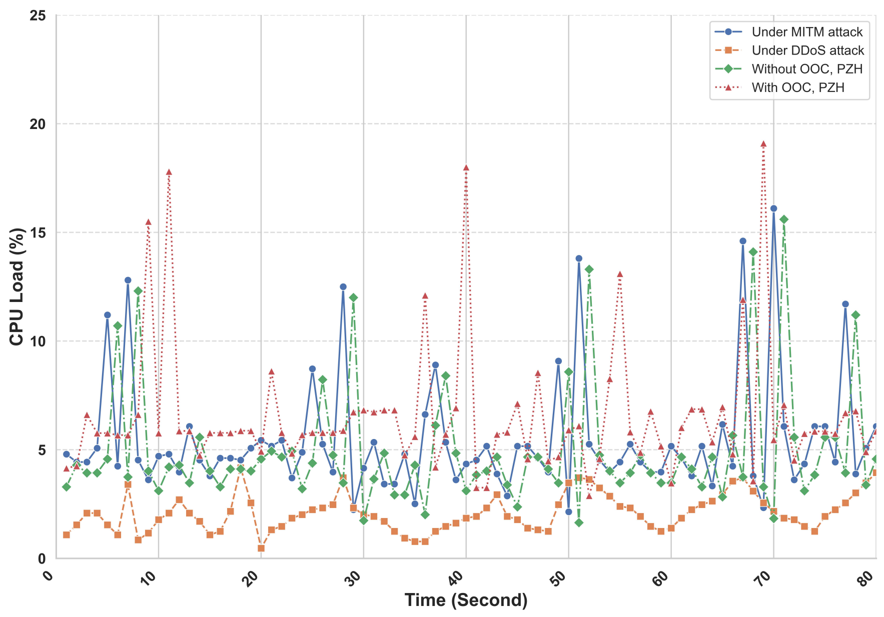

Figure 5.4 The CPU load for a smart home application with OOC, PZH patterns, without OOC, PZH patterns, and under DDoS and MITM attack.

Table 5.3 P-values, Cliff's δ effect sizes, and Confidence Intervals (CI) for average energy consumption under MITM and DDoS attacks.

Version	SmartHome			SmartCity			HealthCare		
	P-Value	Effect Size	CI (95%)	P-Value	Effect Size	CI (95%)	P-Value	Effect Size	CI (95%)
P_0 vs. $P_{1,3}$	0.0001	0.1	48.13–51.87	0.0001	0.0	53.68–56.32	0.0001	0.0	57.39–62.61
$P_{1,3}$ vs. $P_{1,3} - MITM$	0.0226	0.493	46.59–49.41	0.3173	0.218	51.68–54.32	0.9840	0.112	57.02–60.98
$P_{1,3}$ vs. $P_{1,3} - DDoS$	0.0120	0.542	50.13–53.87	0.0238	0.428	54.01–57.99	0.0794	0.578	59.39–62.61

According to the P-value for the DDoS attack, we reject $H^2_{1,3}$ for the smart home and smart city applications. The findings indicate a statistically significant difference in the energy consumption of by $P_{1,3} - DDoS$. However, we cannot reject $H^2_{1,3}$ for the healthcare applications, implying that there is no significant difference from the energy consumed by $P_{1,3} - DDoS$. According to the effect sizes and confidence intervals presented in Table 5.3, the impact of IoT security patterns and attack scenarios on energy consumption across the edge gateways. For smart home, the effect size for P_0 vs. $P_{1,3}$ is small (0.1), indicating the minimal practical significance of security patterns under normal conditions. The confidence interval (48.13–51.87) reflects consistent energy consumption in this scenario. However, under attacks, medium effect sizes (0.493 for $P_{1,3}$ vs. $P_{1,3} - MITM$ and 0.542 for $P_{1,3}$ vs. $P_{1,3} - DDoS$) demonstrate moderate practical significance, with confidence intervals (46.59–49.41 and 50.13–53.87, respectively) indicating slightly increased variability.

For smart city, the effect size for P_0 vs. $P_{1,3}$ is (0.0), suggesting no significant impact of the

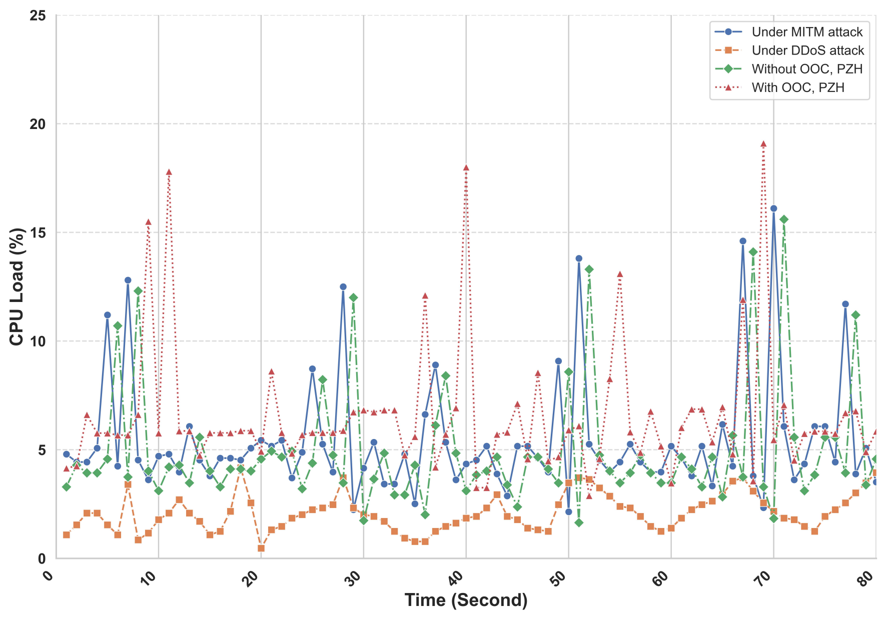

Figure 5.5 The CPU load for smart city applications with OOC, PZH patterns, without OOC, PZH patterns, and under DDoS and MITM attacks.

Table 5.4 P-value of Mann–Whitney U test, Cliff's δ effect size, and Confidence Intervals (CI) for the average CPU usage under MITM and DDoS attack.

Version	**SmartHome**			**SmartCity**			**HealthCare**		
	P-Value	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**
P_0 vs. $P_{1,3}$	0.0105	0.73	9.26–10.74	0.3320	0.213	14.24–15.76	0.0818	0.738	11.04–12.96
$P_{1,3}$ vs. $P_{1,3} - MITM$	0.0004	0.165	10.79–13.21	0.0009	0.667	16.15–19.85	0.0054	0.747	12.27–15.73
$P_{1,3}$ vs. $P_{1,3} - DDoS$	0.0001	1.0	18.41–21.59	0.0001	1.0	23.70–26.30	0.0001	1.0	20.49–23.51

patterns during normal operations, with a narrow confidence interval (53.68–56.32). During attack scenarios, the effect size increases to small (0.218 for $P_{1,3}$ vs. $P_{1,3} - MITM$) and medium (0.428 for $P_{1,3}$ vs. $P_{1,3} - DDoS$), with confidence intervals (51.68–54.32 and 54.01–57.99, respectively), showing moderate variability in energy consumption.

In health care, the effect sizes for all scenarios are minimal under normal operations (P_0 vs. $P_{1,3}$: 0.0) but increase to small (0.112 for $P_{1,3}$ vs. $P_{1,3} - MITM$) and medium (0.578 for $P_{1,3}$ vs. $P_{1,3} - DDoS$) under attack conditions. The confidence intervals widen from (57.39–62.61) under normal operations to (57.02–60.98) for MITM attacks and 59.39–62.61 for DDoS attacks, indicating increased variability.

Fig. 5.7 shows the results obtained for all the OOC and PZH pattern implementations. The examination of energy consumption patterns reveals a discernible escalation in energy utilization. This effect becomes more pronounced when subjecting the applications (with patterns) to DDoS and MITM attacks, further exacerbating energy consumption levels. Nonetheless,

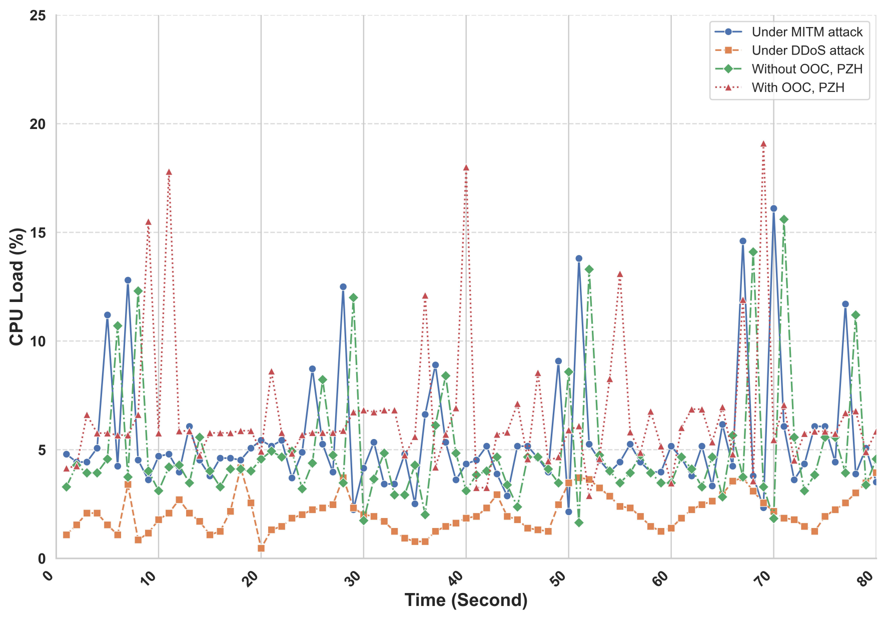

Figure 5.6 The CPU load for a healthcare application with OOC, PZH patterns, without OOC, PZH patterns, and under DDoS and MITM attacks.

a statistically noteworthy disparity in energy consumption becomes evident when comparing the with/without patterns in all applications. Additionally, the disparity in energy consumption attains statistical significance when assessing the impact of an MITM attack on a smart home environment. Furthermore, statistical significance manifests in energy consumption discrepancies when the smart home and city confront DDoS attack scenarios.

Table 5.4 reports the P-values of the Mann-Whitney U test and the Cliff's δ effect size for CPU usage evaluation. The table compares the average CPU usage levels observed during two distinct attack scenarios (MITM and DDoS) with/without security patterns.

According to the P-value in Table 5.4, we reject $H^4_{1,3}$ for smart home application. Our analysis indicates a statistically significant difference in the average CPU usage when OOC and PZH patterns are applied in the smart home application. However, we cannot reject $H^4_{1,3}$ smart city and healthcare applications. There is no statistically significant difference between the average amount of CPU usage under different conditions with OOC and PZH patterns and without them in the two applications.

For the MITM attack, we reject $H^6_{1,3}$ for all applications. Upon conducting the statistical tests, it is evident that there is a statistically significant difference in CPU usage by $P_{1,3} - MITM$.

In the case of the DDoS attack, we reject $H^6_{1,3}$ for all applications. After conducting the statistical tests, compelling evidence arises, indicating a statistically significant difference in

CPU usage caused by $P_{1,3} - DDoS$.

According to the effect sizes and confidence intervals presented in Table 5.4, the analysis reveals significant impacts of IoT security patterns and attack scenarios on CPU usage across the edge gateways.

For smart home, the medium to large effect sizes (e.g., 0.73 for P_0 vs. $P_{1,3}$ and 1.0 for $P_{1,3}$ vs. $P_{1,3}$-DDoS) indicate substantial increases in CPU usage, with relatively narrow confidence intervals during normal conditions (e.g., 9.26–10.74) and broader intervals under attacks (e.g., 18.41–21.59).

Similarly, for smart city, small effect sizes (e.g., 0.213 for P_0 vs. $P_{1,3}$) during normal conditions suggest minimal overhead, while medium (0.667) and large effect sizes (1.0) during attacks demonstrate significant resource impacts. Confidence intervals expand notably from 14.24–15.76 under normal conditions to 23.70–26.30 during DDoS, reflecting increased variability due to attack response.

In health care, large effect sizes across all comparisons (e.g., 0.738 for P_0 vs. $P_{1,3}$ and 1.0 for $P_{1,3}$ vs. $P_{1,3}$-DDoS) highlight the considerable CPU usage increase from both security mechanisms and attack scenarios. The confidence intervals, such as 11.04–12.96 during normal conditions and 20.49–23.51 under DDoS, illustrate consistent impacts during regular operations and increased variability under severe attack conditions.

Fig. 5.8 summarizes the results obtained for all the OOC and PZH pattern implementations. The examination of CPU usage during DDoS attacks reveals an increase when patterns are introduced. In addition, it is imperative to emphasize that a statistically significant variance in CPU usage is observed between scenarios for the smart home with and without patterns unless for the smart city and healthcare case. Moreover, a statistically significant variance in CPU usage is observed when applications are under DDoS and MITM attacks.

Finding 1:

The PZH mechanism indicates limitations in effectively countering MITM attacks. Additionally, the pattern's impact on energy consumption in all applications becomes evident in diverse situations, including those involving attacks and non-attack circumstances, except the smart city and healthcare under MITM attack and healthcare under DDoS attack. Furthermore, the patterns' impact on CPU usage in all applications becomes evident in diverse situations, including attacks and non-attack circumstances, except for the smart city and healthcare with/without patterns.

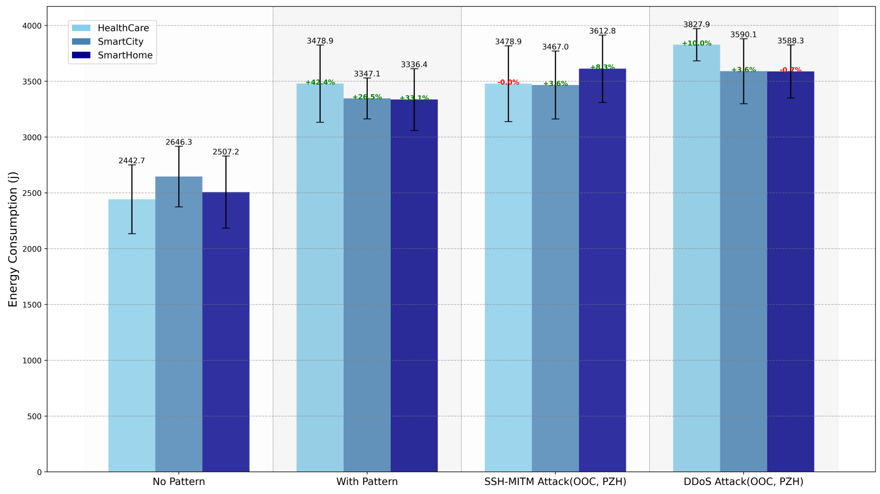

Figure 5.7 Energy consumption for different applications (smart home, smart city, and healthcare) with OOC, PZH patterns, without OOC, PZH patterns, and under DDoS and MITM attacks.

5.6.2 Impact of WL and BL Security Patterns on CPU Load, Security, CPU Usage, and Energy Consumption

To address RQ1, we analyze the system's behavior under the application of the WL and BL security patterns during a brute-force attack and in a normal operational state. Figures 5.9, 5.10, and 5.11 present the CPU load variations across smart home, smart city, and healthcare applications, respectively. Under normal conditions, CPU usage ranges from 0.5% to 1.75% without the WL and BL patterns. However, when the WL and BL patterns are activated to enhance access control, CPU load increases from 1% to 2.5%, indicating moderate additional resource usage to maintain security protocols. During a brute-force attack, the CPU load further escalates, spiking between 2% and 5.5% as the system intensifies security measures to block unauthorized access attempts. Despite the increased CPU load, the system remained operational and resilient under attack conditions, effectively managing resources while blocking unauthorized users.

To address RQ3, the following set of null hypotheses are defined to answer RQ3: H^x, $x \in \{1....6\}$. Hypotheses are defined as follows where P_0 corresponds to the version of applications that do not use patterns, $P_{4,5}$ indicates the applied patterns, namely WL and BL (see Table 5.2), $P_{4,5} - BF$ indicates the occurrence or presence of brute-force attack in $P_{4,5}$, respectively:

$H^1_{4,5}$: There is no difference between the average energy consumption of $P_{4,5}$ and P_0.

$H^2_{4,5}$: The average amount energy consumption of $P_{4,5}$ is not different from $P_{4,5} - BF$.

$H^3_{4,5}$: There is no difference between the average amount of CPU usage of $P_{4,5}$ and P_0.

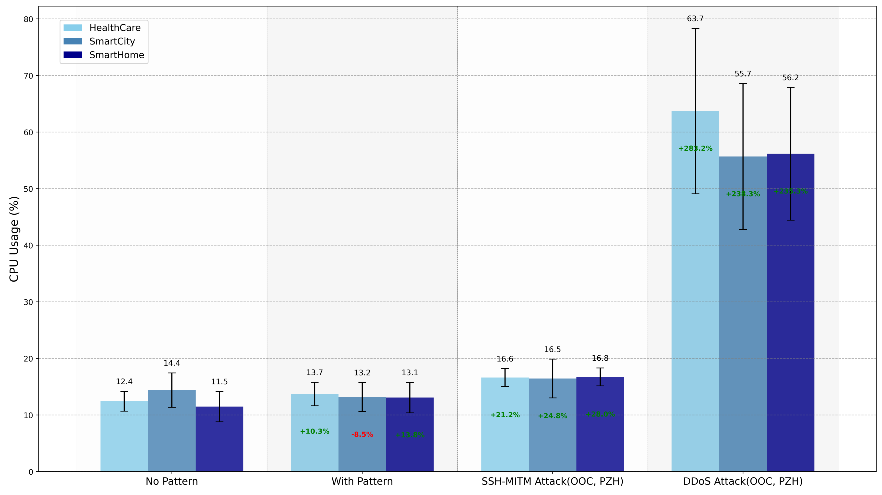

Figure 5.8 CPU usage for different applications (smart home, smart city, and healthcare) with OOC, PZH patterns, without OOC, PZH patterns, and under DDoS and MITM attacks.

$H^4_{4,5}$: The average amount of CPU usage by $P_{4,5}$ is not different from the $P_{4,5} - BF$.

Table 5.5 reports the P-values of the Mann-Whitney U test and Cliff's δ effect size for energy consumption evaluation. The table compares the average energy consumption levels observed during distinct attack scenarios (brute-force), with/without security patterns.

According to the P-value in Table 5.5, we reject $H^1_{4,5}$; the analysis reveals a statistically significant difference in the average energy consumption when WL and BL patterns were applied in all applications.

For the brute-force attack, we reject $H^2_{4,5}$ for all applications. The results reveal that there is a statistically significant difference between the average amount of energy consumed when using WL and BL patterns for all examined applications under $P_{4,5} - BF$.

According to the effect sizes and confidence intervals presented in Table 5.5, the analysis highlights the impact of brute-force attacks on energy consumption across the edge gateways.

For smart home, the effect size for P_0 vs. $P_{4,5}$ is calculated as (0.0), indicating no practical significance of the energy consumption differences under normal conditions, as confirmed by the narrow confidence interval (48.35–51.65). However, the large effect size (0.819) for $P_{4,5}$ vs. $P_{4,5} - BF$ demonstrates a substantial increase in energy consumption during brute-force attacks, with a confidence interval of 53.89–56.11 reflecting moderate variability.

Similarly, in smart city, the effect size of 0.0 for P_0 vs. $P_{4,5}$ indicates no significant impact of the baseline conditions on energy consumption, supported by the narrow confidence interval (58.45–61.55). In contrast, the huge effect size (0.929) for $P_{4,5}$ vs. $P_{4,5} - BF$ signifies a significant increase in energy demand during attacks, with the confidence interval (63.56–66.44)

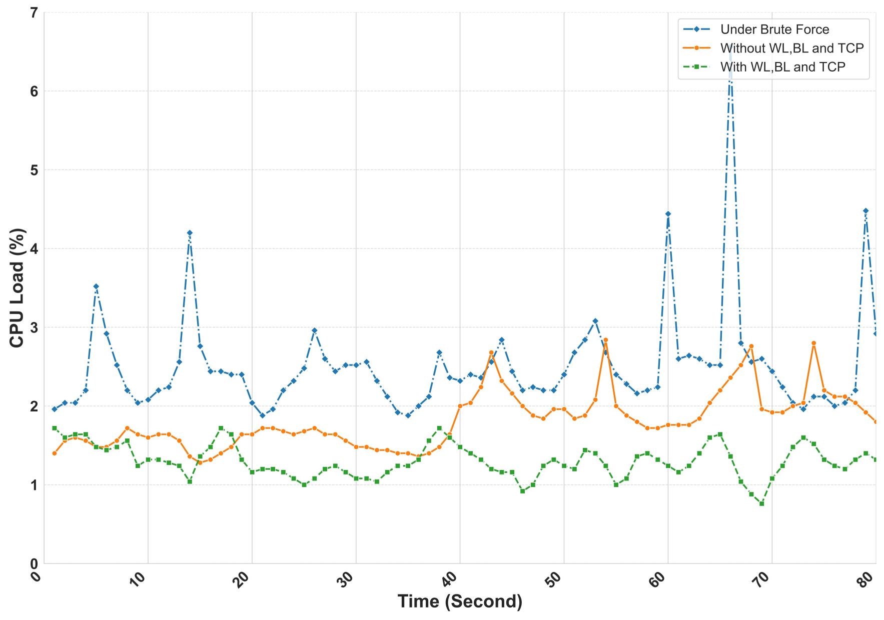

Figure 5.9 CPU load for a smart home application with WL, BL without WL, BL, and under brute-force attack.

Table 5.5 P-values, Cliff's δ effect sizes, and Confidence Intervals (CI) for average energy consumption under brute-force attack.

Version	**SmartHome**			**SmartCity**			**HealthCare**		
	P-Value	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**
P_0 vs. $P_{4,5}$	0.0001	0.0	48.35–51.65	0.0001	0.0	58.45–61.55	0.0001	0.867	68.61–71.39
$P_{4,5}$ vs. $P_{4,5} - BF$	0.0008	0.819	53.89–56.11	0.0001	0.929	63.56–66.44	0.0018	0.809	73.56–76.44

indicating consistent but higher energy usage.

For health care, the significant effect size (0.867) for P_0 vs. $P_{4,5}$ already suggests a significant baseline energy consumption difference, likely due to the higher demands in this environment, with the confidence interval of 68.61–71.39 indicating stable usage. The considerable effect size (0.809) for $P_{4,5}$ vs. $P_{4,5} - BF$ further underscores the significant impact of brute-force attacks, with the confidence interval (73.56–76.44) reflecting higher variability under attack conditions.

Fig. 5.12 illustrates the results obtained for all the WL and BL pattern implementations. The analysis of energy consumption during a brute-force attack indicates an escalation when patterns are introduced. Also, our results show a statistically significant difference in any cases under investigation.

Table 5.6 reports the P-values of the Mann-Whitney U test and the Cliff's δ effect size for CPU usage evaluation. The table compares the average CPU usage observed during distinct attack scenarios (i.e., brute-force) with/without security patterns.

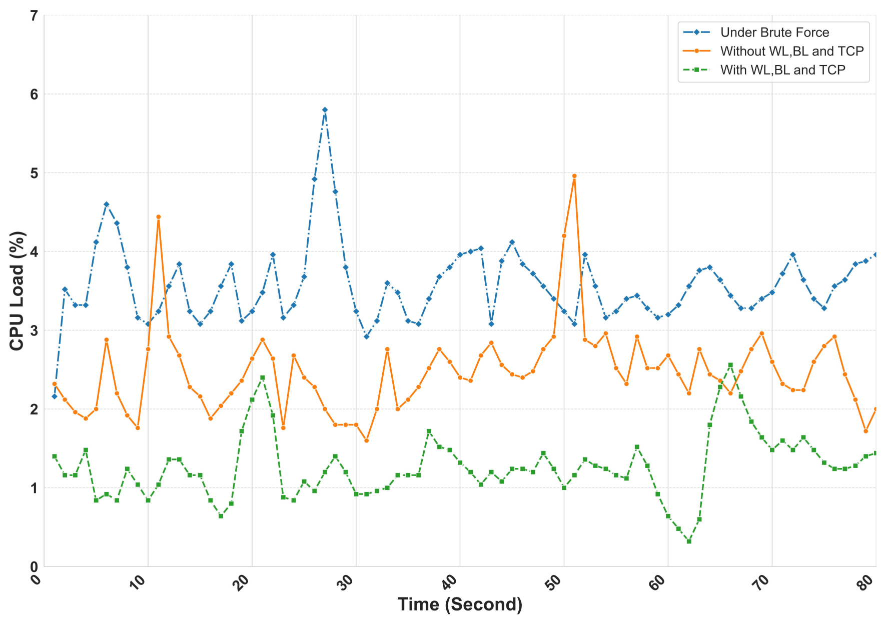

Figure 5.10 CPU load for a smart city application with WL, BL without WL, BL, and under brute-force attack.

Table 5.6 P-values, Cliff's δ effect sizes, and Confidence Intervals (CI) for average CPU usage under brute-force attack.

Version	SmartHome			SmartCity			HealthCare		
	P-Value	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**
P_0 vs. $P_{4,5}$	0.0028	0.782	19.26–20.74	0.0057	0.391	21.59–22.41	0.2996	0.471	23.41–24.59
$P_{4,5}$ vs. $P_{4,5} - BF$	0.0001	1.0	24.26–25.74	0.0001	1.0	26.42–27.58	0.0001	0.996	29.41–30.59

According to the P-value in Table 5.6, we reject $H^3_{4,5}$ in smart home and smart city applications. The analysis unequivocally demonstrates a statistically significant difference in the average CPU usage across various cases, considering the presence of WL and BL patterns, compared to cases without WL and BL in smart home and smart city applications. Moreover, we cannot reject $H^4_{4,5}$ for healthcare applications as there is no statistically significant difference in CPU usage with/without WL and BL patterns.

For the brute-force attack, we reject $H^3_{4,5}$ in all applications. The analysis unequivocally demonstrates a statistically significant difference in the average CPU usage across various cases when using $P_{4,5} - BF$.

According to the effect sizes and confidence intervals presented in Table 5.6, the analysis demonstrates the significant impacts of brute-force attacks on CPU usage across the edge gateways.

For smart home, the effect size for P_0 vs. $P_{4,5}$ is large (0.782), indicating a considerable increase in CPU usage under baseline conditions, with a confidence interval of 19.26–20.74,

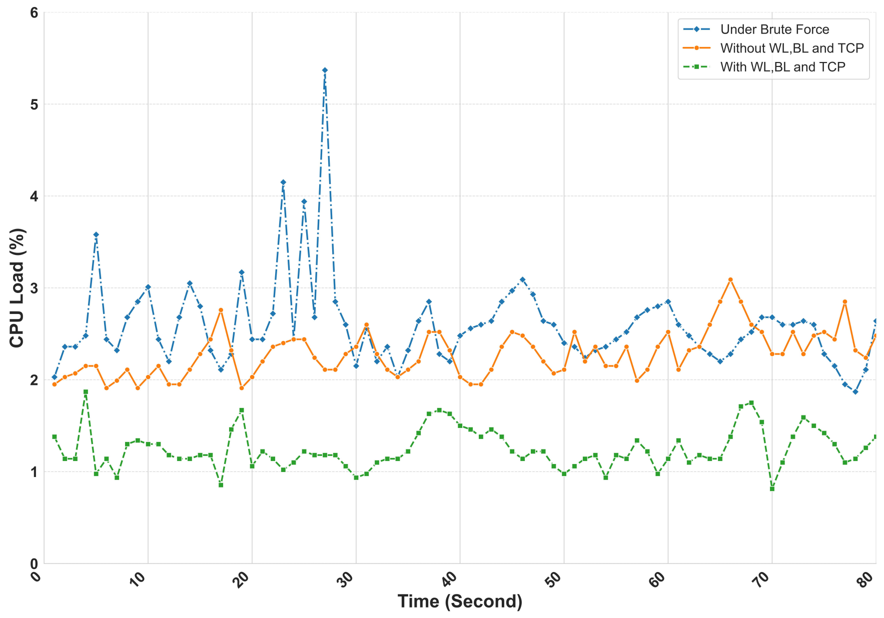

Figure 5.11 CPU load for a healthcare application with WL, BL without WL, BL, and under brute-force attack.

reflecting relatively stable estimates. For $P_{4,5}$ vs. $P_{4,5} - BF$, the effect size increases to 1.0, signifying a substantial and consistent impact of brute-force attacks, as seen in the confidence interval of 24.26–25.74.

In smart city, a medium effect size (0.391) for P_0 vs. $P_{4,5}$ suggests moderate increases in CPU usage under baseline conditions, with the confidence interval (21.59–22.41) showing minimal variability. Under attack scenarios ($P_{4,5}$ vs. $P_{4,5} - BF$), the effect size rises to 1.0, indicating a significant and consistent increase in CPU usage, supported by the confidence interval of 26.42–27.58.

For health care, the effect size for P_0 vs. $P_{4,5}$ is moderate (0.471), reflecting perceptible CPU overhead during baseline operations, with the confidence interval (23.41–24.59) showing stable usage. Under brute-force attacks ($P_{4,5}$ vs. $P_{4,5} - BF$), the effect size approaches perfection (0.996), highlighting a substantial impact on CPU usage, as evidenced by the confidence interval of 29.41–30.59.

Fig. 5.13 shows the results obtained for all the WL and BL pattern implementations. The analysis of CPU usage during a brute-force attack indicates an escalation when patterns are introduced. However, our analysis identifies a statistically significant difference in any cases under investigation, except healthcare with/without patterns.

Finding 2:

Our experimental findings suggest the efficacy of WL and BL patterns against brute-force attacks. Notably, incorporating patterns impacts CPU usage and energy consumption in active-attack scenarios and the absence of attacks. Also, results show a statistically significant difference in any cases under investigation.

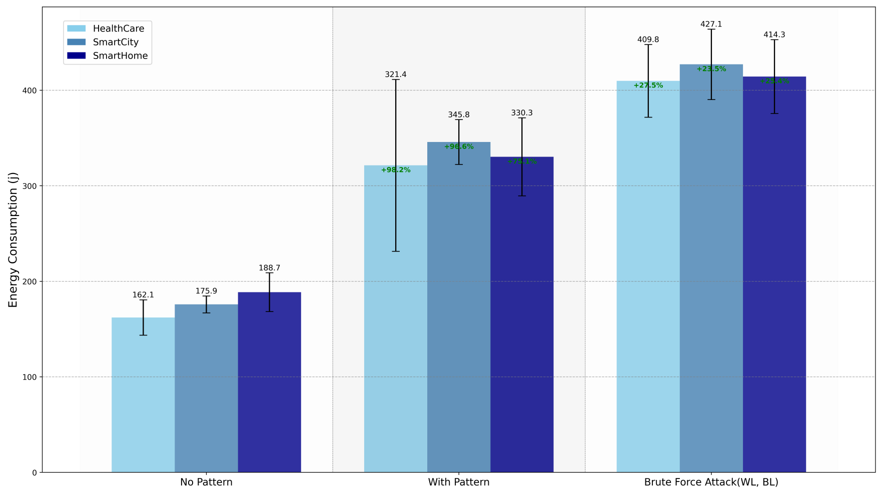

Figure 5.12 Energy consumption for smart home, smart city, and healthcare applications with WL, BL patterns, without them, and under brute-force attack.

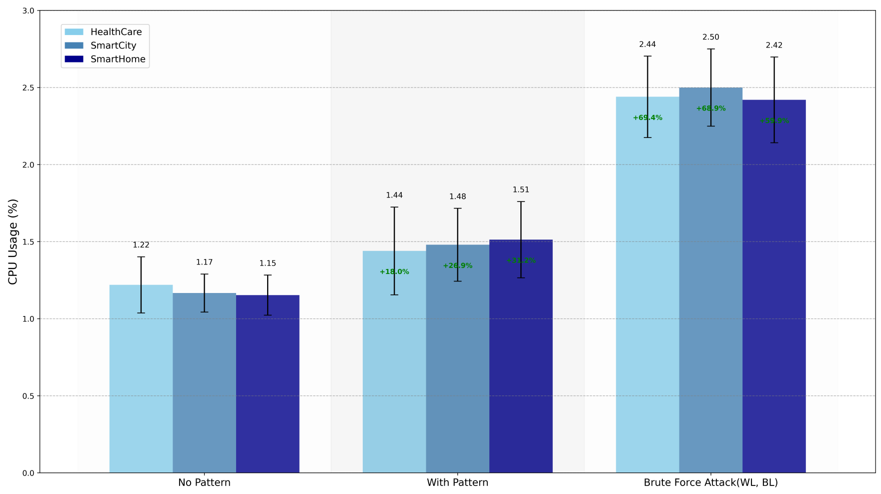

Figure 5.13 CPU usage for smart home, smart city, and healthcare applications with WL, BL patterns, without them, and under brute-force attack.

5.6.3 Impact of WL, BL, and SNN Security Patterns on CPU Load, Security, CPU Usage, and Energy Consumption

To address RQ1, we assess the WL, BL, and SSN security patterns in managing and mitigating cyber threats, particularly during brute-force and SSH-MITM attacks. These patterns were implemented to enhance the security of the smart home, smart city, and healthcare applications, focusing on maintaining a stable CPU load under various conditions. Figures 5.14, 5.15, and 5.16 illustrate the impact of the **WL**, **BL**, and **SSN** security patterns on CPU load across these applications. Testing was conducted in two primary scenarios: regular operation under active brute-force and MITM attacks. The findings indicate that, without the security patterns, CPU load remains relatively low, ranging from 0.5% to 2%. When the WL, BL, and SSN patterns are applied, CPU load increases slightly, from 2% to 3.5%, reflecting the additional resources required for enhanced security. Under SSH-MITM attack conditions, CPU load rises further, ranging between 2.5% and 4.5%, as the system actively enforces access controls and data protection.

To address RQ3, the following set of null hypotheses are defined to answer RQ3: H^x, $x \in \{1....6\}$. Hypotheses are defined as follows where P_0 corresponds to the version of application that does not use patterns, $P_{4,5,6}$ indicates the applied patterns, namely WL, BL, and SSN (see Table 5.2), $P_{4,5,6} - BF$ and $P_{4,5,6} - MITM$ indicate the occurrence or presence of brute-force and MITM attacks in $P_{4,5,6}$, respectively:

$H^1_{4,5,6}$: There exists no difference between the average amount of energy consumed by $P_{4,5,6}$ and P_0.

$H^2_{4,5,6}$: There is no difference in the average energy consumption between $P_{4,5,6}$ and $P_{4,5,6} - MITM$.

$H^3_{4,5,6}$: The average amount of the CPU usage by $P_{4,5,6}$ is not different from $P_{4,5,6} - BF$.

$H^4_{4,5,6}$: There is no difference between the average amount of CPU usage by $P_{4,5,6}$ and P_0.

$H^5_{4,5,6}$: The average CPU usage of $P_{4,5,6}$ indicates no difference when compared to $P_{4,5,6} - MITM$.

$H^6_{4,5,6}$: The average CPU usage in $P_{4,5,6}$ is not different from $P_{4,5,6} - BF$.

Table 5.7 reports the P-values of the Mann-Whitney U test and Cliff's δ effect size for energy consumption evaluation. The table compares the average energy consumption observed during two attack scenarios (MITM, brute-force) with/without security patterns.

According to the P-value reported in Table 5.7, we reject $H^1_{4,5,6}$ for all applications; based on the observed data, there exists a statistically significant disparity in the average energy consumption across various cases when considering the presence or absence of WL, BL, and SSN patterns in all applications.

For the MITM attack, We also reject $H^2_{4,5,6}$ for smart home and smart city applications as

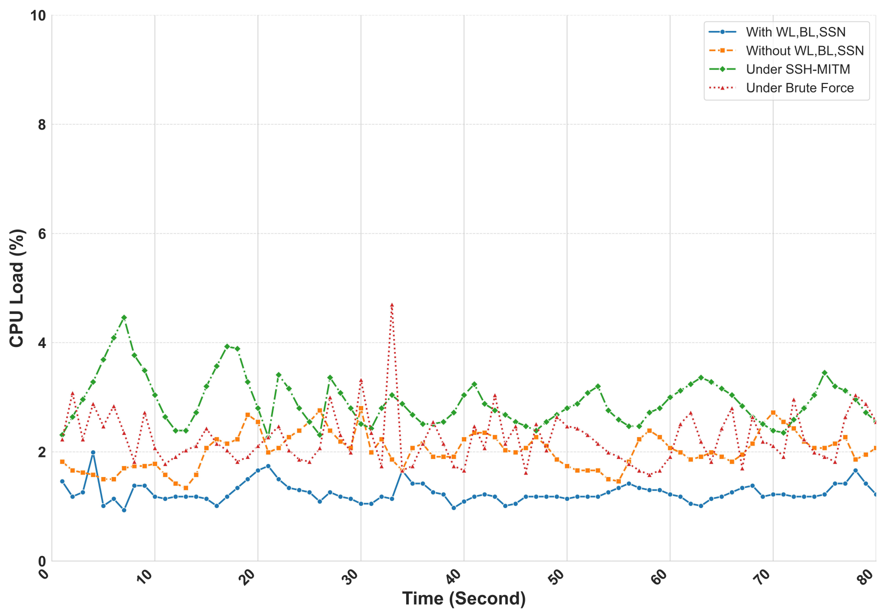

Figure 5.14 CPU load for smart home applications with WL, BL, SSN, without WL, BL, SSN, and under brute-force attack as well as MITM.

the data-driven analysis corroborates the belief that there is a statistically significant difference in the energy consumed by $P_{4,5,6} - MITM$. Moreover, we cannot reject $H^2_{4,5,6}$ for the healthcare application. The analysis unequivocally demonstrates no statistically significant difference in the average CPU usage by using $P_{4,5,6} - MITM$.

For the brute-force attack, we reject $H^3_{4,5,6}$ in all applications. The data reveals a statistically significant difference in the energy consumed by $P_{4,5,6} - BF$.

According to the effect sizes and confidence intervals presented in Table 5.7, the analysis highlights the significant impact of MITM and brute-force attacks on energy consumption across the edge gateways.

For smart home, the effect size for P_0 vs. $P_{4,5,6}$ is 1.0, indicating a substantial difference in energy consumption under normal and attack-inclusive conditions, with the confidence interval (48.35–51.65) reflecting stable variability. Similarly, for $P_{4,5,6}$ vs. $P_{4,5,6} - MITM$ and $P_{4,5,6}$ vs. $P_{4,5,6} - BF$, the effect size remains 1.0, demonstrating consistent and significant increases in energy consumption during these attacks, as evidenced by confidence intervals of 53.89–56.11 and 58.89–61.11, respectively.

For smart city, the effect size for P_0 vs. $P_{4,5,6}$ is also 1.0, with a confidence interval (58.45–61.55) indicating consistent energy demands in the baseline scenario. Under attack conditions, the effect size for $P_{4,5,6}$ vs. $P_{4,5,6} - MITM$ is 0.893, highlighting a slightly lower but still significant increase in energy consumption, with a wider confidence interval (63.56–66.44) reflecting

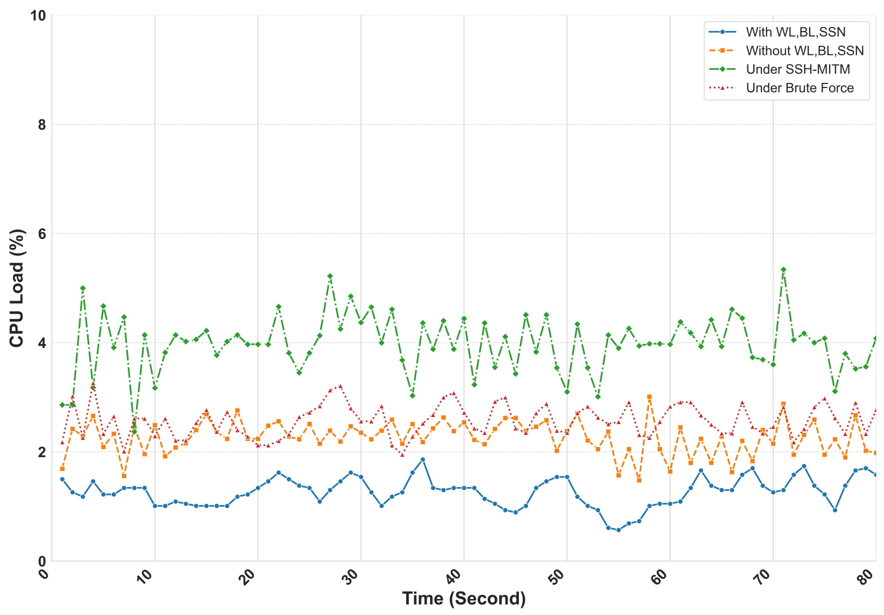

Figure 5.15 CPU load for smart city applications with WL, BL, SSN, without WL, BL, SSN, and under brute-force attack as well as MITM.

more variability. The effect size for $P_{4,5,6}$ vs. $P_{4,5,6} - BF$ returns to 1.0, with a confidence interval of 68.56–71.44, signifying substantial energy consumption under brute-force attacks. In health care, the effect size for P_0 vs. $P_{4,5,6}$ is 1.0, with the confidence interval (68.61–71.39) showing stable energy consumption during normal operations. For $P_{4,5,6}$ vs. $P_{4,5,6} - MITM$, the effect size drops to 0.098, suggesting minimal practical significance in energy consumption differences, with a confidence interval
(73.56–76.44) indicating moderate variability. However, for $P_{4,5,6}$ vs. $P_{4,5,6} - BF$, the effect size returns to 1.0, with a confidence interval (78.56–81.44), emphasizing a significant increase in energy consumption during brute-force attacks.

Fig. 5.17 shows the results obtained for all investigated scenarios for WL, BL, and SSN patterns. The analysis of energy consumption during brute-force and MITM attacks reveals a jump when patterns are introduced; however, there is a statistically significant variance in energy consumption. This consistent observation applies to three applications under investigation, irrespective of with and without patterns and under attack, except healthcare under MITM attack.

Table 5.8 reports the P-values of the Mann-Whitney U test and Cliff's δ effect size for CPU usage evaluation. The table compares the average CPU usage observed during two distinct attack scenarios (MITM, brute-force) with/without security patterns.

According to the P-value we reported in Table 5.8, we reject $H^4_{4,5,6}$ for all applications, as there is a statistically significant difference between the average amount of CPU usage with

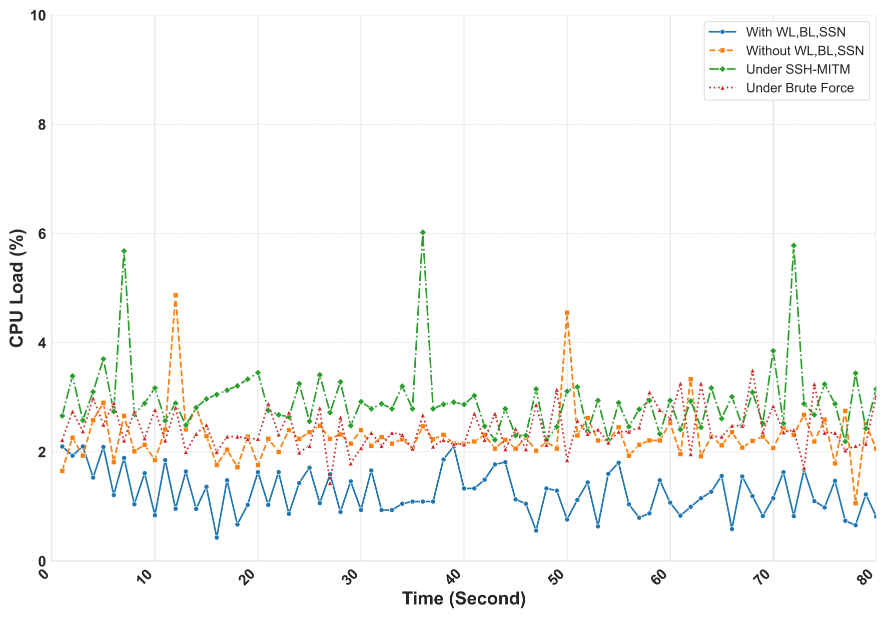

Figure 5.16 CPU load for healthcare applications with WL, BL, SSN, without WL, BL, SSN, and under brute-force attack as well as MITM.

Table 5.7 P-values, Cliff's δ effect sizes, and Confidence Intervals (CI) for average energy consumption under MITM and brute-force attacks.

Version	**SmartHome**			**SmartCity**			**HealthCare**		
	P-Value	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**
P_0 vs. $P_{4,5,6}$	0.0001	1.0	48.35–51.65	0.0009	1.0	58.45–61.55	0.0005	1.0	68.61–71.39
$P_{4,5,6}$ vs. $P_{4,5,6} - MITM$	0.0002	1.0	53.89–56.11	0.0001	0.893	63.56–66.44	0.6599	0.098	73.56–76.44
$P_{4,5,6}$ vs. $P_{4,5,6} - BF$	0.0001	1.0	58.89–61.11	0.0001	1.0	68.56–71.44	0.0006	1.0	78.56–81.44

and without WL, BL, and SSN patterns in all cases.

In the case of the MITM attack, we reject $H^5_{4,5,6}$ for all applications. The analysis reveals a statistically significant disparity in the CPU usage of $P_{4,5,6} - MITM$.

For the brute-force attack, we reject $H^6_{4,5,6}$ for all applications. The data does indicate a statistically significant difference in the CPU usage of $P_{4,5,6} - BF$.

According to the effect sizes and confidence intervals presented in Table 5.8, the analysis highlights the significant impact of MITM and brute-force attacks on CPU usage across the edge gateways.

For smart home, the effect size for P_0 vs. $P_{4,5,6}$ is 1.0, indicating a substantial difference in CPU usage between baseline and attack-inclusive conditions, with the confidence interval (29.19–30.81) reflecting stable estimates. Under attack scenarios, the effect size remains 1.0 for both $P_{4,5,6}$ vs. $P_{4,5,6}$-MITM and $P_{4,5,6}$ vs. $P_{4,5,6}$-BF, with confidence intervals (34.15–35.85 and 39.13–40.87, respectively) showing increasing variability during attacks.

For smart city, the effect size of 1.0 for P_0 vs. $P_{4,5,6}$ suggests a consistent and substantial

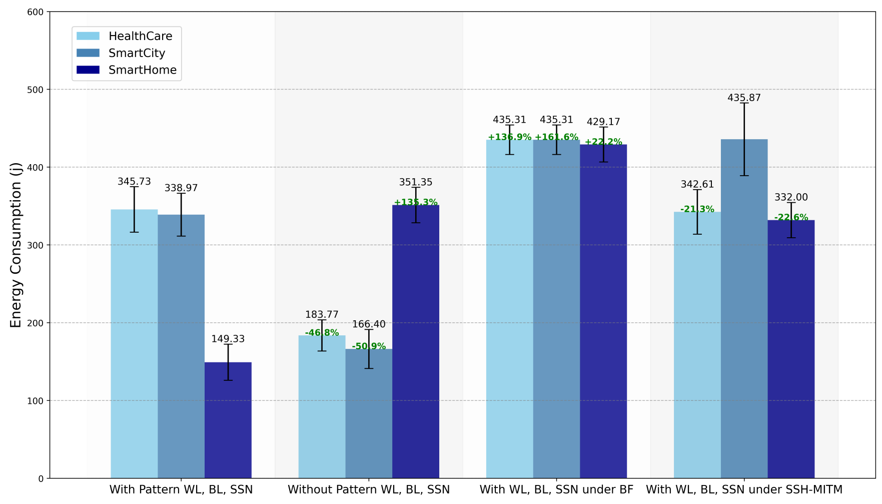

Figure 5.17 Energy consumption for smart home, smart city, and healthcare applications with WL, BL, SSN patterns, without them, and under brute-force and MITM attacks.

Table 5.8 P-values, Cliff's δ effect sizes, and Confidence Intervals (CI) for average CPU usage under MITM and brute-force attacks.

Version	**SmartHome**			**SmartCity**			**HealthCare**		
	P-Value	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**
P_0 vs. $P_{4,5,6}$	0.0008	1.0	29.19–30.81	0.0007	1.0	38.77–41.23	0.0001	1.0	48.56–51.44
$P_{4,5,6}$ vs. $P_{4,5,6}$-MITM	0.0001	1.0	34.15–35.85	0.0002	0.893	43.63–46.37	0.0007	0.098	52.68–57.32
$P_{4,5,6}$ vs. $P_{4,5,6}$-BF	0.0001	1.0	39.13–40.87	0.0003	1.0	48.58–51.42	0.0005	1.0	58.63–61.37

increase in CPU usage, with a confidence interval of 38.77–41.23 reflecting precise estimates. During attack scenarios, the effect size decreases slightly to 0.893 for $P_{4,5,6}$ vs. $P_{4,5,6}$-MITM, indicating a moderate practical significance, with a wider confidence interval (43.63–46.37). For $P_{4,5,6}$ vs. $P_{4,5,6}$-BF, the effect size returns to 1.0, with a confidence interval (48.58–51.42) reflecting substantial increases in CPU usage during brute-force attacks.

In health care, the effect size of 1.0 for P_0 vs. $P_{4,5,6}$ highlights a significant baseline difference, with a confidence interval (48.56–51.44) showing stable CPU usage during normal operations. For $P_{4,5,6}$ vs. $P_{4,5,6}$-MITM, the effect size drops significantly to 0.098, suggesting minimal practical significance in CPU usage differences during MITM attacks, with a confidence interval (52.68–57.32) reflecting moderate variability. For $P_{4,5,6}$ vs. $P_{4,5,6}$-BF, the effect size returns to 1.0, with a confidence interval (58.63–61.37) showing a substantial increase in CPU usage during brute-force attacks.

Fig. 5.18 illustrates our results for all WL, BL, and SSN pattern implementations. The investigation of CPU usage during brute-force and MITM attacks yields an escalation when patterns are introduced. However, it is crucial to emphasize that there is a statistically significant variance in CPU usage and every consumption among scenarios under attack. This

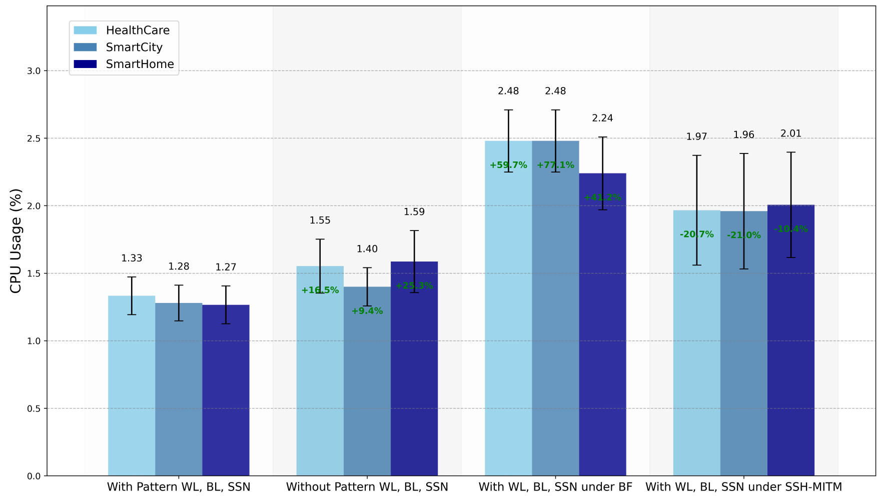

Figure 5.18 CPU usage for smart home, smart city, and healthcare applications with WL, BL, SSN patterns, without them, and under brute-force and MITM attacks.

observation holds for three applications under investigation, except healthcare under MITM attack in energy consumption.

Finding 3:

The experimental results show the efficacy of WL, BL, and SSN patterns in mitigating BF and MITM attacks. Importantly, incorporating patterns during these attacks indicates a discernible impact on CPU usage and energy consumption in all applications, with/without patterns and under attacks.

5.6.4 Impact of Combination Security Patterns on CPU Load, Security, CPU Usage, and Energy Consumption

To address RQ1, we compared all testbeds with and without the combined security patterns, subjecting the applications to various attacks, including DDoS, MITM, and brute-force, similar to previous experiments.

Figures 5.19, 5.20, and 5.21 illustrate the CPU load in the smart home, smart city, and healthcare applications under different conditions. Without any security patterns, the CPU load remains low at 2%, reflecting typical operations. With complete security patterns implemented, the load increases to 5%, indicating the overhead of security mechanisms. During SSH-MITM and brute-force attacks, the load rises to 15%, showing the system's defensive response to these moderate threats. However, under a DDoS attack, the CPU load surges to 50%, significantly impacting system performance.

To address RQ3, the following set of null hypotheses are defined to answer RQ3: H^x, $x \in \{1....8\}$. Hypotheses are defined as follows where P_0 corresponds to the version of the applications that do not use patterns, P_{all} indicates the scenario of applying all patterns, namely PZH, OOC, WL, BL, and SSN (see Table 5.2), $P_{all} - MITM$, $P_{all} - BF$ and $P_{all} - DDoS$ are indicate the occurrence or presence of MITM, brute-force, and DDoS attacks in P_{all}, respectively:

H^1_{all}: There is no difference between the average amount of energy consumed by P_{all} and P_0.

H^2_{all}: There is no difference between the average energy consumption of P_{all} and the $P_{all} - MITM$.

H^3_{all}: There is no difference in the average energy consumed by P_{all} when compared to $P_{all} - BF$.

H^4_{all}: The average amount of energy consumed by P_{all} is not different from $P_{all} - DDoS$.

H^5_{all}: There is no difference between the average CPU usage by P_{all} and P_0.

H^6_{all}: There is no difference between the average CPU usage of P_{all} and $P_{all} - MITM$.

H^7_{all}: There is no difference in the average CPU usage by P_{all} when compared to $P_{all} - BF$.

H^8_{all}: The average amount of CPU usage in P_{all} is not different from $P_{all} - DDoS$.

Table 5.9 reports the P-values of the Mann-Whitney U test and Cliff's δ effect size for energy consumption evaluation. The table compares the average energy consumption observed during three distinct attack scenarios (MITM, brute-force, and DDoS) with/without security patterns.

According to the P-value in Table 5.9, we reject H^1_{all} in all applications; the analysis suggests that there is a trace of statistically significant differentiation in the average energy consumption in the domains of all applications.

According to the P-value for all applications under the MITM attack, we reject H^2_{all} for smart home and smart city applications. Upon statistical examination, a significant variation is found in the energy consumed by $P_{all} - MITM$ compared to P_{all}. However, we cannot reject H^6_{all} for healthcare application. The analysis indicates no statistically significant difference in energy consumed by $P_{all} - MITM$.

According to the P-value for all applications under the brute-force attack, we cannot reject H^3_{all} for all applications. The analysis uncovers no statistically significant effect on the energy consumed by $P_{all} - BF$.

According to the P-value for all applications under the DDoS attack, we reject H^4_{all} for all applications as a statistically significant difference is observed in the energy consumption of $P_{all} - DDoS$ for this application.

According to the effect sizes and confidence intervals presented in Table 5.9, the analysis highlights the impact of MITM, BF, and DDoS attacks on energy consumption across the

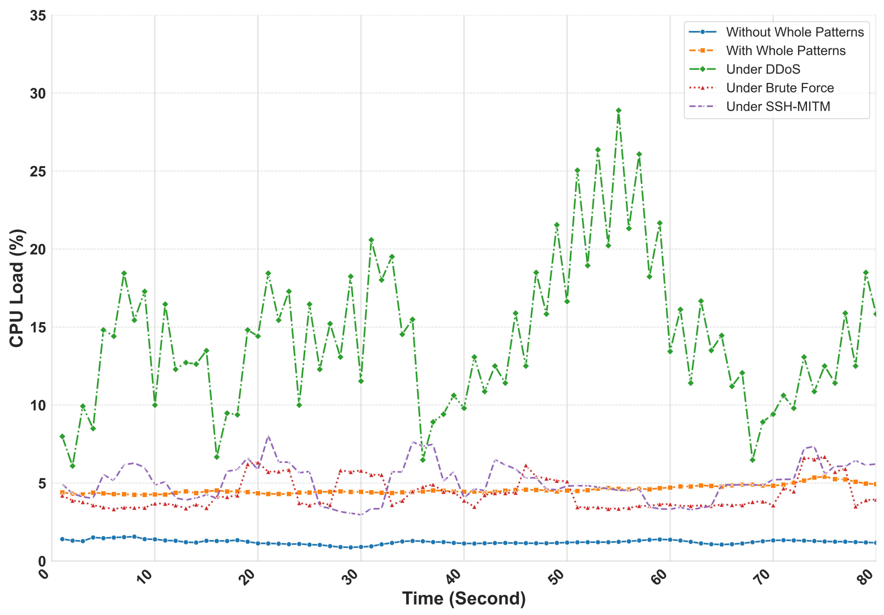

Figure 5.19 CPU load for smart home application with combination patterns, without combination patterns, and three attacks: DDoS, MITM, and brute-force.

edge gateways.

For smart home, the effect size for P_0 vs. P_{all} is 1.0, indicating a substantial increase in energy consumption under attack-inclusive conditions, with a confidence interval of 48.35–51.65 reflecting stable variability. Similarly, under MITM and DDoS attacks, the effect size remains 1.0, with confidence intervals of 53.89–56.11 and 63.89–66.11, respectively, demonstrating significant increases in energy consumption. The smaller effect size (0.347) and confidence interval (58.89–61.11) suggest moderate practical impact for BF attacks.

In smart city, the effect size of 1.0 for P_0 vs. P_{all} and P_{all} vs. P_{all}-MITM indicates consistent and significant increases in energy consumption, with confidence intervals of 58.45–61.55 and 63.56–66.44, respectively. For brute-force attacks, the effect size decreases to 0.210, reflecting minimal practical significance, with the confidence interval (68.56–71.44) indicating moderate variability. Under DDoS attacks, the effect size remains at 1.0, with a confidence interval of 73.56–76.44, showing substantial resource demands.

In health care, the effect size for P_0 vs. P_{all} and P_{all} vs. P_{all}-MITM is 1.0, highlighting significant energy consumption increases, with confidence intervals of
68.61–71.39 and 73.56–76.44, respectively. For brute-force attacks, the effect size is moderate (0.529), with the confidence interval (78.56–81.44) reflecting perceptible but less pronounced increases in energy demands. During DDoS attacks, the effect size returns to 1.0, with a confidence interval of 83.56–86.44, showing the highest energy consumption across all scenarios.

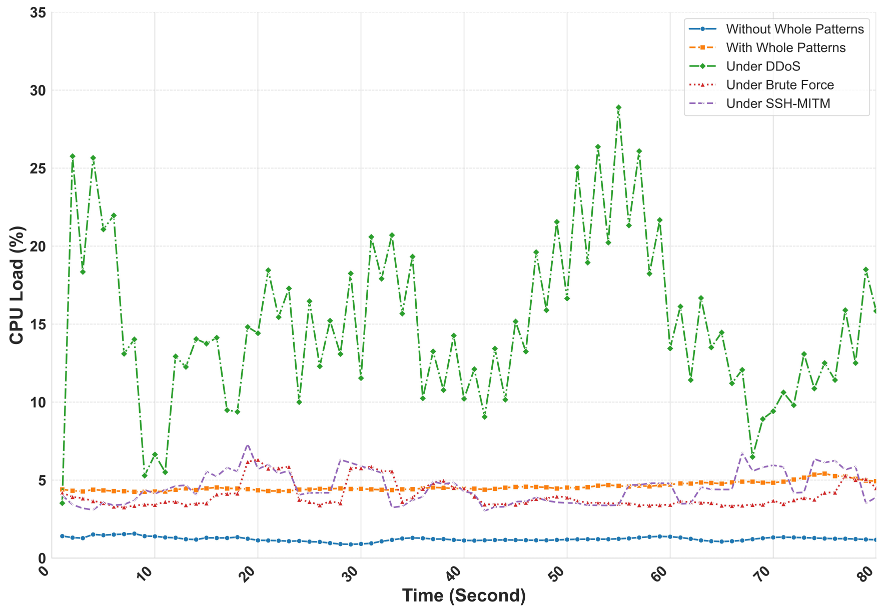

Figure 5.20 CPU load for a smart city application with combination patterns, without combination patterns, and three attacks: DDoS, MITM, and brute-force.

Fig. 5.22 shows the outcomes achieved across various implementations of the combined patterns. The examination of energy consumption during DDoS, brute-force, and MITM attacks reveals an increase when all patterns are introduced. Nevertheless, it is imperative to underscore that a statistically significant variance in energy consumption exists between scenarios with/without all patterns. Furthermore, there is a statistically significant variance in energy consumption under brute-force attacks in the smart home and in smart home and healthcare during MITM attacks and all patterns during DDoS attacks.

Table 5.10 reports the P-values of the Mann-Whitney U test and the Cliff's δ effect size for CPU usage evaluation. The table compares the average CPU usage levels observed during three distinct attack scenarios (MITM, brute-force, and DDoS) with/without security patterns.

According to the P-value in Table 5.10, we reject H^5_{all} for all applications, as there is a statistically significant difference between the average amount of CPU usage under different conditions with/without a combination of patterns in three applications.

According to the P-value for all applications under the MITM attack, we cannot reject H^6_{all} for all applications. The analysis indicates no statistically significant difference in CPU usage by $P_{all} - MITM$.

According to the P-value for all applications under the brute-force attack, we cannot reject H^7_{all} for smart home and healthcare applications. The data reveals no significant disparity in

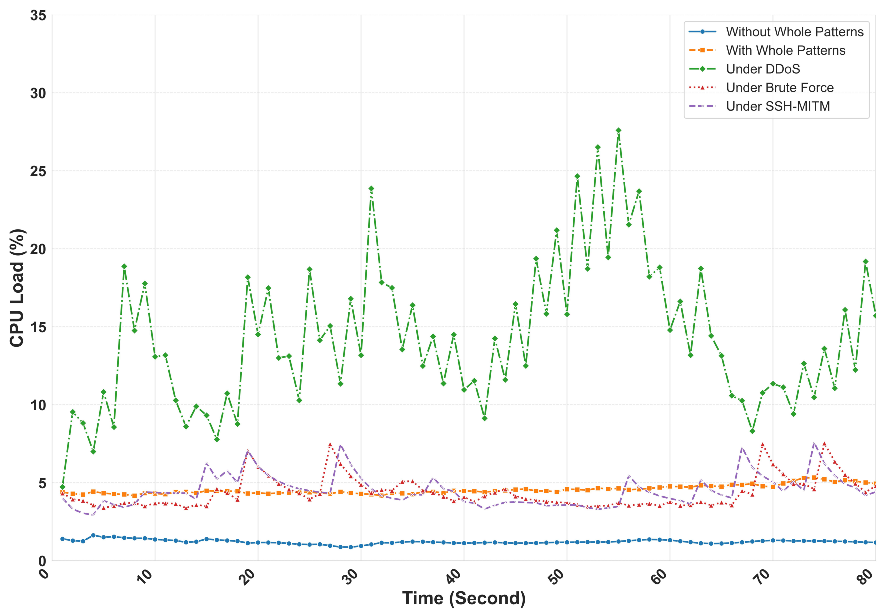

Figure 5.21 CPU load for a healthcare application with combination patterns, without combination patterns, and three attacks: DDoS, MITM, and brute-force.

Table 5.9 P-values, Cliff's δ effect sizes, and Confidence Intervals (CI) for average energy consumption under MITM, brute-force, and DDoS attacks.

Version	**SmartHome**			**SmartCity**			**HealthCare**		
	P-Value	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**
P_0 vs. P_{all}	0.0001	1.0	48.35–51.65	0.0001	1.0	58.45–61.55	0.0001	1.0	68.61–71.39
P_{all} vs. P_{all}-MITM	0.0001	0.893	53.89–56.11	0.0001	1.0	63.56–66.44	0.3953	1.0	73.56–76.44
P_{all} vs. P_{all}-BF	0.1096	0.347	58.89–61.11	0.3173	0.210	68.56–71.44	0.1428	0.529	78.56–81.44
P_{all} vs. P_{all}-DDoS	0.0001	1.0	63.89–66.11	0.0001	0.1	73.56–76.44	0.0001	1.0	83.56–86.44

CPU usage by $P_{all} - BF$. Moreover, we also reject H^7_{all} for smart city application.The data reveals a significant disparity in CPU usage by $P_{all} - BF$.

For the case of the DDoS attack, we reject H^8_{all} for the smart city and healthcare applications. A careful analysis affirms that statistically significant variation is found in the CPU usage by $P_{all} - DDoS$. However, we cannot reject H^8_{all} for the smart home application as a statistically significant variation is not found in the CPU usage by $P_{all} - DDoS$.

According to the effect sizes and confidence intervals presented in Table 5.10, the analysis illustrates the impact of MITM, brute-force, and DDoS attacks on CPU usage across the edge gateways.

For smart home, the effect size for P_0 vs. P_{all} is 0.813, indicating a substantial increase in CPU usage between baseline and attack-inclusive conditions, with a confidence interval of 39.32–40.68, reflecting stable variability. Under MITM attacks (P_{all} vs. P_{all}-MITM), the effect size decreases to 0.369, suggesting a moderate practical significance, with the confidence

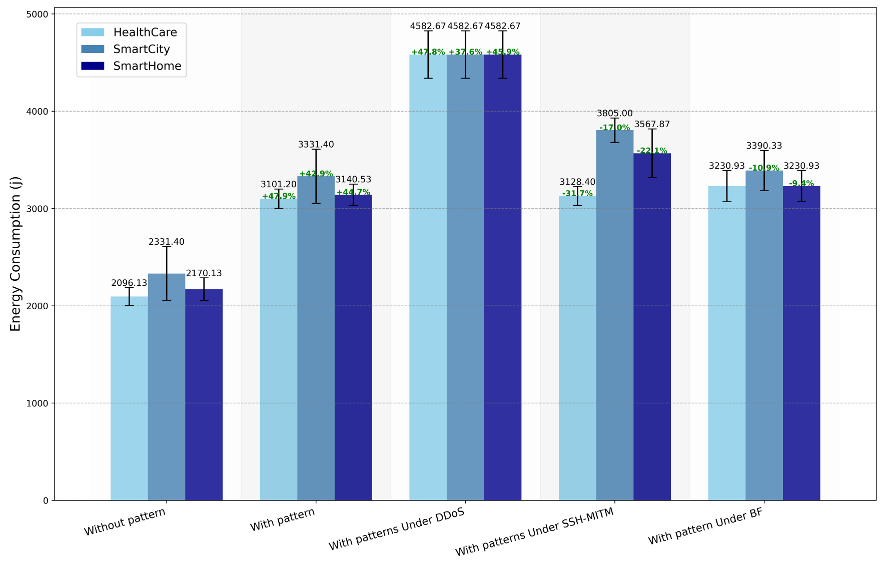

Figure 5.22 CPU usage for smart home, smart city, and healthcare applications with combination security patterns, without these patterns, and under three attack types: DDoS, MITM, and brute-force.

interval
(44.01–45.99) indicating consistent estimates. The effect sizes for brute-force and DDoS attacks are 0.458 and 0.982, respectively, with confidence intervals of 49.01–50.99 and 54.01–55.99, showing higher resource demands under DDoS scenarios.

For smart city, the effect size for P_0 vs. P_{all} is also 0.813, with a confidence interval (49.21–50.79) indicating a significant increase in CPU usage during attack-inclusive conditions. Under MITM attacks, the effect size is 0.458, reflecting moderate practical significance, with the confidence interval (54.02–55.98) showing increased variability. For brute-force and DDoS attacks, the effect sizes are 0.547 and 1.0, with confidence intervals of 59.01–60.99 and 64.02–65.98, respectively, highlighting the substantial impact of DDoS attacks on CPU resources.

In health care, the effect size for P_0 vs. P_{all} is 0.804, indicating a significant increase in CPU usage with a confidence interval of 59.18–60.82. For MITM attacks, the effect size decreases significantly to 0.031, suggesting minimal practical significance, with the confidence interval (64.04–65.96) showing moderate variability. Under brute-force and DDoS attacks, the effect sizes are 0.293 and 0.804, respectively, with confidence intervals of 69.02–70.98 and 74.01–75.99, showing the significant impact of DDoS attacks on CPU resources.

Fig. 5.23 illustrates the results obtained for all implementations of the combined patterns. The analysis of CPU usage during DDoS, brute-force, and MITM attacks demonstrates an increase when all patterns are introduced. However, it is essential to emphasize that there

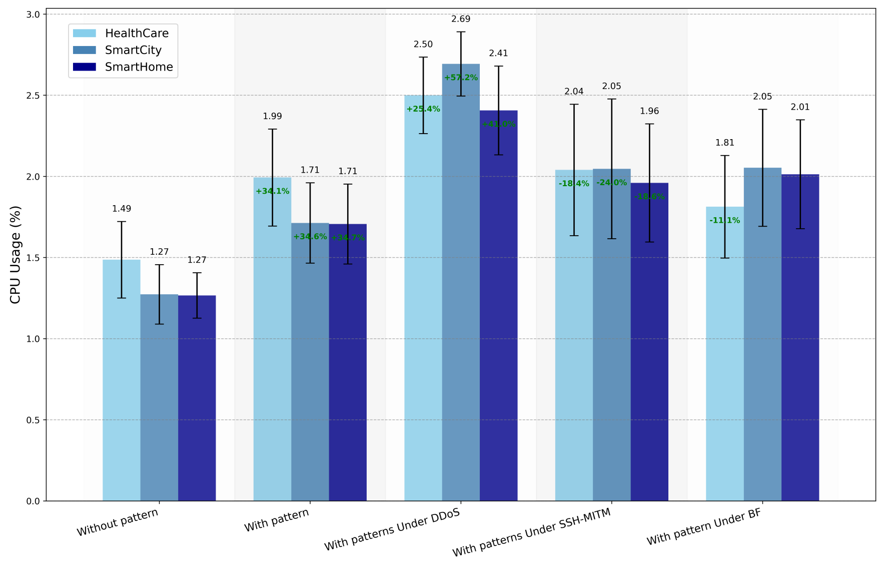

Figure 5.23 CPU usage for smart home, smart city, and healthcare applications with combination security patterns, without these patterns, and under three attack types: DDoS, MITM, and brute-force.

is a statistically significant variance in CPU usage between scenarios with and without all patterns. Moreover, there is no statistically significant variance in CPU usage during the brute-force attack in the smart city or during the DDoS attack in the smart city or healthcare.

Finding 4:

Our results indicate all patterns (when combined) in mitigating BF, MITM, and DDoS attacks. Importantly, we observe a discernible impact on CPU usage and energy consumption, both in the attack scenarios and with and without all patterns with statistically significant differences.

5.6.5 Experimental Findings of Dynamic Selection Security Patterns

To address RQ2, this section presents the experimental results for our proposed DRL-based dynamic pattern selection, designed to select security patterns dynamically in response to real-time network threats at the edge. Fig. 5.24, 5.25, and 5.26 present the average reward over 25,000 episodes across five runs of the DRL-based IDS for various ϵ values (the exploration rate), trained to detect multiple attack types (DDoS, DoS Hulk, DoS Slowloris, DoS GoldenEye).

Figure 5.24 ($\epsilon = 0.6$), the model achieves stable rewards around the 90 episodes. This bal-

Table 5.10 P-values, Cliff's δ effect sizes, and Confidence Intervals (CI) for average CPU usage under MITM, brute-force, and DDoS attacks.

Version	**SmartHome**			**SmartCity**			**HealthCare**		
	P-Value	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**	**P-Value**	**Effect Size**	**CI (95%)**
P_0 vs. P_{all}	0.0012	0.813	39.32–40.68	0.0001	0.813	49.21–50.79	0.0002	0.804	59.18–60.82
P_{all} vs. P_{all}-MITM	0.8014	0.369	44.01–45.99	0.3400	0.458	54.02–55.98	0.1770	0.031	64.04–65.96
P_{all} vs. P_{all}-BF	0.3400	0.458	49.01–50.99	0.0114	0.547	59.01–60.99	0.9044	0.293	69.02–70.98
P_{all} vs. P_{all}-DDoS	0.3400	0.982	54.01–55.99	0.0001	1.0	64.02–65.98	0.0002	0.804	74.01–75.99

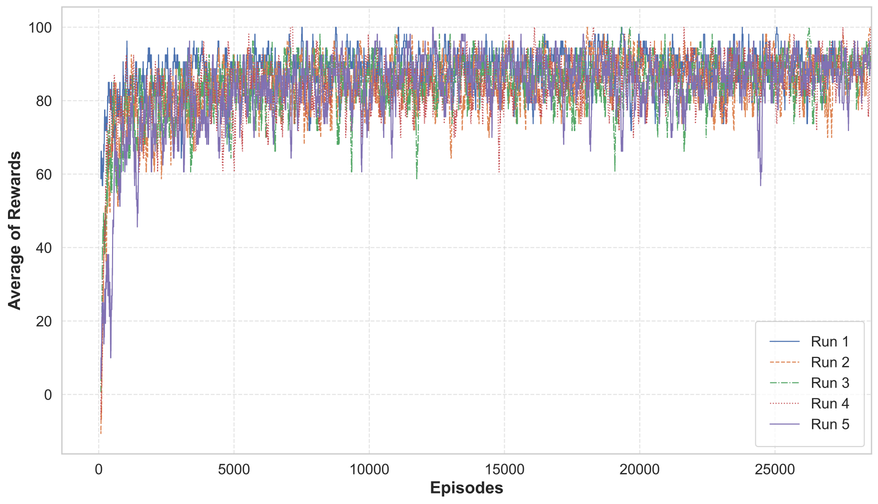

Figure 5.24 Average of rewards over 25,000 episodes with $\epsilon = 0.6$ for five runs

ance of exploration enables consistent attack detection and reliable pattern selection [244], favoring **OOC** for external threats and **PZH** for internal control. Figure 5.25 ($\epsilon = 0.7$) shows slightly faster stabilization, as the higher exploration rate aids in handling varied attack types. This adaptability translates to an agile pattern-selection process that can flexibly choose between **OOC** and **PZH** based on threat conditions. Figure 5.26 ($\epsilon = 0.9$) exhibits rapid initial learning and quick reward stabilization. High exploration during training equips the model to detect and respond to novel attacks, albeit with occasional variability in pattern selection. This configuration suits dynamic environments where frequent adjustments are needed. In addition, lower epsilon values (e.g., 0.6) are optimal for predictable attack patterns, while higher values (e.g., 0.9) enhance adaptability in dynamic settings. Across all settings, the trained model's robust detection ensures effective threat mitigation. Table 5.11 presents the performance metrics of our DRL-based IDS, highlighting its effectiveness in detecting attacks. The model achieves an **accuracy** of 0.92, demonstrating its ability to accurately classify normal and malicious traffic. An **F1 score** of 0.91 indicates a strong balance between precision and recall, suggesting that the model successfully minimizes false positives and false negatives in its detection. The **precision** of 0.92 signifies that the model

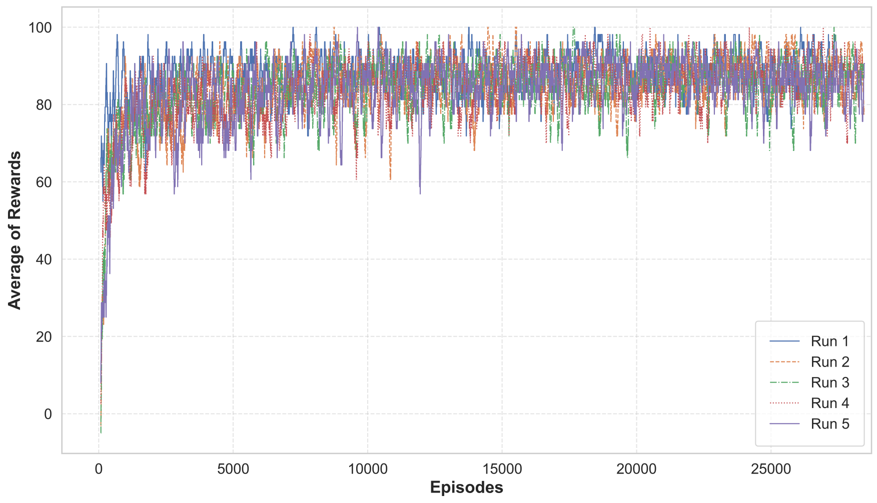

Figure 5.25 Average of rewards over 25,000 episodes with $\epsilon = 0.7$ for five runs

accurately identifies true positives, meaning it effectively distinguishes actual attacks from benign activity. Additionally, a **recall** of 0.93 reflects the model's capability to detect many accurate attacks, minimizing missed detections.

5.6.6 Accuracy

Table 5.11 Performance Metrics of the DRL-based IDS Model

Accuracy	**F1 Score**	**Precision**	**Recall**
0.92	0.91	0.92	0.93

5.6.7 Resource Consumption

Fig. 5.27 presents the CPU and Memory Usage alongside Energy Consumption (J) for three conditions: No Attack (baseline), OOC, and PZH. In the baseline state, CPU usage is 40.8%, memory usage is 67.1%, and energy consumption is steady at 2.04 J. Under the OOC pattern, activated for high-risk outbound threats, CPU and memory usage increase to 60.3% and 86.7%, respectively, reflecting the added computational demand for threat mitigation. Energy consumption rises to 2.08 J, indicating a slight increase in energy usage under high computational load. The PZH pattern, designed for internal control, shows CPU and memory usage similar to the baseline (40.6% and 66.8%), with energy consumption maintaining the same level as OOC at 2.08 J, reflecting consistent energy utilization across operational conditions.

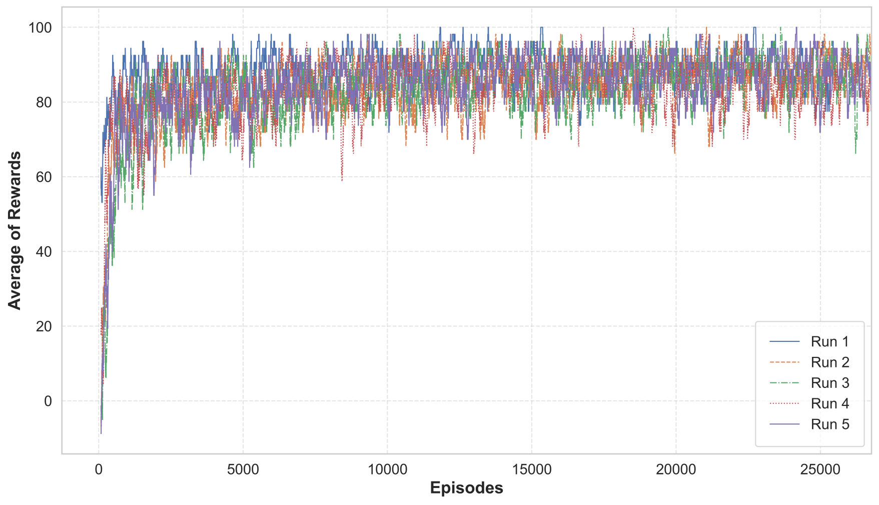

Figure 5.26 Average of rewards over 25,000 episodes with $\epsilon = 0.9$ for five runs

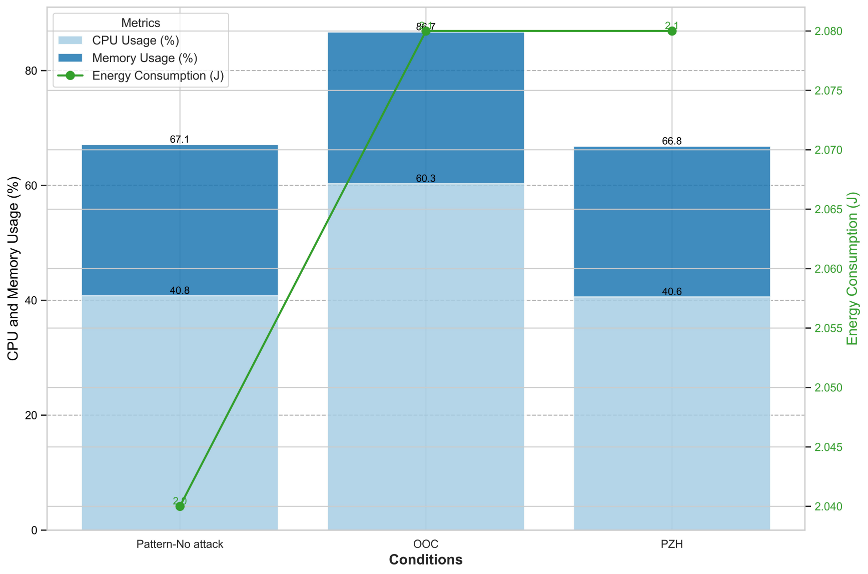

Figure 5.27 CPU usage, memory demands, and energy consumption under No Attack, OOC, and PZH conditions, showing stable energy consumption despite varying CPU usage and memory demands.

5.6.8 Dynamic Pattern Selection in Threat Response

The log output 5.28 illustrates the DRL-based IDS adaptive security system's effectiveness in mitigating real-time threats. Initially, the system enforces the OOC pattern to block unauthorized inbound traffic, establishing a secure baseline. Upon detecting anomalies, such as a sudden increase in CPU usage, the system dynamically activates targeted security patterns in response to specific threats. For instance, Network Throttling is employed to manage high-demand scenarios, while IP Blocking is initiated to counteract DDoS attacks effectively. The PZH pattern further strengthens internal security by enforcing access control and mitigating insider threats. This dynamic, pattern-driven approach exemplifies the system's resilience by selecting optimal security measures based on real-time network conditions, enhancing protection against diverse threats. The DRL-based IDS dynamic pattern selection

```
2024-10-25 12:40:13,543 - INFO - Repeated pattern execution: General system
    ↪ state - No pattern executed
2024-10-25 12:40:13,894 - INFO - Predicted action: [[1. 0.]]
2024-10-25 12:40:15,987 - INFO - CPU Usage: 46.9\%, Memory Usage: 26.4\%,
    ↪ Energy Usage: 2.04j
2024-10-25 12:40:16,996 - INFO - Repeated pattern execution: General system
    ↪ state - No pattern executed
2024-10-25 12:40:17,499 - INFO - Predicted action: [[1. 0.]]
2024-10-25 12:40:19,093 - INFO - CPU Usage: 13.2\%, Memory Usage: 26.4\%,
    ↪ Energy Usage: 2.04j
2024-10-25 12:40:26,230 - INFO - CPU Usage:  60.3%, Memory Usage:  26.4%, Energy Usage:
    2.08j, CPU Temperature:  50.0°C
2024-10-25 12:40:26,230 - INFO - CPU usage exceeded threshold:  60.3%
2024-10-25 12:40:26,230 - INFO - DDoS attack detected based on request threshold.
2024-10-25 12:40:26,230 - INFO - DDoS detected.  Executing Pattern - Blocking IP.
2024-10-25 12:40:26,231 - INFO - Detected attacker IP: 192.168.1.100
2024-10-25 12:40:26,231 - INFO - Blocking IP 192.168.1.100 via iptables
```

Figure 5.28 System Monitor Log Output

achieves a 45% CPU usage and 1.5 J energy consumption, significantly reducing both metrics compared to the 65% CPU usage and 2.4 J energy consumption of static OOC/PZH patterns under DDoS. This illustrates the DRL-based IDS's efficiency in dynamically managing resources to mitigate attacks.

5.6.9 Comparative Analysis of Static vs. Dynamic Patterns

Table 5.12 highlights the resource utilization of static configurations (OOC and PZH) compared to the dynamic DRL-based IDS under attack scenarios. OOC and PZH exhibit high CPU usage (65.2% and 63.8%, respectively) due to their inability to adapt. In comparison, the DRL-based IDS slightly lowers CPU usage to 60.3% by dynamically selecting patterns in real-time. Regarding energy consumption, the DRL-based IDS achieves significant efficiency, reducing usage to 2.08 J compared to 2.4 J and 2.3 J for OOC and PZH. These results demonstrate the advantages of dynamic adaptability, balancing real-time threat mitigation with optimized resource usage. Although the DRL-based IDS shows peak CPU usage during active attacks, its efficiency ensures stability and scalability for diverse at the edge gateway.

Finding 5:

Our findings highlight the efficiency of the DRL-based IDS in dynamically managing resources and mitigating threats. Under DoS Hulk, DoS Slowloris, DDoS, and DoS GoldenEye attacks, the dynamic approach reduced CPU usage to 45% and energy consumption to 1.5 J, compared to the static OOC/PZH patterns' 65% CPU usage and 2.4 J energy consumption. This adaptive selection of patterns ensures security and optimizes resource usage, underscoring the system's capability to maintain robust performance under varying conditions.

5.7 Discussion

In selecting IoT security patterns, developers encounter a complex trade-off between security, energy, and CPU usage. An effective IoT system should proactively predict risks and adapt patterns based on context, optimizing resource utilization while maintaining security. Applying all security patterns statically is resource-intensive; thus, context-specific pattern selection is more efficient. This challenge is analogous to the "Robot in the Grid World" problem, where the software aims to optimally balance security, CPU, and energy. Our proposed DRL-based system dynamically selects patterns, effectively mitigating threats such as DDoS while ensuring resource efficiency and robust performance.

In addition, while static evaluations of security patterns have inherent limitations in addressing cyber threats' dynamic and evolving nature, integrating the DRL-based IDS in our study overcomes this challenge. By dynamically selecting and reconfiguring real-time patterns, the system ensures robust defense mechanisms that remain effective even as new threats emerge. This adaptability makes the proposed approach particularly relevant for real-world edge gateways.

Our findings demonstrate that the DRL-based approach significantly enhances adaptive security by dynamically selecting patterns in response to real-time network conditions, effectively balancing security and resource demands. Additionally, the initial findings in this chapter offer developers and practitioners valuable insights into implementing IoT security patterns. For a detailed

overview, Table 5.13 presents the effects of six IoT security patterns on security, energy

Table 5.12 Comparison of Static and Dynamic Patterns Under Attacks

Metric	Static (OOC)	Static (PZH)	Dynamic (DRL-based IDS)
CPU Usage (%)	65.2	63.8	**60.3**
Energy Consumption (J)	2.4	2.3	**2.08**

efficiency, and CPU usage. To situate this work within the broader IoT security landscape, advancements in blockchain-assisted security protocols are worth noting. Shahidinejad et al. [245] review blockchain-based authentication mechanisms and Session Key Generation Protocols (SKGPs) for IoT systems, highlighting lightweight and scalable solutions for resource-constrained edge gateway. While our DRL-based IDS dynamically selects security patterns to optimize resource efficiency and mitigate threats in real-time, we did not integrate blockchain due to its computational overhead and complexity in resource-constrained edge gateway. However, combining adaptive pattern selection with distributed ledger technologies offers a promising research direction for enhancing authentication, trust, and traceability in IoT security.

Security: IoT-edge applications face significant security risks due to limited protocols, authentication challenges, and privacy issues, primarily as they rely on communication between multiple devices and the cloud. Many edge devices lack the computational power and memory for robust security, making them vulnerable to attacks such as DDoS. While existing IoT security patterns offer some protection, more advanced solutions are needed.

Energy consumption and CPU usage: Edge devices in IoT are typically compact and resource-constrained, with limited energy and processing capacity. These devices often rely on cost-effective hardware, necessitating a careful balance between functionality and resource use. Our experiments reveal that applying security patterns influences energy consumption and CPU usage, highlighting developers' importance in selecting appropriate patterns based on available resources. As shown in Table A.1 (Appendix), the guidelines help choose the

Table 5.13 Impact of security patterns on security, energy efficiency, and CPU usage.

Context Problem	**Pattern**	**Security**	**Energy**	**CPU usage**
Security, No Open Ports, Firewalls, Low Energy	OOC	Improved	Increased	Increased
Centralized Access Control, User Control, Trust	PZH	Not Effective	Increased	Increased
Explicit Allowance, Flexibility, Trust, Simplicity, Completeness	WL	Improved	Increased	Increased
Flexibility, Explicit Blocking, Simplicity, Outdated Entries	BL	Improved	Increased	Increased
Secure Communication, Access Control	SSN	Improved	Increased	Increased

The TCP pattern is not mentioned because this pattern is generally used along with WL and BL patterns, and we did not analyze the impact of an isolated implementation of this pattern.

most effective patterns for IoT security.

5.8 Threats to validity

Empirical research inevitably encounters issues related to the validity of findings. In light of this, the present section seeks to identify and discuss possible threats to our research's validity, per the recommendations of [207].

5.8.1 Threats to Internal Validity

Threats to internal validity concern factors internal to our study that could have impacted our results [246]. In evaluating the internal validity of our empirical study, we recognize potential limitations. First, testing a single implementation of security patterns in three IoT applications may limit the generalizability of results, as different implementations could yield varied outcomes in terms of security, CPU usage, and energy efficiency. Additionally, real-time fluctuations in IoT device performance, especially with Raspberry Pis, can impact the accuracy of CPU usage and security efficacy measurements. To address these, we ran each experiment fifteen times to average anomalies. We also ensured unbiased security algorithm configurations by aligning with existing research standardized settings, enhancing findings' reliability across different attack scenarios.

5.8.2 Threats to External Validity

These threats concern the generalization of our findings [246]. To enhance the validity and reliability of our findings, we conducted tests on multiple edge gateways. Additionally, we provided a detailed description of our experimental setup to account for potential influences from hardware, sensors, and environmental factors. We have also made our source codes publicly accessible for further research [247]. These measures aim to bolster our study's credibility, transparency, and reproducibility, aiding practitioners and researchers in validating our findings.

5.9 Chapter Summary

This research examined the impact of six IoT security patterns—PZH, OOC, BL, WL, and SSN—on the edge gateway's energy consumption, CPU usage, and CPU load. While these patterns enhance security, they significantly increase energy consumption, CPU usage, and CPU load. Our findings indicate that while some patterns, such as OOC and WL, offer strong security measures, they impose a considerable computational overhead. Conversely, SSN demonstrates a balanced trade-off between security and resource consumption, making it a viable option for the resource-constrained edge gateway. Additionally, our analysis shows that CPU-intensive patterns, e.g., PZH, may lead to bottlenecks in real-time processing, emphasizing the need for adaptive security mechanisms that optimize performance dynamically. In addition, challenges related to scalability and response time remain unresolved. To address these, we developed a DRL-based IDS that dynamically selects security patterns based on real-time network conditions, optimizing security and resource efficiency. However, further research is needed to evaluate its performance in large-scale systems. The study also highlights the need for comprehensive security patterns addressing data privacy, fault tolerance, and device authentication, which were beyond the scope of this work.

CHAPTER 6 ARTICLE 4: A DYNAMIC SECURITY PATTERN SELECTION FRAMEWORK USING DEEP REINFORCEMENT LEARNING

6.1 Article Metadata

- **Title:** A Dynamic Security Pattern Selection Framework Using Deep Reinforcement Learning
- **Authors:** Saeid Jamshidi, Amin Nikanjam, Kawser Wazed Nafi, Foutse Khomh
- **Affiliations:**
  - SWAT Laboratory, Polytechnique Montréal, Quebec, H3T 1J4, Canada
  - Huawei Distributed Scheduling and Data Engine Lab, Canada
- **Submitted Date:** 10 March 2025
- **Accepted Date:** 14 May 2025
- **Conference:** *2025 IEEE World Congress on Services (SSE 2025)*

6.2 Chapter Overview

The rapid expansion of IoT networks has brought transformative benefits across various domains, introducing significant security challenges, especially in resource-constrained edge gateways. This chapter proposes an innovative IDS powered by DRL to detect and mitigate network threats dynamically by selecting IoT security patterns. Leveraging adaptive IoT security patterns, the system efficiently addresses diverse attack scenarios(e.g., DDoS, DoS GoldenEye, DoS Hulk, and Port Scanning). Achieving an average detection accuracy of 97%, the system demonstrates rapid response times and efficient resource utilization, making it well-suited for edge gateways. The experimental evaluations validate the proposed model's ability to enhance security while optimizing CPU and memory usage, reducing energy consumption, and lowering carbon emissions. Furthermore, its adaptability to evolving cyber threats and alignment with green computing principles highlight its potential to support secure and sustainable IoT networks.

6.3 Study Design

The IoT has emerged as a transformative technology, driving advancements in smart cities, healthcare, and industrial automation. However, the rapid expansion of IoT networks has also heightened their vulnerability to cybersecurity threats. Resource-constrained IoT devices are particularly susceptible to network-based attacks, e.g., DDoS [10, 205], Port Scanning [248], DoS Hulk [249,250], and Goldeneye [251,252]. These attacks disrupt IoT operations, increase energy consumption, and waste resources, posing significant challenges to the sustainability and reliability of IoT systems [253].

In addition, researchers have proposed security patterns as modular solutions for recurring IoT security issues to address these challenges, e.g., PZH, OOC, BL, WL [1], and SSN patterns [94]. These patterns provide structured approaches to securing IoT systems by addressing their inherent heterogeneity and complexity [93]. However, previous work has primarily focused on theoretical discussions without evaluating these patterns under real-world attack scenarios. In our previous study [254], we assessed these patterns under real-time attack conditions, analyzing their impact on energy consumption and CPU usage. Despite this, the patterns were treated as static solutions, leaving their dynamic networks unexplored. This study introduces a novel framework that employs DRL to fill this gap and enable dynamic security pattern selection. This approach ensures real-time adaptation of patterns to diverse attack scenarios, improving IoT security while optimizing resource utilization.

This chapter proposes a DRL-based IDS that detects and mitigates network attacks using well-suited resources. Unlike traditional IDS approaches, the proposed system intelligently selects and activates security patterns for specific threats, enhancing detection accuracy and well-suited CPU usage, memory usage, energy consumption, and carbon emissions. Additionally, this study aligns with the United Nations Sustainable Development Goals (SDGs), particularly SDG 9 (Industry, Innovation, and Infrastructure), SDG 11 (Sustainable Cities and Communities), and SDG 13 (Climate Action) [255].

The purpose method is experimentally validated under real-world attack scenarios)(e.g., DDoS, DoS GoldenEye, DoS Hulk, and Port Scanning). Moreover, evaluations focus on critical metrics, e.g., detection accuracy, IDS detection response time, energy consumption, memory usage, and carbon emissions. Findings demonstrate the system's robustness and efficiency, achieving high detection rates and resource optimization. This study makes the following key contributions:

- **Dynamic and Adaptive Security Framework:** It proposes an intelligent IDS that leverages DRL for real-time detection and adaptation to network threats, incorporating the dynamic selection of six distinct security patterns (OOC, BL, WL, SSN, and PZH) to enhance IoT security.

- **Energy-Efficient Mitigation:** Incorporation of green computing principles to optimize resource usage, including energy consumption, CPU usage, and carbon emissions, while maintaining high detection accuracy.

6.4 Security Pattern Selection Using DRL-Based IDS

The proposed DRL-based IDS focuses on detecting attacks to ensure IoT system security. The model dynamically analyzes network traffic to identify specific attacks and selects the most appropriate security patterns to neutralize threats. This section outlines the model's key components, including states, actions, and rewards, all of which are designed explicitly for security purposes. The overall architecture of the proposed DRL-based IDS is illustrated in Figure 6.1. It demonstrates the process flow, beginning with observing network states and computing rewards, followed by the agent selecting optimal actions and security patterns for threat mitigation. This flow ensures an intelligent, adaptable, and efficient defense mechanism for the edge.

6.4.1 State, Action, and Reward Definitions

State (s_t): The state represents the edge's security condition at time t, captured as a feature vector $s_t \in \mathbb{R}^n$. These features are derived from network traffic data and include:

- Packet-level metrics: Packet counts, rates, and sizes.
- Flow-level metrics: Flow durations and inter-arrival times.
- Protocol flags: SYN, ACK, and FIN indicators.

The state vector encapsulates real-time network behavior, enabling the DRL agent to detect anomalies indicative of attacks. **Action (a):** The action a corresponds to traffic classification and specific attack type. The action space is defined as:

$$a \in \{0, 1, 2, \ldots, k\}$$

Where $a = 0$ indicates normal traffic, and $a > 0$ denotes specific attack types, such as:

- $a = 1$: DDoS.
- $a = 2$: Port Scanning.

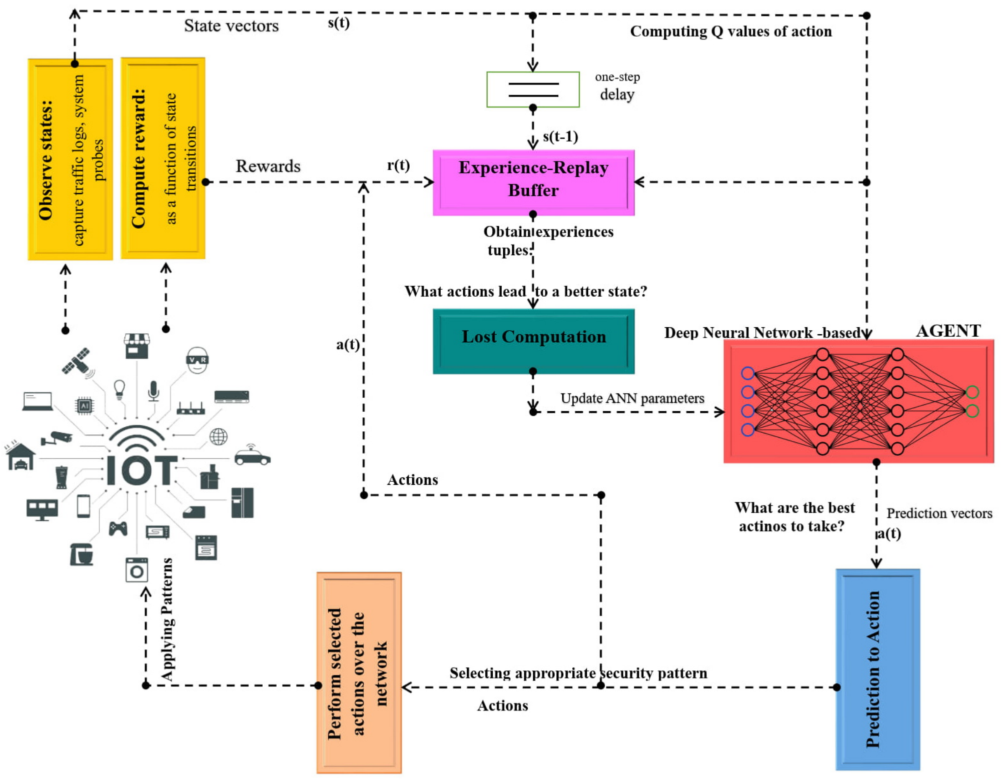
State vectors
s(t)
Computing Q values of action
one-step delay
s(t-1)
Observe states: capture traffic logs, system probes
Compute reward: as a function of state transitions
Rewards
r(t)
Experience-Replay Buffer
Obtain experiences tuples
What actions lead to a better state?
a(t)
Lost Computation
Deep Neural Network -based
AGENT
Update ANN parameters
IOT
Actions
What are the best actinos to take?
Prediction vectors
a(t)
Applying Patterns
Perform selected actions over the network
Selecting appropriate security pattern
Actions
Prediction to Action

Figure 6.1 Process flow of the proposed DRL-based IDS for attack detection and security pattern selection. The system observes IoT network states, computes rewards, and selects optimal security patterns for threat mitigation.

- $a = 3$: DoS Hulk.
- $a = 4$: DoS GoldenEye.

The optimal action a_t for state s_t is determined by maximizing the Q-value:

$$a_t = \arg\max_a Q(s_t, a) \tag{6.1}$$

This ensures that the system accurately identifies the type of attack or confirms that the traffic is regular.

Reward (R): The reward function evaluates the effectiveness of the selected action in mitigating the identified attack while minimizing disruption to the network. It is defined as:

$$R(s_t, a_t) = w_1 \cdot M(s_t, a_t) - w_2 \cdot C(s_t, a_t) \tag{6.2}$$

Where:

- $M(s_t, a_t)$: Mitigation effectiveness, measuring the percentage of the threat neutralized by the selected action:

$$M(s_t, a_t) = \frac{\text{Threat Neutralized}}{\text{Total Threat Impact}} \tag{6.3}$$

- $C(s_t, a_t)$: Cost of deploying the action, representing the system's resource usage.
- w_1, w_2: Weights prioritise mitigation effectiveness and resource efficiency.

6.4.2 Security Pattern Selection

Once an attack is identified ($a_t > 0$), the model selects the most appropriate security pattern p_k from a predefined set $P = \{p_1, p_2, \ldots, p_m\}$. Each pattern addresses specific attacks and ensures effective mitigation:

- **OOC:** Mitigates DDoS attacks by restricting outgoing traffic, preventing resource exhaustion.
- **BL:** Blocks malicious IP addresses, effectively countering reconnaissance attacks, e.g., Port Scanning.
- **WL:** Permits only authorized traffic, ensuring secure communication during DoS Hulk and DoS GoldenEye attacks.
- **SSN:** Protects IoT sensor data by enforcing encryption and secure communication protocols.
- **PZH:** Establishes isolated secure zones to contain compromised devices and prevent lateral attacks.

The optimal pattern p_t^* is selected based on a reward evaluation:

$$p_t^* = \arg\max_{p_k \in P} R(s_t, p_k) \tag{6.4}$$

Where $R(s_t, p_k)$ evaluates the pattern's effectiveness in neutralizing the attack while maintaining system stability and minimizing resource consumption.

6.4.3 Attack Mitigation Process

The process begins with the DRL agent observing the network state s_t and computing Q-values for all possible actions. Based on the selected action a_t, the system classifies the traffic as usual or identifies the specific attack type. If an attack is detected, the system evaluates the predefined security patterns using the reward function and deploys the one that maximizes threat mitigation. For example:

- A DDoS attack ($a_t = 1$) triggers the OOC, which restricts outgoing traffic to reduce resource exhaustion.
- A Port Scanning attack ($a_t = 2$) activates the BL to block malicious IPs effectively.
- DoS Hulk ($a_t = 3$) deploy the WL to prioritize legitimate traffic.

This security-focused approach ensures real-time adaptability and precise threat mitigation. The proposed DRL-based IDS enhances edge security by continuously optimizing the reward function while maintaining efficiency and stability.

6.4.4 Dynamic Threat Detection Using DRL-based IDS

The proposed algorithm, detailed in Algorithm 6, leverages a DRL-based IDS to dynamically detect edge attacks by analyzing the network state s_t. The state is represented as a high-dimensional feature vector derived from traffic metrics (e.g., packet counts, rates, and sizes), flow-level statistics (e.g., average flow duration and inter-arrival times), and protocol indicators (e.g., SYN, ACK, and FIN flags). This representation enables the DRL agent to model and identify anomalies indicative of potential attacks.

Algorithm 6: Advanced DRL-based IDS for Attack Detection

```
Input  : Edge state $s_t$, Action space $A$, Security patterns $P$, DRL model $\mathcal{M}$
Output: Optimal security pattern $p_t^*$
1  Initialize: $s_0$, DRL model parameters $\theta$, replay buffer $\mathcal{D}$
2  while System is active do
3      Observe current network state $s_t$ based on:
4          Traffic Metrics: Packet counts, rates, sizes
5          Flow Metrics: Average flow duration, inter-arrival times
6          Protocol Indicators: Flags
7      Compute Q-values $Q(s_t, a; \theta)$ for $a \in A$
8      Select action $a_t$ based on:
           $a_t \leftarrow \begin{cases} \text{random action}, & \epsilon \text{ probability} \\ \arg\max_{a \in A} Q(s_t, a; \theta), & \text{otherwise} \end{cases}$
9      if $a_t = 0$ then
10         Classify traffic as normal; continue monitoring
11     else
12         Identify attack type corresponding to $a_t$
13         Compute impact: $I(s_t, a_t) = \frac{\text{Severity of attack}}{\text{Resilience of system}}$
14         Call Pattern Selection$(a_t, s_t, P, I(s_t, a_t))$
15         Observe reward $r_t$ and next state $s_{t+1}$
16         Store transition $(s_t, a_t, r_t, s_{t+1})$ in $\mathcal{D}$
17     end
18     Sample minibatch $(s, a, r, s')$ from $\mathcal{D}$
19     Update the DRL model by minimizing TD error:
           $\delta_t = r_t + \gamma \max_{a'} Q(s_{t+1}, a'; \theta) - Q(s_t, a_t; \theta)$
       Perform gradient descent to minimize:
           $\mathcal{L}(\theta) = \mathbb{E}_{\mathcal{D}} \left[\delta_t^2\right]$
20 end
```

The DRL agent computes Q-values $Q(s_t, a; \theta)$ for each possible action a in the action space $A = \{a_0, a_1, \ldots, a_k\}$, where a_0 represents standard traffic classification, and $a_1, a_2, \ldots, a_k$ correspond to specific attack types, i.e,. DDoS, Port Scanning, and DoS Hulk and DoS GoldenEye. Using an ϵ-greedy strategy, the agent balances exploration and exploitation to select the optimal action a_t:

$$a_t \leftarrow \begin{cases} \text{random action} & \text{(exploration probability } \epsilon\text{)} \\ \arg\max_{a \in A} Q(s_t, a; \theta) & \text{(otherwise, exploitation)} \end{cases}$$

The system continues monitoring if the action corresponds to normal traffic ($a_t = 0$). However, if an attack is detected ($a_t > 0$), the severity of the attack is quantified using the impact function:

$$I(s_t, a_t) = \frac{\text{Severity}}{\text{Resilience}}$$

Algorithm 7: Pattern Selection

```
   Input  : Attack type $a_t$, Network state $s_t$, Security patterns $P = \{p_1, p_2, \dots, p_m\}$,
            Predicted impact $I(s_t, a_t)$, Thresholds $T = \{T_1, T_2, \dots, T_m\}$
   Output: Optimal security pattern $p_t^*$
1  Initialize: $R_{\max} \leftarrow -\infty$, $p_t^* \leftarrow \emptyset$
2  for each $p_k \in P$ do
3      if $I(s_t, a_t) < T_k$ then
4          Skip $p_k$ as it does not meet the threshold
5          Continue to next pattern
6      end
7      Compute reward for pattern $p_k$:
           $$R(s_t, p_k) = w_1 \cdot M(s_t, p_k) + w_2 \cdot E(s_t, p_k)$$
8      Evaluate mitigation effectiveness:
           $$M(s_t, p_k) = \frac{\text{Threat Neutralized by } p_k}{\text{Total Threat Impact}}$$
9      Evaluate system stability impact $E(s_t, p_k)$
10     if $R(s_t, p_k) > R_{max}$ then
11         Update $R_{\max} \leftarrow R(s_t, p_k)$
12         Update $p_t^* \leftarrow p_k$
13     end
14 end
15 Deploy optimal pattern $p_t^*$ to mitigate the attack
```

This calculated impact guides the invocation of the selection pattern (Algorithm 7), which evaluates predefined security patterns to identify and deploy the optimal mitigation strategy p_t^*. Each pattern is assessed based on its reward, incorporating factors, e.g., mitigation effectiveness and system stability. Given the attack's impact, it is subject to a threshold check to ensure its applicability. The learning process is reinforced by storing transitions

(s_t, a_t, r_t, s_{t+1}) in a replay buffer $\mathcal{D}$, enabling the DRL agent to learn from past interactions. The agent updates its parameters by minimizing the Temporal Difference (TD) error:

$$\delta_t = r_t + \gamma \max_{a'} Q(s_{t+1}, a'; \theta) - Q(s_t, a_t; \theta)$$

The loss function for training the DRL model is defined as:

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathcal{D}}[\delta_t^2]$$

Algorithm 6 outlines the overall workflow, highlighting the interaction between attack detection and dynamic pattern selection. The proposed algorithm achieves robust, real-time threat mitigation by continuously learning and adapting while optimizing resource usage. This makes it highly suitable for securing the edge gateway.

6.5 Experimental setup

This section outlines the experimental methodology to evaluate the DRL-based IDS.

6.5.1 Research Questions:

RQ1: How can a DRL-based IDS dynamically select and deploy security patterns to mitigate specific attack types at the edge gateway?
This RQ investigates the ability of a DRL-based IDS to adaptively choose and apply appropriate security patterns, achieving effective threat mitigation at the edge gateway.
RQ2: How does integrating dynamic security pattern selection in a DRL-based IDS impact resource utilization and sustainability at the edge gateway?
This RQ investigates the impact of DRL-based IDS dynamic security measures on system efficiency, e.g., CPU, memory, energy efficiency, and carbon emission, while ensuring resource optimization.

6.5.2 Data Set Evaluation

We used the CICIDS2017 dataset [134], a widely recognized benchmark for evaluating IDS. This dataset includes traffic data representing various network attacks(e.g., DDoS, DoS GoldenEye, DoS Hulk, and Port Scanning). It contains 83 traffic features extracted over varying time windows, enabling a thorough performance assessment of IDS models. To conduct the evaluation, a subset of 23 critical features was selected [256] to effectively capture key patterns in network traffic while minimizing computational overhead.

6.5.3 Feature Selection

We focused on detecting cyber threats efficiently on edge gateways by selecting impactful traffic features and reducing computational load. Using CICFlowMeter [257], we extracted 83 features and identified 23 critical ones by analyzing their variance through the Cumulative Distribution Function (CDF) [256]. These features captured essential network behaviors to detect attacks, i.e., DDoS, DoS GoldenEye, DoS Hulk, and Port Scanning. We applied the Correlation-based Feature Selection (CFS) method to refine the feature set, eliminating redundant or highly correlated features. This approach enhanced independence and improved classification accuracy.

6.6 Experiments and Evaluation

To address RQ1, we conducted experiments to evaluate the proposed DRL-based IDS framework at the edge under real-time attack scenarios. This section details the experimental setup and methodology used for the evaluation.

6.6.1 Experiments and Evaluation

The experiments conducted for the proposed DRL-based IDS focus on evaluating its performance on the edge gateway under varying exploration probabilities (ϵ) during training. Figures 6.2, 6.3, and 6.4 illustrate the average reward trends across 40,000 episodes for three exploration settings ($\epsilon = 0.1, 0.2, 0.3$), demonstrating the system's learning progression and its ability to adapt to security-critical scenarios dynamically.

Figure 6.2 represents the performance with minimal exploration ($\epsilon = 0.1$), where the DRL agent primarily exploits its learned knowledge, achieving stable and high rewards. This setup is ideal for static environments with predictable attack patterns.

In contrast, Figure 6.3 depicts the balanced exploration- exploitation scenario ($\epsilon = 0.2$), where the model effectively mitigates both known and emerging attack types while preserving mitigation efficiency. Finally, Figure 6.4 shows the system's behavior under high exploration ($\epsilon = 0.3$), allowing the discovery of evolving threats but causing higher reward variability due to increased exploratory actions.

6.6.2 Real-World Testbed Evaluation

As Figure 8.5 illustrates, the testbed simulates a standard IoT edge network. It consists of six NodeMCUs, two Raspberry Pi 4 Model B units acting as edge nodes, a laptop device,

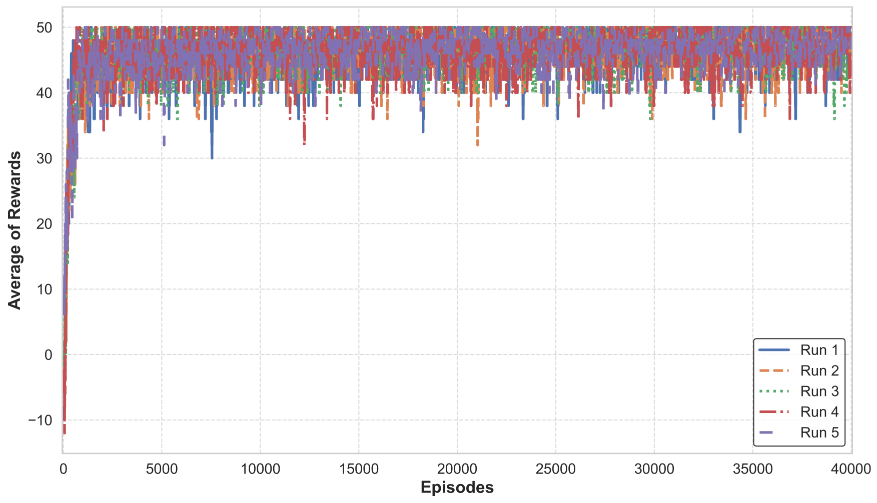

Figure 6.2 Average reward progression for $\epsilon = 0.1$. The model demonstrates exploitation-focused learning, achieving consistent and effective attack mitigation under stable scenarios.

a workstation, a router, a physical server, and a simulated attacker. Each Raspberry Pi 4 Model B is configured with 8GB of RAM and powered by a 1.5GHz 64-bit quad-core CPU, providing the computational resources necessary for replicating complex, real-world networks. The NodeMCUs and laptops serve as IoT devices distributed across two separate IoT domains, interfacing with the edge gateways for seamless communication and security processing.

The Raspberry Pi devices function as IoT edge gateways hosting the DRL-based IDS. At the same time, the physical server operates as the edge server in the network, running the DRL-based IDS. The simulated attacker, equipped with Kali Linux [185], generates a range of cyberattacks to test the system's robustness. Concurrently, the workstation monitors and captures network traffic using Wireshark to evaluate security. To evaluate the resource impact of the deployed security patterns, metrics such as energy consumption, CPU usage, memory usage, IDS response time, and carbon emissions are analyzed using the Mann-Whitney U Test. The null hypothesis for each test assumed no significant difference in these metrics between the baseline (pattern inactive) and active (pattern enabled) states. Rejecting the null hypothesis indicated that the specific security pattern statistically affected the corresponding metric. This analysis provided critical insights into the trade-offs between deploying effective security patterns and maintaining efficient system performance, ensuring a balance between robust threat mitigation and resource optimization.

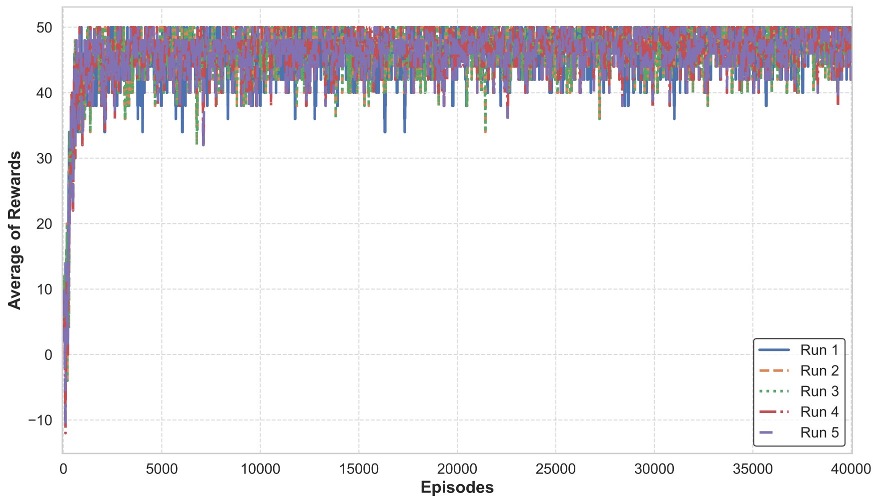

Figure 6.3 Average reward progression for $\epsilon = 0.2$. A balanced exploration-exploitation strategy enhances adaptability to evolving threats, maintaining robust mitigation performance.

6.6.3 Dynamic Attack Detection and Pattern-Based Mitigation

The DRL-based IDS dynamically adapts to real-time threats by detecting attacks and selecting the most effective security pattern for mitigation. Utilizing its DRL model, the system analyzes network traffic, classifies anomalies, and applies patterns based on the type of attack. This ensures efficient mitigation for DDoS, DoS Hulk, DoS GoldenEye, and Port Scanning.

```
Timestamp: 2024-12-14 15:03:54
System Status: NORMAL
CPU Usage: 11.8%, Memory Usage: 20.6%
Energy Consumption: 0.118 Joules, Carbon Emissions: 0.0118 Kg
IDS Response Time: 0.2478 seconds
Pattern Applied: None, Prediction Accuracy: N/A
The system is operating normally.
--------------------------------------------------
Timestamp: 2024-12-14 15:03:55
System Status: **ATTACK DETECTED (DDoS)**
CPU Usage: 53.7%, Memory Usage: 20.7%
Energy Consumption: 0.537 Joules, Carbon Emissions: 0.0537 Kg
IDS Response Time: 1.0024 seconds
Pattern Applied: OOC, Prediction Accuracy: 95%
**ALERT: DDoS attack detected. Initiating countermeasures.**
Mitigation: Blocking suspicious IP addresses.
```

Figure 6.6 The sample of system Logs: normal and attack condition

Figure 6.6 illustrates the functionality of the IDS through system logs, showcasing its efficiency in managing dynamic network states. The system maintains low resource consumption

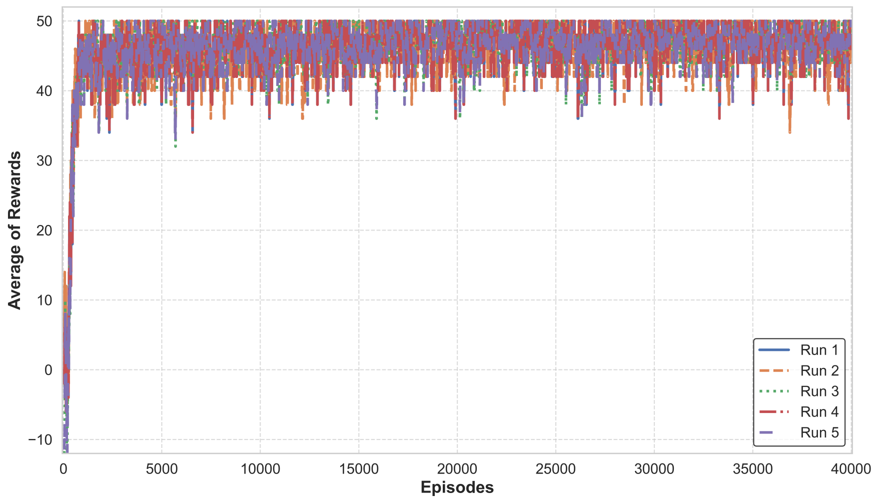

Figure 6.4 Average reward progression for $\epsilon = 0.3$. High exploration fosters learning across diverse and emerging attack scenarios but introduces greater reward variability.

during regular operation without activating mitigation patterns, reflecting a stable and optimized baseline performance. For example, in this state, the CPU usage is 11.8%, energy consumption is 0.118 Joules, and no mitigation patterns are triggered, ensuring efficient utilization of resources.

When the IDS detects a DDoS attack, it swiftly selects and activates the OOC pattern to block suspicious IPs and mitigate the threat. This targeted response demonstrates the system's capability to adapt dynamically to high-risk scenarios. In this active state, resource demands increase, with CPU usage rising to 53.7% and energy consumption reaching 0.537 Joules. Despite these heightened demands, the IDS achieves a rapid response time of 1.0024 seconds, ensuring timely and effective threat mitigation.

6.6.4 Performance Metrics: Training vs. Real-Time Testing

Table 8.2 summarizes the IDS's strong performance during training and real-time testing. With training accuracy at 9% and real-time accuracy at 95%, the system demonstrates high reliability in detecting attacks across dynamic environments. Metrics such as the F1 score (0.96 training, 0.94 real-time) and recall (0.97 training, 0.95 real-time) highlight its ability to identify and mitigate threats with minimal false positives.

The slight decrease in performance during real-time testing compared to training can be attributed to the unpredictable nature of real-world scenarios. Unlike the controlled training phase, real-time environments present more significant variability, including previously unseen attack patterns, noisy network traffic, and dynamic system loads. These factors intro-

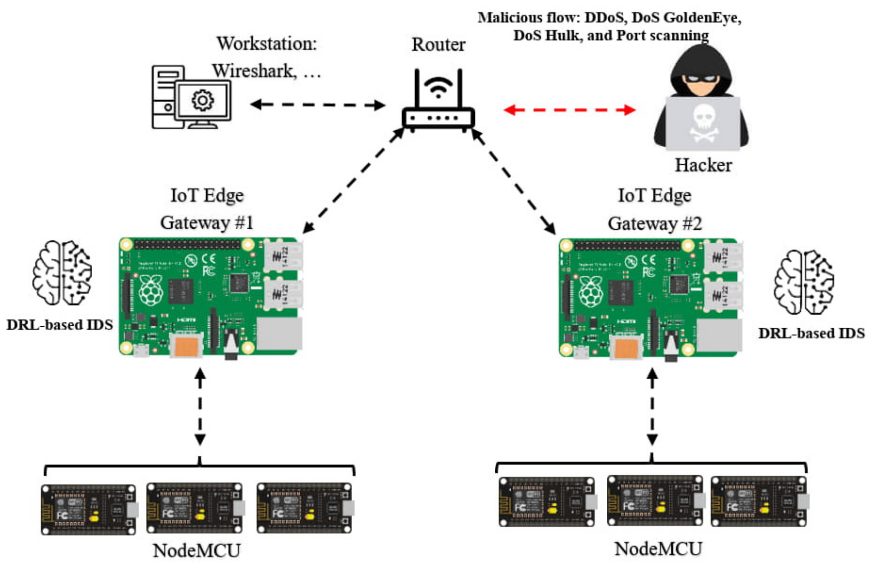
Malicious flow: DDoS, DoS GoldenEye, DoS Hulk, and Port scanning
Workstation: Wireshark, ...
Router
Hacker
IoT Edge Gateway #1
IoT Edge Gateway #2
DRL-based IDS
DRL-based IDS
NodeMCU
NodeMCU

Figure 6.5 Real-world testbed configuration illustrating the deployment of edge gateways.

Table 6.1 Performance Metrics Comparison

Metrics	**Training**	**Real-time Test**
Accuracy	0.97	0.95
F1 Score	0.96	0.94
Recall	0.97	0.95
Precision	0.96	0.95

duce complexities not fully represented in the training dataset, slightly affecting the model's detection accuracy.

6.6.5 Model Performance Evaluation

To address RQ2, we measured resource utilization at the edge under real-time attack scenarios.

6.6.6 Performance Metrics Under DDoS Attacks

The evaluation of DDoS mitigation patterns demonstrates trade-offs in CPU usage, memory usage, energy consumption, IDS response time, and carbon emissions (Fig. 6.7). The BL pattern stands out with the lowest CPU usage (0.69%), energy consumption (50.50 Joules), and fastest response time (0.05 seconds), making it highly efficient for a resource-constrained edge gateway. Moreover, while consuming slightly more CPU (0.82%) and energy (51.64 Joules), the SSN pattern maintains a quick response time of 0.05 seconds. The OOC pattern offers the highest CPU (0.86%) energy consumption (54.36 Joules) and carbon emissions (20.67 kg). All three patterns exhibit consistent memory usage, around 20%. This analysis underscores the BL pattern's resource efficiency, while SSN and OOC cater to scenarios demanding advanced security mitigation.

In addition, the analysis of the impact of DDoS scenarios on various metrics reveals sta-

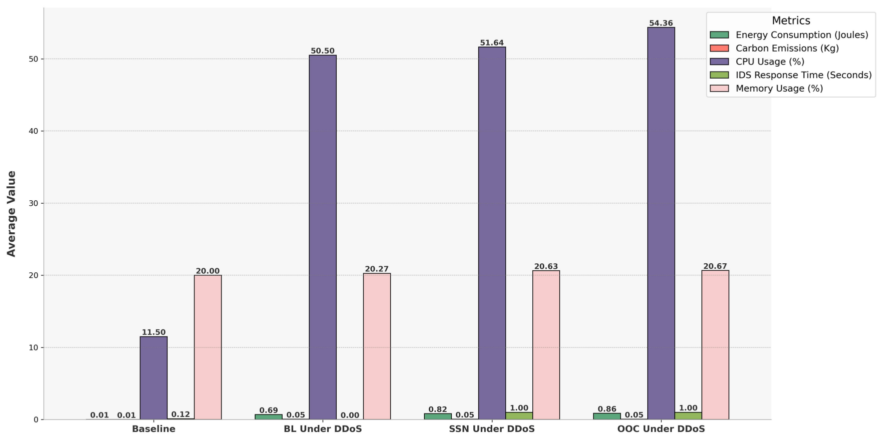

Figure 6.7 Evaluation of mitigation patterns for DDoS attacks, highlighting metrics such as CPU usage, memory usage, energy consumption, IDS response time, and carbon emissions for Blacklist, SSN, and OOC patterns.

tistically significant differences across all comparisons, as p-values are consistently below the threshold of 0.05, leading to the rejection of the null hypothesis. However, the effect sizes vary across metrics, providing critical insights into their practical implications for the DRL-based IDS.

For **Carbon Emissions**, the null hypothesis is rejected across all scenarios (BL_DDoS, SSN_DDoS, OOC_DDoS), confirming statistically significant impacts. The small effect size ($r = 0.01$) highlights minimal environmental overhead, aligning with the IDS's operational efficiency and green computing principles. For **CPU Usage**, statistically significant results lead to rejecting the null hypothesis, confirming substantial changes during the execution of mitigation patterns. Despite this, the small effect size ($r = 0.02$) reflects a marginal increase, demonstrating the system's capacity to handle attacks in real-time without compromising efficiency. Regarding **Energy Consumption**, the null hypothesis is rejected due to statistically significant results, indicating a moderate increase in energy usage. The effect size ($r = 0.03$) remains within acceptable bounds, justifying the trade-off required to neutralize DDoS attacks effectively. For **Memory Usage**, statistically significant differences are confirmed, resulting in rejecting the null hypothesis. The small effect size ($r = 0.02$) denotes a manageable increase, ensuring the system's stability and performance during attack mitigation. Finally, for **IDS Response Time**, the null hypothesis is rejected for the SSN_DDoS and OOC_DDoS scenarios, confirming statistically significant differences. The small effect size ($r = 0.01$) emphasizes minimal practical impact, ensuring the IDS maintains rapid response capabilities to meet critical performance requirements for real-time attack mitigation.

6.6.7 Performance Metrics Under DoS GoldenEye Attacks

Evaluating mitigation patterns BL and PZH under the DoS GoldenEye attack (Fig. 6.8) highlights varied trade-offs in performance metrics. The BL pattern demonstrates the lowest CPU usage (0.62%) and energy consumption (56.33 Joules), coupled with the fastest response time of 0.05 seconds, making it a highly efficient option for a resource-limited edge gateway. In contrast, the PZH pattern requires slightly higher CPU usage (0.56%) and energy consumption (56.61 Joules). Furthermore, both patterns exhibit stable memory usage of around 20%, with the PZH pattern showing slightly higher carbon emissions (20.60 kg) than the BL (20.41 kg).

The analysis of the DoS GoldenEye attack's impact on various system metrics reveals statis-

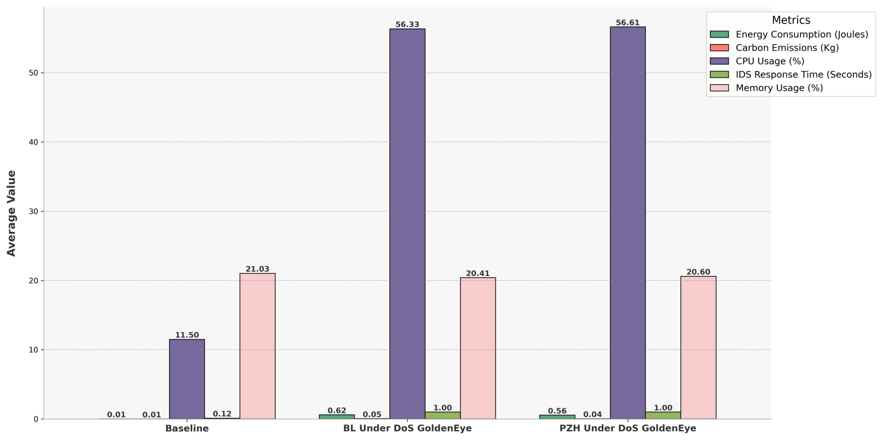

Figure 6.8 Performance metrics for DoS GoldenEye attack mitigation using Blacklist and PZH patterns, comparing CPU usage, memory usage, energy consumption, IDS response time, and carbon emissions.

tically significant differences across all scenarios, as indicated by p-values consistently below the 0.05 threshold, leading to the rejection of the null hypothesis. However, the corresponding effect sizes, which denote the magnitude of these differences, vary across metrics. Rejecting the null hypothesis for **Carbon Emissions** confirms statistically significant variations between scenarios. Nevertheless, the effect size ($r = 0.02$) indicates a minimal environmental impact, underscoring the system's alignment with green computing principles. Similarly, the null hypothesis is rejected for **CPU Usage**, confirming notable differences in resource utilization. Despite this, the negligible effect size ($r = 0.01$) highlights the efficiency of the DRL-based IDS in managing CPU resources without compromising real-time responsiveness. Regarding **Energy Consumption**, rejecting the null hypothesis points to a moderate increase in energy usage during attack mitigation. However, the observed effect size ($r = 0.03$) remains within acceptable bounds, illustrating the system's capacity to neutralize attacks while maintaining a balance in resource consumption effectively. For **Memory Usage**, statistically significant differences are confirmed across both DoS GoldenEye_BL and DoS GoldenEye_PZH scenarios, as indicated by the rejection of the null hypothesis. The small effect size ($r = 0.01$) reflects the IDS's efficiency in preserving memory stability under attack conditions. Finally, for **IDS Response Time**, significant differences are observed across all scenarios, leading to the rejection of the null hypothesis. Despite these differences, the small effect size ($r = 0.01$) underscores the system's capability to maintain rapid detection and response, ensuring minimal delay in threat mitigation.

6.6.8 Performance Metrics Under DoS Hulk Attacks

Evaluating mitigation patterns PZH and WL under the DoS Hulk attack (Fig. 6.9) demonstrates varied performance metrics. The PZH pattern offers minimal CPU usage (0.40%) and a fast response time of 0.04 seconds, consuming 39.85 Joules of energy and emitting 20.59 kg of carbon. On the other hand, the WL pattern achieves slightly lower energy consumption (39.10 Joules) and carbon emissions (20.48 kg) while requiring marginally higher CPU usage (0.39%). Both patterns exhibit consistent memory usage at approximately 20%, ensuring stability in the edge gateway.

The analysis of the DoS Hulk scenarios across various metrics reveals statistically significant differences compared to the baseline, as p-values are consistently below the threshold of 0.05, leading to the rejection of the null hypothesis. However, the effect sizes vary across metrics, offering critical insights into the practical implications of the DRL-based IDS.

For **Memory Usage**, the null hypothesis is rejected for both PZH_Hulk and WL_DoS Hulk scenarios, confirming statistically significant differences. Despite this, the small effect sizes ($r = 0.01$) suggest a minimal practical impact on memory utilization, demonstrating the IDS's ability to handle attack scenarios without compromising system stability. In the case of **Carbon Emissions**, rejecting the null hypothesis confirms significant changes in

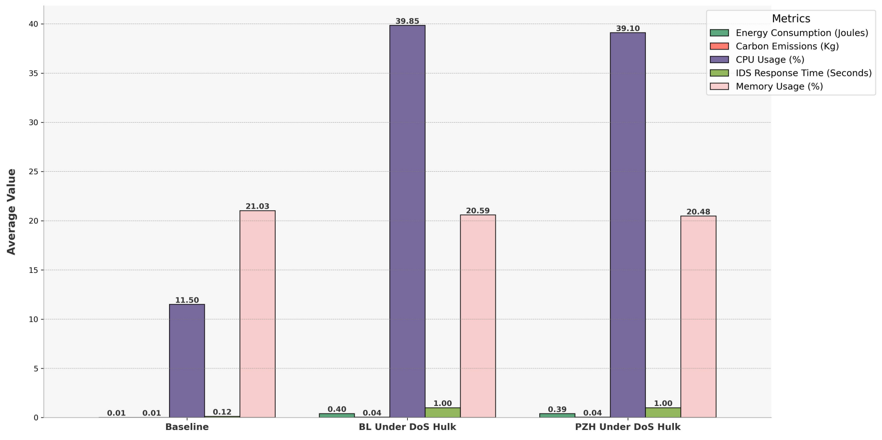

Figure 6.9 Analysis of DoS Hulk attack mitigation patterns (PZH and Whitelist), showing the trade-offs in CPU usage, memory usage, energy consumption, IDS response time, and carbon emissions.

emissions across both scenarios. However, the negligible effect sizes ($r = 0.02$) imply limited real-world significance, aligning with the system's adherence to green computing principles. Both scenarios show statistically significant differences for **CPU Usage**, resulting in rejecting the null hypothesis. Yet, the small effect sizes ($r = 0.01$) emphasize that the increase in CPU utilization remains negligible, reflecting the system's efficiency in managing resource demands during real-time mitigation. Regarding **Energy Consumption**, the null hypothesis is rejected based on statistically significant results, highlighting a measurable increase in energy usage. However, the small effect sizes ($r = 0.03$) indicate minimal changes, ensuring that the IDS balances energy demands with effective attack mitigation. Finally, for **IDS Response Time**, statistically significant differences across both scenarios lead to rejecting the null hypothesis. The small effect sizes ($r = 0.01$) emphasize minimal practical impact on response times, ensuring that the IDS maintains rapid threat detection and mitigation capabilities.

6.6.9 Performance Metrics Under Port Scanning Attacks

The analysis of mitigation patterns OOC, SSN, and WL under Port Scanning attacks (Fig. 6.10) reveals diverse performance characteristics.

While consuming the most carbon emissions (6.05 kg) and energy (0.06 Joules), the OOC pattern achieves a remarkably low CPU usage of 0.01%. The SSN pattern balances energy consumption (1.11 Joules) with consistent CPU usage (0.01%), providing robust IoT security with minimal overhead. Moreover, the WL pattern offers enhanced protection with slightly higher energy consumption (1.92 Joules) and efficient memory utilization (20.33%). Additionally, all patterns exhibit a fast response time of 0.01 seconds, highlighting the efficiency of detecting Port Scanning threats. The analysis of the Port Scanning scenarios across vari-

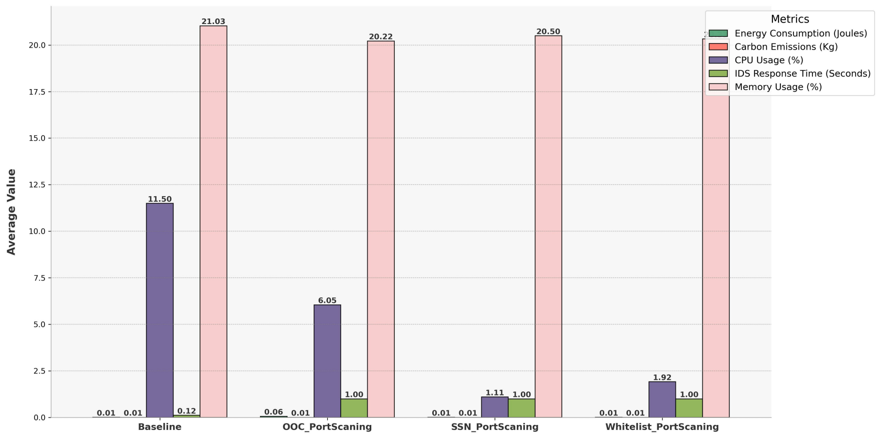

Figure 6.10 Assessment of Port Scanning attack mitigation patterns (OOC, SSN, and Whitelist), evaluating CPU usage, memory usage, energy consumption, IDS response time, and carbon emissions.

ous metrics reveals statistically significant differences compared to the baseline, as p-values are consistently below the threshold of 0.05, leading to the rejection of the null hypothesis. However, the effect sizes vary across metrics, providing details of their practical implications for the DRL-based IDS.

For **Memory Usage**, the null hypothesis is rejected for both OOC_PortScanning and WL_PortScanning scenarios, confirming statistically significant differences. Despite this, the small effect sizes ($r = 0.01$) indicate minimal practical impact on memory utilization, demonstrating the system's ability to maintain stable performance during attack mitigation. In the case of **Carbon Emissions**, statistically significant differences are observed across all scenarios (OOC_PortScanning, SSN_PortScanning, and WL_PortScanning). However, the small effect sizes ($r = 0.02$) suggest limited real-world significance, highlighting the IDS's alignment with green computing principles. For **CPU Usage**, statistically significant differences are confirmed for all scenarios, leading to rejecting the null hypothesis. Despite this, the small effect sizes ($r = 0.01$) emphasize negligible practical differences, reflecting the IDS's efficiency in managing CPU demands. Regarding **Energy Consumption**, the null hypothesis is not rejected, as p-values exceed 0.05 across all scenarios, suggesting no measurable impact on energy usage. This indicates that the IDS effectively mitigates Port Scanning attacks without significant changes in energy consumption. Finally, for **IDS Response Time**, statistically significant differences across all scenarios result in rejecting the null hypothesis. The small effect sizes ($r = 0.01$) emphasize minimal practical impact, ensuring that the IDS maintains rapid detection and mitigation capabilities.

6.7 Discussion

The proposed DRL-based IDS framework represents a significant advancement in securing IoT edge networks by dynamically adapting to evolving threats and optimizing resource utilization. By leveraging adaptive security patterns, the system effectively mitigates diverse attack types, achieving a detection accuracy of 97% during training and 95% in real-world scenarios. The integration of security patterns ensures a modular and scalable defense mechanism. Moreover, the framework aligns with green computing principles, minimizing CPU usage, energy consumption, and carbon emissions while maintaining rapid IDS response times. These capabilities make it particularly well-suited for resource-constrained environments, addressing the dual challenges of robust security and sustainability.
In addition, the performance under real-world conditions highlights its adaptability and practicality for IoT networks. Its dynamic threat detection and mitigation capabilities significantly reduce response times, ensuring minimal disruption to critical operations. Furthermore, balancing mitigation effectiveness with resource efficiency, the framework promotes deploying intelligent security solutions in modern IoT ecosystems.

6.8 Chapter Summary

This study presents a DRL-based IDS framework that dynamically detects and mitigates network attacks at the edge gateway by leveraging adaptive security patterns. Focusing on attacks, e.g., DDoS, DoS Hulk, DoS GoldenEye, and Port Scanning, the system enhances detection efficiency, response time, and resource utilization while aligning with sustainability goals through reduced energy consumption and carbon emissions. Using patterns, i.e., BL, WL, SSN, PZH, and OOC, it effectively balances mitigation strategies with resource constraints, as demonstrated in a real-world IoT edge testbed.

CHAPTER 7 ARTICLE 5: SELF-ADAPTIVE CYBER DEFENSE FOR SUSTAINABLE IOT: A DRL-BASED IDS OPTIMIZING SECURITY AND ENERGY EFFICIENCY

7.1 Article Metadata

- **Title:** Self-Adaptive Cyber Defense for Sustainable IoT: A DRL-Based IDS Optimizing Security and Energy Efficiency
- **Authors:** Saeid Jamshidi, Ashkan Amirina, Amin Nikanjam, Kawser Wazed Nafi, Foutse Khomh, Samira Keivanpour
- **Affiliations:**
  - SWAT Laboratory, Polytechnique Montréal, H3T 1J4, Quebec, Canada
  - Huawei Distributed Scheduling and Data Engine Lab, Canada
  - Poly Circle X.O, Polytechnique Montréal, H3T 1J4, Quebec, Canada
- **Submitted Date:** 8 December 2024
- **Accepted Date:** 17 March 2025
- **DOI:** `https://doi.org/10.1016/j.jnca.2025.104176`
- **Journal:** *Journal of Network and Computer Applications*

7.2 Chapter Overview

The IoT has revolutionized industries by creating a vast, interconnected ecosystem. Still, the rapid deployment of IoT devices has introduced severe security risks, including DDoS, DoS GoldenEye, DoS Hulk attacks, and Port scanning. Traditional ML-based IDS often operate passively, detecting threats without taking action, and are rarely evaluated under real-time attacks. This limits our understanding of their performance within the resource constraints typical of IoT systems—an essential factor for stable, resilient systems. This chapter proposes a Security Edge with Deep Reinforcement Learning (SecuEdge-DRL) specifically designed for the IoT edge, aiming to enhance security while maintaining energy efficiency, contributing to sustainable IoT operations. Our IDS integrates DRL with the MAPE-K (Monitor, Analyze, Plan, Execute, Knowledge) control loop, enabling real-time detection and adaptive response without relying on predefined data models. DRL allows continuous learning, while MAPE-K provides structured self-adaptation, ensuring the system remains effective against evolving threats. We also implemented four targeted security policies tailored to a specific attack type to enhance the IDS's threat mitigation capabilities. Experimental findings indicate that the proposed SecuEdge-DRL achieves an average detection accuracy of 92% across diverse real-world cyber threats (e.g., DoS Hulk, DoS GoldenEyes, DDoS, and Port scanning). Statistical analysis further validates that these security policies enhance IoT systems' defense without compromising performance, establishing our approach as a resilient, resource-efficient security solution for the IoT ecosystem.

7.3 Study Design

The IoT rapidly expands across diverse sectors, including smart homes, healthcare, and industrial automation [258]. By 2025, the number of IoT devices worldwide is projected to surpass 75 billion, driving a market valued at approximately \$10 trillion [259]. This growth has led to an increasingly interconnected digital landscape, offering significant efficiency. However, this connectivity also presents critical security challenges. Cybercriminals exploit IoT vulnerabilities through various attacks, including DDoS, GoldenEye DoS, Hulk DoS, and Port scanning [260] [261] [262] [263]. These attacks disrupt services, compromise data, and lead to financial losses [10]. For instance, GoldenEye DoS attacks have overwhelmed IoT-connected security cameras, causing extended downtime, while Hulk DoS attacks have similarly affected smart home systems by overloading web servers [205] [264]. Additionally, Port scanning enables hackers to identify open ports on IoT devices, increasing the risk of exploitation [265]. Such persistent threats underscore the need for robust, real-time security solutions in IoT systems.

Many IoT devices are limited in processing power and energy capacity, making traditional security solutions challenging to implement [266] [267]. Adequate IoT security requires lightweight and energy-efficient solutions, enabling these devices to function autonomously on limited power [91]. However, current IDS face challenges in balancing detection accuracy and resource efficiency, especially in dynamic IoT systems [268] [10]. These limitations highlight the urgent need for advanced solutions that balance security, resource efficiency, and adaptability.

As IoT devices proliferate, edge systems' energy efficiency and resource consumption become critical for sustainability [46] [45]. Cybersecurity frameworks must balance robust threat detection with minimal resource impact, ensuring IoT systems align with Sustainable Development Goals (SDGs) [269] by reducing energy waste and optimizing computational resources [270]. This chapter addresses this dual challenge by proposing an energy-efficient, adaptive IDS for sustainable IoT ecosystems.

This chapter proposes a novel data-driven Security Edge with Deep Reinforcement Learning (SecuEdge-DRL) for IoT edge gateways. To accomplish this, we first conducted a foundational analysis of IoT cyber threats to understand and extract characteristic patterns. Leveraging these insights, we developed a DRL-based IDS that incorporates the MAPE-K (Monitor, Analyze, Plan, Execute, Knowledge) control loop for self-adaptive threat detection and response [271]. This integration enables the IDS to adjust its detection and response strategies continuously, allowing for real-time learning and adaptation. The system's dynamic approach effectively balances security and resource utilization, even within the constraints of IoT devices. We trained the IDS on attack data from the CICIDS2017 dataset for optimal performance, simulating diverse cyber threats to enhance detection accuracy across the edge.

To validate SecuEdge-DRL's robustness and efficiency, we deployed it on a Raspberry Pi as an edge gateway and subjected it to real-time attack scenarios. We measured the system's energy consumption, CPU load, and CPU usage in both threat and normal states. Statistical analysis, including the Mann–Whitney U test, assessed the impact of each security policy on these metrics. Our findings demonstrate that the IDS maintains high efficiency even under cyber threats, confirming its suitability for a resource-constrained edge gateway.

The significant contributions of this chapter can be summarized as follows:

- Proposing a novel DRL-based IDS (SecuEdge-DRL) for the edge of IoT: It can detect various types of cyber threats, including DDoS, DoS GoldenEye, DoS Hulk, and Port scanning. This DRL-based IDS adapts to the changing situations of the edge of IoT infrastructure to ensure detection effectiveness without needing prior knowledge.

- Develop four distinct security policies customized to respond to specific types of cyber threats, optimizing the system's defensive capabilities on the edge of IoT and integrating the MAPE-K framework to adjust detection and response strategies dynamically.

- Evaluation of the performance of the four distinct security policies, including energy consumption, CPU load, and CPU usage, under both cyber threats and normal conditions. This evaluation includes the statistical analysis of performance metrics using the Mann–Whitney U test to assess the impact of each security policy on energy consumption, CPU load, and CPU usage.

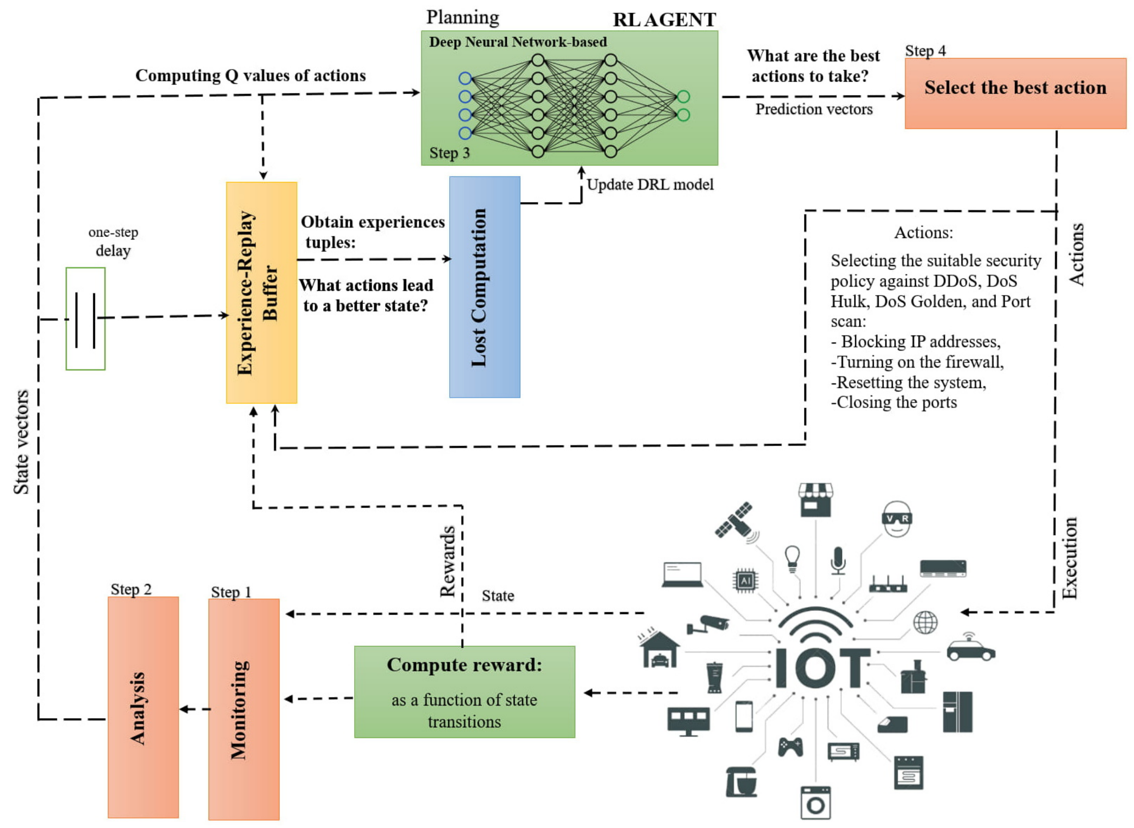

Figure 7.1 Overview of the proposed approach in the MAPE-K framework.

7.4 SecuEdge-DRL Method

The SecuEdge-DRL is a data-driven approach that integrates DRL into the MAPE-K framework to detect cyber threats at the edge and adapt automatically. As Figure 7.1 illustrates, SecuEdge-DRL progresses through several key phases. It begins with the monitoring phase, which is focused on data preprocessing. This step is crucial for managing the large volume of data samples and reducing performance overhead, ensuring that the data is correctly formatted for effective attack detection.

Data collected at each edge gateway is periodically transmitted to the edge server, where our DRL model analyzes it. Next is the analysis phase, in which the DRL model identifies abnormal patterns in the preprocessed data. These patterns are then forwarded to the planning phase, in which the DRL model evaluates the anomalies and determines the optimal response.

Then, the execution phase implements the chosen actions (e.g., security policies), calculating a reward based on feedback from the edge gateway. This transitions the model from the current state to the next. Simultaneously, the model is iteratively updated using the knowledge base, which stores Markov Decision Processes (MDPs) batches containing past states, actions, rewards, and transitions. This iterative update enhances the model's learning capabilities by drawing on historical data.

Each edge server equipped with the DRL model continuously refines its detection capabilities through the reward-driven learning mechanism inherent in the MDP framework. The current state, action, reward, and resulting state are logged after each iteration, and a batch of this history is periodically sampled to train the DRL agent. This ongoing process dynamically improves the model's effectiveness in monitoring and analysis.

7.4.1 MAPE-K cycle

Figure 7.2 illustrates the SecuEdge-DRL workflow within the MAPE-K cycle, adapted from our previous work [2]. The system continuously monitors edge traffic, analyzes detected anomalies using DRL, plans adaptive security policies, executes mitigation actions, and updates its knowledge base to enhance continuous learning and adaptability.

7.4.2 Monitoring

The monitoring phase at the edge involves collecting data D from various strategically placed sensors $S = \{S_1, S_2, \ldots, S_n\}$ in the system, for example, network traffic or firewall connections. The primary objectives of monitoring are identifying suspicious activities and ensuring system safety. The collected data D_t at time t is given by:

$$D_t = \bigcup_{i=1}^{n} f(S_i, t) \tag{7.1}$$

where $f(S_i, t)$ represents the data collected from sensor S_i at time t. Each $f(S_i, t)$ captures the raw data from an individual sensor, including observing network traffic N_t, tracing inbound and outbound connections via firewalls F_t, tracking database access DB_t, and monitoring end device controller activity C_t. Each monitoring event involves extracting relevant data from the collected information:

$$\text{Events}_t = g(D_t) \tag{7.2}$$

where $g(D_t)$ represents a function that processes the raw data D_t to extract actionable insights. For instance, $g(D_t)$ could involve identifying unusual spikes in network traffic volume, abnormal packet rates, or specific patterns (e.g., repeated attempts to access closed ports), which is indicative of potential Port scanning. By aggregating and filtering this information,

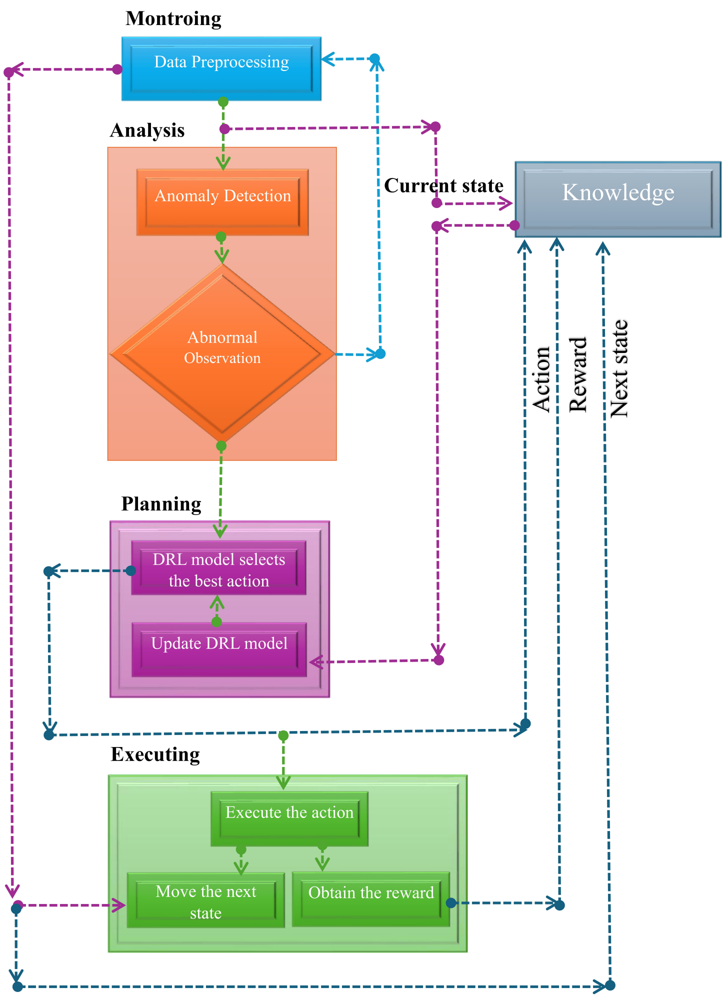

Figure 7.2 SecuEdge-DRL process flow in the MAPE-K framework [2].

$g(D_t)$ converts raw data into meaningful events that signify potential threats.

7.4.3 Analyzing

During the analysis phase, the collected data D_t is processed to identify security threats at the edge. This involves applying a set of detection functions $\mathcal{A} = \{a_1, a_2, \ldots, a_m\}$:

$$\text{Threats}_t = \bigcup_{j=1}^{m} a_j(D_t) \tag{7.3}$$

Where a_j is a function that identifies threat type j. If any threats are detected, the corresponding set of events E_t is generated:

$$E_t = \{e_k \mid e_k \in \text{Threats}_t\} \tag{7.4}$$

Here, Threats_t represents the identified threats at time t, capturing the dynamic nature of threat detection at the edge gateway. Associating threats with t reflects the system's state at specific intervals, which is crucial for timely responses. The detection functions a_j generate alerts based on observed security metrics, triggering change requests when vulnerabilities are identified. These requests are forwarded to the planning phase for appropriate self-adaptation policies.

7.4.4 Planning

In the planning phase, a mitigation strategy P_t is selected from available self-adaptation strategies to address detected threats. Based on the analysis findings, a change plan is generated:

$$P_t = \arg\max_{p \in \mathcal{P}} \mathbb{E}[R(p, E_t)] \tag{7.5}$$

Where $R(p, E_t)$ is the expected reward of applying policy p given the events E_t. The selected policy P_t defines the necessary actions A_t:

$$A_t = \text{Actions}(P_t) \tag{7.6}$$

These actions, such as blocking IP addresses, activating firewalls, resetting the system, or closing ports, are detailed in Section 7.4.7. The DRL model dynamically evaluates and learns from previous observations to make informed decisions against cyber threats. This process operates within the Markov Decision Process (MDP) framework:

$$\text{MDP} = \begin{cases} \text{State (S):} & \text{Security metrics} \\ \text{Action (A):} & \text{Security policy} \\ \text{Reward (R):} & \text{Environment feedback (+/-)} \end{cases} \tag{7.7}$$

7.4.5 Executing

The execution function carries out actions A_t defined in the planning phase through actuators on the target system. Specific actions are executed to address the detected or potential threats, leading the system to the next state:

$$\text{State}_{t+1} = \bigcup_{l=1}^{r} x_l(A_t) \tag{7.8}$$

Where State_{t+1} represents the new state of the system after executing the actions A_t. These actions directly modify the security state of the edge gateway by addressing identified threats through predefined mitigation strategies.

7.4.6 Knowledge

The knowledge base (K) maintains details of the agent and its interactions with the environment, adaptation goals, and other relevant information across all MAPE phases. At time t, the knowledge base can be represented as:

$$K_t = \{D_\tau, E_\tau, P_\tau, A_\tau \mid \tau \leq t\} \tag{7.9}$$

In other words, the knowledge base systematically records the system's state, actions, events, and adaptation decisions, serving as a critical repository for continuous learning and reference across MAPE phases. This structured history enables the model to refine its decision-making process over time, ensuring it remains responsive to evolving cyber threats. A key advantage of this knowledge-driven approach is that it enhances the interpretability of the SecuEdge-DRL framework, addressing the common issue of deep learning models being hard to interpret. While logging actions alone does not provide complete traceability, SecuEdge-DRL's knowledge base goes beyond simple action tracking by storing the reasoning behind decisions. Precisely, it captures which security indicators (e.g., abnormal traffic spikes, unauthorized access attempts, protocol anomalies) influenced a particular action, along with a structured representation of state transitions, feature importance, and policy selection criteria. This allows analysts to understand not just what the model did but why it made that choice. On the other hand, the reward function is explicitly designed to reinforce interpretable decision-making, linking security actions to measurable threat assessments rather than opaque, model-internal computations. For instance, when SecuEdge-DRL blocks an IP address or enables a firewall rule, it decides based on an aggregated threat score derived from historical observations, ensuring its responses are context-aware and justifiable. By integrating structured knowledge representation, feature attribution, and decision-context tracking, SecuEdge-DRL provides a level of explainability that is practical for security analysis, making it possible to audit and refine its threat mitigation strategies.

7.4.7 DRL-Enhanced MAPE-K for Adaptive Threat Response

This section explains how DRL integrates into the MAPE-K framework to enable adaptive, real-time threat mitigation, enhancing edge security through continuous learning.

7.4.8 State

The **state** s encapsulates the current state of the edge, which is crucial for the decision-making process in SecuEdge-DRL. It is a vector of features extracted from network traffic and other security-related metrics collected during the monitoring phase, such as the rate of incoming and outgoing packets, the number of failed login attempts, and abnormal port access patterns. These features provide a snapshot of the security posture of the edge at any given time and are vital for detecting and mitigating threats. Formally, the state for a given timestep t is defined as:

$$s_t = X_{\text{train}}[i] \tag{7.10}$$

Where X_{train} represents the dataset of security-relevant features derived from the monitoring phase, and i is the index of the current training sample. These features include packet flow rates, abnormal traffic patterns, port access behaviors, and unusual packet size or volume spikes. By capturing these critical aspects of network traffic and other security-related activities, the state s_t effectively represents the security posture of the edge gateway at time t. Since these features are directly obtained from the monitoring phase (as detailed in Section 7.4.2), they provide real-time insights into the network's status. By continuously monitoring these features, the DRL-based IDS can dynamically learn and adapt to evolving threats, ensuring timely and accurate detection of cyberattacks while maintaining an up-to-date understanding of the edge's security status.

7.4.9 Action

Section 7.4.4 mentions that the proposed method defines **actions** as the predefined security responses the system can execute to mitigate detected threats. The action space includes:

- **Blocking IP Addresses**: Preventing communication from suspicious sources by adding the IP addresses to a blocklist. Mathematically, this is defined as a mapping:

$$a_{\text{block}}(s_t) = \{ip \in \mathbb{IP} \mid f_{\text{risk}}(ip, s_t) > \theta_{\text{risk}}\} \tag{7.11}$$

  Where $\mathbb{IP}$ is the set of all IP addresses. The function $f_{\text{risk}}(ip, s_t)$ represents a risk assessment score derived from factors, e.g., the frequency of malicious requests or known attack sources. The threshold θ_{risk} is determined empirically based on prior training data and security policies.

- **Activating Firewalls**: Enforcing security rules to control incoming and outgoing network traffic, effectively filtering out malicious packets. This function is defined as:

$$a_{\text{firewall}}(s_t) = \sum_{r \in R} \delta_{s_t}(r) \cdot w_r \tag{7.12}$$

  Where R is the set of firewall rules. The activation state $\delta_{s_t}(r)$ is determined based on predefined rules such as traffic volume thresholds or specific protocol violations observed in s_t. Each rule r is weighted by w_r, assigned during system configuration to prioritize critical security policies.

- **Resetting Systems**: Resetting devices to disrupt ongoing attacks and clear any malicious activity. Mathematically, this action is modeled as:

$$a_{\text{reset}}(s_t) = \text{Reset}(D_{\text{affected}}(s_t)) \tag{7.13}$$

  Where $D_{\text{affected}}(s_t)$ identifies compromised devices in state s_t based on anomaly detection or predefined indicators such as repeated login failures or unauthorized access attempts.

- **Closing Ports**: Disabling network ports being attacked or scanned to prevent unauthorized access and further exploitation. This is expressed by:

$$a_{\text{close}}(s_t) = \{p \in P \mid g_{\text{scan}}(p, s_t) = \text{true}\} \tag{7.14}$$

  Where P is the set of all network ports. The function $g_{\text{scan}}(p, s_t)$ identifies whether a port p is currently under attack based on metrics such as the frequency of connection attempts or patterns indicative of scanning activity.

Actions are determined by either a policy that maximizes the expected utility of the response (exploitation) or by exploring new actions (exploration), depending on the epsilon-greedy strategy:

$$a_t = \begin{cases} \arg\max_a Q(s_t, a) & \text{if random} > \epsilon \\ \text{random action from the action space} & \text{otherwise} \end{cases} \tag{7.15}$$

The agent selects the most appropriate action based on the current state to maximize the reward, which is designed to keep the edge gateway secure.

7.4.10 Adaptive Resilience Against Hardware Failures

SecuEdge-DRL integrates real-time monitoring for cyber threats and hardware failures to ensure long-term system reliability. The system continuously evaluates CPU usage, memory availability, temperature metrics, and hardware stress factors, enabling proactive failure prevention alongside attack mitigation. Let H_t represent the overall system health at time t, now defined as:

$$H_t = \{CPU_t, Mem_t, Temp_t, P_t, F_t\} \tag{7.16}$$

Where:

- CPU_t = CPU usage
- Mem_t = Available memory
- $Temp_t$ = Device temperature
- P_t = Attack probability
- F_t = Hardware failure risk score

An attack and hardware failure risk is detected when the deviation from baseline system metrics $\bar{H}$ exceeds an adaptive threshold H_{thr}:

$$\Delta H_t = |H_t - \bar{H}| \tag{7.17}$$

$$\text{If } \Delta H_t > H_{thr}, \text{ then initiate mitigation.} \tag{7.18}$$

Unlike traditional reactionary failure handling, SecuEdge-DRL employs predictive failure mitigation strategies through its MAPE-K adaptation cycle:

$$M_t = \begin{cases} \text{Redistribute workload,} & CPU_t \geq CPU_{thr} \\ \text{Activate cooling measures,} & Temp_t \geq Temp_{crit} \\ \text{Optimize memory usage,} & Mem_t \leq Mem_{low} \\ \text{Enable failover mode,} & F_t \gg F_{thr} \\ \text{Isolate failing components,} & F_t \ggg F_{thr} \end{cases} \tag{7.19}$$

- Dynamic Load Redistribution: Prevents CPU overload by balancing tasks before reaching critical utilization at CPU_{thr}.
- Automated Cooling Activation: Engages active cooling and performance throttling when the temperature approaches 85°C ($Temp_{crit}$).
- Memory Pressure Handling: Detects memory constraints below 10% (Mem_{low}) and initiates process cleanup and reallocation.
- Hardware Failure Isolation: Predicts critical hardware failures and proactively isolates failing nodes before reaching F_{thr}.

These adaptive policies ensure that SecuEdge-DRL is resilient against cyber threats and hardware degradation, maintaining operational stability even under a resource-constrained edge gateway.

7.4.11 Security Policy Selection and Real-Time Execution

The SecuEdge-DRL framework dynamically enforces four security policies to mitigate cyber threats at the edge. These policies are selected based on real-time network states, using DQN to optimize decision-making. The DRL model continuously updates its policy choices based on observed threat indicators, ensuring an adaptive and efficient response.

7.4.12 Policy Selection Criteria

Each security policy P_t is selected based on an evaluation of the current network state s_t, following:

$$P_t = \arg\max_{p \in \mathcal{P}} Q(s_t, p) \tag{7.20}$$

Where:

- P_t is the selected policy at time t,
- $\mathcal{P}$ is the set of available security policies,
- $Q(s_t, p)$ represents the expected reward of executing policy p under state s_t.

The decision-making process considers real-time metrics, including traffic anomalies, failed authentication attempts, and unusual port activity.

7.4.13 Real-Time Policy Execution and Practical Examples

The four policies and their respective execution triggers are detailed below:

1. **Blocking IP Addresses (P_1)** – Mitigates DDoS and DoS Hulk attacks.
   Trigger: Activated when an IP sends an abnormally high volume of requests exceeding the threshold θ_{block}.
   $$P_1 = \{ip \mid f_{\text{anomaly}}(ip, s_t) > \theta_{\text{block}}\} \tag{7.21}$$
   Example: A DDoS attack is detected when multiple requests from the same IP exceed the threshold. The system blocks the IP:
   ```
   iptables -A INPUT -s [IP_ADDRESS] -j DROP
   ```

2. **Activating Firewall Rules (P_2)** – Prevents DoS GoldenEye attacks.
   Trigger: Initiated when the firewall detects repeated suspicious connections surpassing θ_{firewall}.
   $$P_2 = \{r \in R \mid f_{\text{traffic}}(s_t) > \theta_{\text{firewall}}\} \tag{7.22}$$
   Example: A DoS GoldenEye attack floods the edge gateway with HTTP requests. The system dynamically enables stricter firewall rules:
   ```
   ufw enable
   ```

3. **Resetting the System (P_3)** – Counters sustained attacks.
   Trigger: Engaged when multiple intrusion attempts cause system degradation beyond θ_{reset}.
   $$P_3 = \{S \mid f_{\text{compromise}}(S, s_t) > \theta_{\text{reset}}\} \tag{7.23}$$
   Example: A persistent malware infiltration bypassing the firewall. The system resets itself to restore normal operations:
   ```
   sudo reboot
   ```

4. **Closing Ports (P_4)** – Prevents Port Scanning and Unauthorized Access.
   Trigger: Executed when failed access attempts to an unused or unauthorized port exceed θ_{port}.
   $$P_4 = \{p \in P \mid f_{\text{scan}}(p, s_t) > \theta_{\text{port}}\} \tag{7.24}$$

Example: A hacker attempts a port scan to discover open network ports. The IDS closes the targeted ports:

```
iptables -A INPUT -p tcp --dport [PORT_NUMBER] -j DROP
```

7.4.14 Reward

The **reward** r measures the success of the actions taken by the agent. It motivates the agent to learn effective strategies for mitigating threats. The reward function in our IDS is defined as:

$$r_t = \begin{cases} \text{Positive Reward} & \text{if action mitigates the threat} \\ \text{Negative Reward} & \text{if action fails or has negative impact} \end{cases} \tag{7.25}$$

- **Positive Reward**: Assigned when the action successfully mitigates a threat, aligning with the expected outcome. The positive reward value is set based on the severity of the mitigated threat. For example, successfully blocking a DDoS attack might yield a higher reward (e.g., +10) than blocking a low-risk port scan (e.g., +5). These values are determined empirically during training to encourage prioritization of critical threats while ensuring consistent learning.
- **Negative Reward**: Assigned when the action fails to mitigate the threat or causes a negative impact, such as an unnecessary system reset or failure to block malicious traffic. The negative reward value is also set based on the severity of the failure. For instance, failing to respond to a DDoS attack might incur a significant penalty (e.g., -10). At the same time, an incorrect response to a less critical threat may result in a more minor penalty (e.g., -5). These values are calibrated during training to penalize ineffective actions without destabilizing the learning process.

The reward function ensures that the DRL agent learns to take actions that effectively protect the edge while avoiding actions that do not contribute to security. This feedback enables the agent to improve decision-making over time, optimizing the balance between security and resource utilization.

7.4.15 Implementing SecuEdge-DRL at the IoT Edge

This section presents the implementation of the SecuEdge-DRL at the edge gateway. The training process of the SecuEdge-DRL model, including action selection, policy execution, and reward-based updates, is detailed in Algorithm 8.

The process begins with the initialization of key parameters, including the exploration probability (ϵ), its minimum value (ϵ_{min}), decay rate (ϵ_{decay}), discount factor (γ), and replay memory (M). The DRL model is configured with these parameters to facilitate dynamic learning. Note that in Algorithm 8, replay memory (M) plays a critical role in storing state transitions, which are later used for training and updating the Q-function.

At the start of each episode, the system observes the initial state (s_i) by extracting key security-relevant features from the edge gateway, as described in Section 7.4.2. These features provide a snapshot of the current network and device status.

During the analysis phase, the system determines the appropriate action. If the network is under attack, the SecuEdge-DRL is invoked to select the optimal security policy. This decision-making process follows an epsilon-greedy strategy:

$$a_i = \begin{cases} \text{Random action}, & \text{if } \xi < \epsilon, \quad \xi \sim \text{Uniform}(0,1) \\ \arg\max_a Q(s_i, a), & \text{otherwise.} \end{cases}$$

The exploration-exploitation trade-off ensures the model balances trying new actions (exploration) with leveraging learned actions that maximize rewards (exploitation).

Algorithm 8: DRL-based IDS Incorporating MAPE-K with Security Policy Execution

```
   Input  : $\epsilon$: Initial exploration rate, $\epsilon_{\text{min}}$: Minimum exploration rate, $\epsilon_{\text{decay}}$: Exploration decay rate,
            $\gamma$: Discount factor, $M$: Replay memory, $K$: Number of episodes
   Output: Trained DRL-based IDS model
 1 Initialize: Model parameters and replay memory $M$
 2 for each episode $i \in \{1, 2, \ldots, K\}$ do
 3     Monitor: Extract security-relevant features and observe initial state $s_i$
 4     Analyze: Choose action $a_i$ using $\epsilon$-greedy policy:
 5     if random value $< \epsilon$ then
 6         Select a random action (exploration)
 7     else
 8         Select action maximizing $Q(s_i, a)$ (exploitation)
 9     end
10     Plan: Map action $a_i$ to a security policy:
11         $P_1$: Block IP addresses
12         $P_2$: Enable firewall rules
13         $P_3$: Reset the system
14         $P_4$: Close vulnerable ports
15     Execute: Apply selected policy, observe reward $r_i$, and transition to next state $s_{i+1}$
16     Update Knowledge: Store transition $(s_i, a_i, r_i, s_{i+1})$ in replay memory $M$
17     Optimize:
18         Sample a batch $\mathcal{B}$ from $M$
19         Update Q-values using Bellman equation:
               $Q(s, a) \leftarrow Q(s, a) + \alpha \left[ r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$
20     Adjust Exploration: Reduce $\epsilon$:
               $\epsilon \leftarrow \max(\epsilon_{\text{min}}, \epsilon \cdot \epsilon_{\text{decay}})$
21 end
22 return Trained DRL-based IDS model
```

The chosen action is then executed, transitioning the system from s_t to s_{t+1}. The tuple $(s_t, a_t, r_{t+1}, s_{t+1})$ is stored in the replay memory (M) for future training. The reward r_{t+1} is computed based on the effectiveness of the action in mitigating the detected threat.

Subsequently, a batch of transitions is sampled from the replay memory to train the DRL model. The Q-function is updated using the following equation:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right],$$

Where α is the learning rate, and γ is the discount factor. Finally, the exploration probability ϵ is decayed using the predefined decay rate (ϵ_{decay}):

$$\epsilon \leftarrow \max(\epsilon_{\min}, \epsilon \cdot \epsilon_{\text{decay}}).$$

This ensures a gradual shift from exploration to exploitation as training progresses. This iterative process continues for each episode, enabling the IDS to adapt dynamically and respond effectively to evolving cyber threats.

The execution of the security policies on the edge gateway is managed through specific system commands, as described in Algorithm 9. The trained DRL model predicts the optimal security action based on the observed state. These actions are executed through system commands such as `iptables` for blocking IPs or closing ports and `ufw` for firewall control. Each action is logged to maintain a record of security measures for audit purposes.

Algorithm 9: Execution of Security Policies at the Edge Gateway

```
1:  Load the trained DRL-based IDS model onto the edge gateway
2:  Initialize the system for continuous monitoring of network traffic
3:  while system is operational do
4:      Monitor and analyze incoming network traffic to detect current state state
5:      Utilize the IDS model to predict the optimal security action action for the detected
        state
6:      if action = Block_IP then
7:          Execute command: iptables -A INPUT -s [IP_ADDRESS] -j DROP to block the
            suspicious IP address
8:      else if action = Enable_Firehaul then
9:          Activate the firewall using the command: ufw enable
10:     else if action = Reset_System then
11:         Issue a system reboot with the command: sudo reboot
12:     else if action = Close_Ports then
13:         Close specific ports using command: iptables -A INPUT -p tcp -dport
            [PORT_NUMBER] -j DROP
14:     end if
15:     Log the action and the outcome to maintain a record of security audits
16: end while
```

7.5 Experimental setup

This section outlines the experimental methodology for evaluating our DRL-based IDS for IoT edge security. It includes data evaluation, feature selection, and testbed setup to assess the model's performance effectively.

7.5.1 Data Set Evaluation

We conducted a preliminary performance evaluation of the SecuEdge-DRL using the CI-CIDS2017 dataset [134]. This dataset is widely recognized for its comprehensive traffic data, encompassing various network attacks, including DDoS, DoS GoldenEye, DoS Hulk, and Port scanning. It includes 83 traffic features extracted over different time windows, providing a robust basis for evaluating IDS models (Table 7.1 shows only 23 of such features).

7.5.2 Feature selection

We prioritize selecting the most impactful features from observable traffic data to ensure efficient cyber threat detection on edge gateways without compromising device performance. Minimizing the number of selected features is essential for reducing computational load while maintaining feature independence to enable accurate classification. We focused on detecting DDoS, DoS GoldenEye, DoS Hulk attacks, and Port scanning. Initially, 83 features were extracted using CICFlowMeter [257]. Through a detailed analysis of the Cumulative Distribution Function (CDF) of feature variance, we identified 23 features with significant contributions to detecting these attacks [256]. These features optimize detection for targeted attacks and are potentially effective for identifying other cyber threats by capturing critical network behaviors. The heavy-tailed data, representing extreme values, play a crucial role in detecting anomalies, often indicative of malicious activity. We applied the Correlation-based Feature Selection (CFS) method to refine the feature set further. By calculating the correlation coefficients between features, we discarded redundant or strongly correlated ones to enhance feature independence, ultimately improving classifier performance. This process resulted in selecting 23 essential features, as shown in Table 7.1, which ensures robust, generalizable IoT security across multiple threat scenarios.

7.5.3 Baselines

Two research works mentioned in the related work section are closely aligned with our proposed model. So, for the performance evaluation of the SecuEdge-DRL, we selected these two approaches as our baselines. Ramana et al. [98] introduced an IDS detection approach utilizing a Deep Q Network integrated with RL, designed explicitly for IoT edge devices and their associated data storage layers. In contrast, Y. Feng et al. [110] applied RL at IoT edge gateways to dynamically adjust the parameters of an unsupervised classifier for detecting network attacks. Their model also incorporates a collaborative aggregation module that enables gateway devices to share updated parameters across the network. However, their approach is limited to detecting the DDoS attack.

Table 7.1 Selected Features

Feature	**Definition**
bwd_packet_length_min	Minimum length of backward packets.
subflow_fwd_bytes	Total number of bytes in the forward subflow.
total_length_of_fwd_packets	Total length of all forward packets.
fwd_packet_length_mean	Mean length of forward packets.
bwd_packet_length_std	Standard deviation of the length of backward packets.
flow_iat_min	Minimum inter-arrival time between packets in a flow.
fwd_iat_min	Minimum inter-arrival time of forward packets.
flow_iat_mean	Mean inter-arrival time between packets in a flow.
flow_duration	Duration of the flow.
flow_iat_std	Standard deviation of the inter-arrival time between packets in a flow.
active_min	Minimum active time duration.
active_mean	Mean active time duration.
bwd_iat_mean	Mean inter-arrival time of backward packets.
fwd_iat_mean	Mean inter-arrival time of forward packets.
init_win_bytes_forward	Initial window bytes in the forward direction.
ack_flag_count	Count of packets with the acknowledgment flag.
fwd_psh_flags	Number of forward packets with push flags.
syn_flag_count	Count of packets with the SYN flag.
fwd_packets/s	Number of forwarding packets per second.
init_win_bytes_backward	Initial window bytes in the backward direction.
bwd_packets/s	Number of backward packets per second.
psh_flag_count	Count of packets with push flags.
packet_length_mean	Mean length of packets.

7.6 Experiments and Evaluation

This section discusses and evaluates the SecuEdge-DRL's performance in detecting various network attacks, including DDoS, DoS GoldenEye, DoS Hulk, and Port scanning. We start with an exploratory evaluation, reporting our results and briefly analyzing the findings. Finally, we compare the SecuEdge-DRL to baseline methods, demonstrating how it outperforms existing approaches.

7.6.1 Exploratory Evaluation

To evaluate the performance of the SecuEdge-DRL, we first proceed with an exploratory study to assess the effect of epsilon values. We evaluated the impact of the exploration-

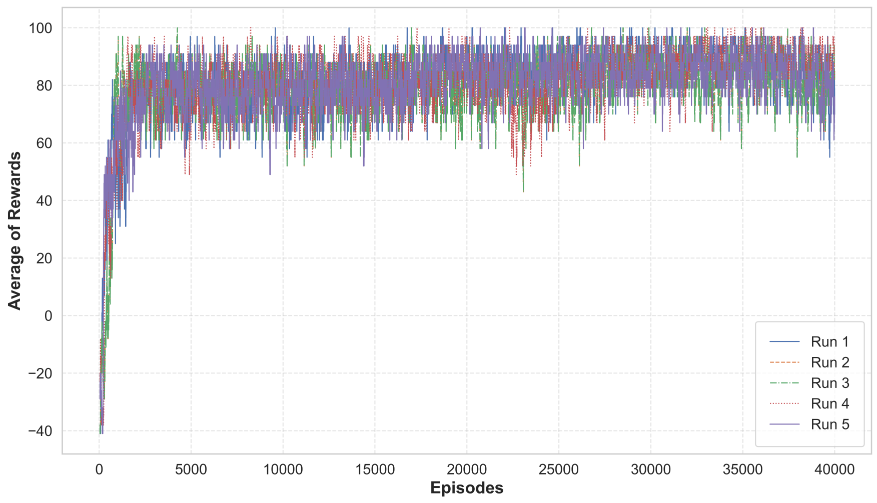

Figure 7.3 Average of rewards across episodes with $\epsilon = 0.2$ for five runs

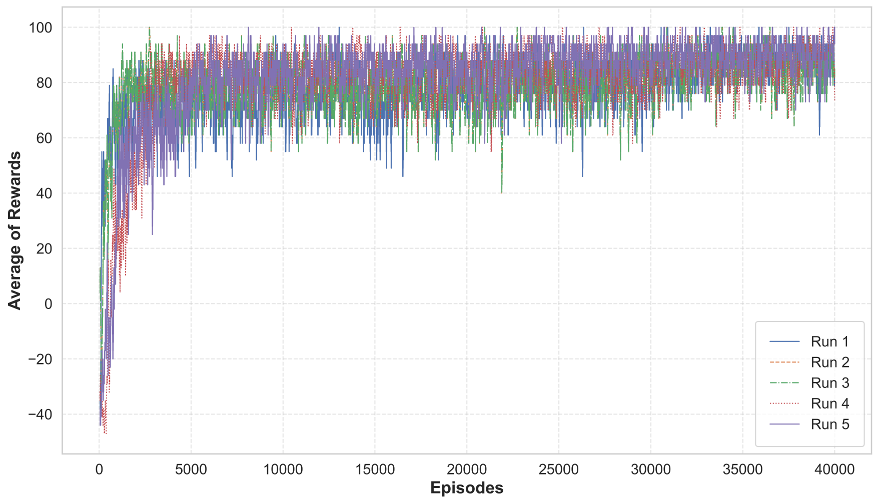

Figure 7.4 Average of rewards over 40,000 episodes with $\epsilon = 0.5$ for five runs

exploitation trade-off on the performance of the SecuEdge-DRL at the edge gateway. We tested three epsilon values ($\epsilon = 0.2$, $\epsilon = 0.5$, and $\epsilon = 1.0$), each run five times, as shown

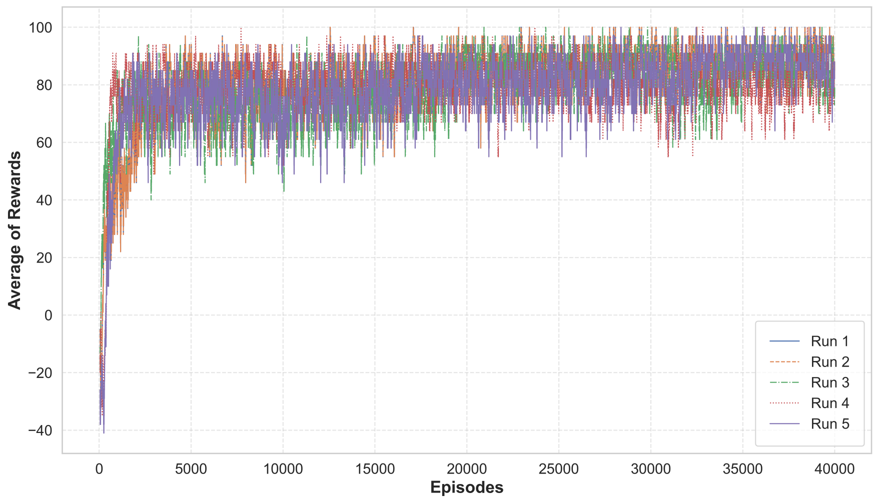

Figure 7.5 Average of rewards over 40,000 episodes with $\epsilon = 1.0$ for five runs

in Figures 7.3, 7.4, and 7.5. The epsilon parameter controls the balance between exploring new strategies and exploiting learned policies [272], which is key for edge-based IoT security where both response time and accuracy are crucial.

As illustrated in Figure 7.3, for $\epsilon = 0.2$ (), the system mainly exploited learned strategies, leading to rapid convergence and stable rewards between 70-90, ideal for real-time threat mitigation. With $\epsilon = 0.5$ (Figure 7.4), the system balanced exploration and exploitation, resulting in more significant reward variability but converging in a similar range, suitable for more dynamic threat environments. Finally, $\epsilon = 1.0$ (Figure 7.5) produced a highly exploratory system, where rewards fluctuated significantly, and convergence was slower, making it less suitable for an immediate response but potentially useful for detecting new vulnerabilities. Lower epsilon values ($\epsilon = 0.2$) provided faster convergence and stability, which are essential for IoT edge security, while higher values ($\epsilon = 1.0$) favored exploration at the cost of delayed system stabilization.

Table 7.2 Average Time Comparison Across Epsilon Values

Run	**Epsilon 0.1**	**Epsilon 0.2**	**Epsilon 0.5**
Run 1	0.0939	0.0933	0.0938
Run 2	0.0929	0.0960	0.0973
Run 3	0.0968	0.0957	0.0975
Run 4	0.0950	0.0976	0.0969
Run 5	0.0973	0.0962	0.0967

In Table 7.2, we present the comparison of execution time across different values of ϵ ($\epsilon = 0.1$,

$\epsilon = 0.2$, and $\epsilon = 0.5$) averaged over five runs. The results show that for $\epsilon = 0.1$, the system achieves the most stable and efficient performance, with minimal variability (ranging between 0.0939 and 0.0973 seconds). This highlights the system's ability to exploit learned security policies effectively at the edge, ensuring rapid threat detection without imposing significant computational overhead on constraints at the edge gateway.
For $\epsilon = 0.2$, the system demonstrates a slight increase in variability training execution time, reflecting a balanced approach between exploration and exploitation. Despite more exploration, the system maintains efficiency, indicating it can dynamically adapt its defense strategies while providing real-time security at the edge. Moreover, with $\epsilon = 0.5$, greater exploration introduces higher variability in the average times. However, the performance remains within acceptable limits for real-time operations, suggesting that the SecuEdge-DRL can handle increased exploration for adapting to evolving threats at the edge without severely impacting its response time.

7.6.2 Model Performance Evaluation

Table 7.3 presents the performance metrics of the SecuEdge-DRL, demonstrating the accuracy of 92.54%. This reflects the model's strong ability to mitigate cyber threats in IoT systems. Additionally, the model achieves high precision (95.73%) and F1 score (93.88%), ensuring reliable detection with minimal false positives or negatives. Figure 7.6 shows the confusion matrix, highlighting the model's prediction capabilities across different attack types such as DDoS, DoS GoldenEye, DoS Hulk, and Port scanning. In comparison, the authors in [98] reported a slightly higher accuracy of 97%, but their model used 83 features, while our model achieves competitive performance with a much smaller feature set. Similarly, [110] reported an accuracy of 91.02%, with recall (94.56%) and F1 score (94.56%), using only eight features. Although the SecuEdge-DRL accuracy is slightly lower than [98], our model balances accuracy and feature efficiency, offering robust performance across various evaluation metrics.

We do not report F1, precision, or recall for [98] in Table 7.3 because these metrics were not provided in their publication. Their evaluation was limited to accuracy, which makes a direct comparison of these additional metrics unavailable. The review includes these metrics to give a more detailed assessment of the model's performance.

7.6.3 Real-World Testbed Evaluation

We implemented a real-world testbed using IoT devices to evaluate the SecuEdge-DRL detecting DDoS, DoS GoldenEye, DoS Hulk attacks, and Port scanning. We designed the

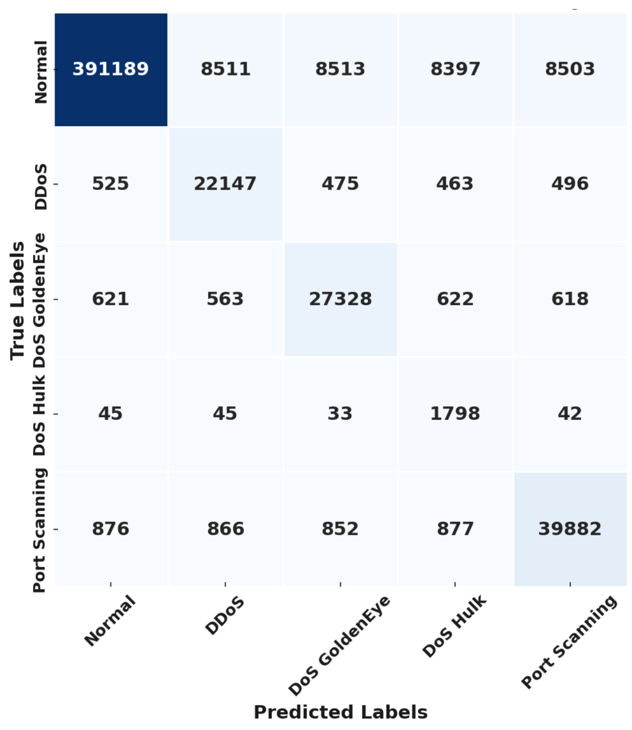

Figure 7.6 Matrix confusion of the proposed DRL-based IDS

Table 7.3 Comparison of performance metrics

Reference	**Accuracy**	**F1 Score**	**Precision**	**Recall**
SecuEdge-DRL	0.9254	0.9388	0.9573	0.9254
[110]	0.9102	0.9456	0.9458	0.9454
[98]	0.97	-	-	-

experiments to analyze and assess the SecuEdge-DRL from various perspectives, including its ability to enforce security policies while minimizing resource costs under these specific IoT-based attacks. We chose a real-world testbed because attackers often exploit the large number and highly distributed nature of IoT devices to launch stealthy, low-rate attacks, making detection challenging. By simulating these scenarios in a controlled environment, we directly assessed the robustness and efficiency of SecuEdge-DRL in identifying and mitigating such threats effectively. The testbed implements a typical IoT edge network, as shown in Figure 8.5, including 6 NodeMCUs, two Raspberry Pi 4 Model B units functioning as edge nodes, one laptop device, one workstation, one router, one physical server, and one hacker. Each Raspberry Pi 4 Model B is equipped with 8GB of RAM and powered by a 1.5GHz 64-bit quad-core CPU, providing a robust foundation for simulating complex networks that mirror real-world scenarios. The NodeMCUs and laptops are deployed as IoT devices across

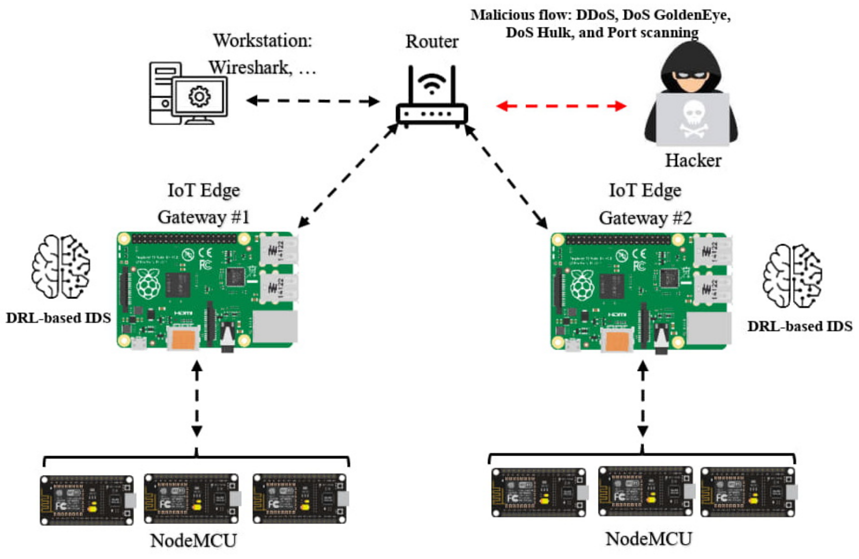
Malicious flow: DDoS, DoS GoldenEye, DoS Hulk, and Port scanning
Workstation: Wireshark, ...
Router
Hacker
IoT Edge Gateway #1
IoT Edge Gateway #2
DRL-based IDS
DRL-based IDS
NodeMCU
NodeMCU

Figure 7.7 Overview of real-world testbed.

two separate IoT domains, connecting with the edge gateways. The Raspberry Pi units are the IoT edge gateways on which the SecuEdge-DRL is deployed. The physical server functions as an edge server in the IoT network, hosting the DRL-based network models. The simulated hacker utilizes Kali Linux [273] to launch attacks. Meanwhile, the workstation performs packet capture and analysis using Wireshark software, ensuring network traffic monitoring and security performance. We analyzed energy consumption, CPU usage, and CPU load using the Mann-Whitney U Test for each security policy under real-time cyber threats. The null hypothesis in each case assumes no significant difference in the system's resource usage between the baseline (policy inactive) and the active state (policy enabled). Rejecting the null hypothesis indicates a statistically significant impact of the security policy on the respective resource metric, following established methodologies in prior research [206]. This approach helped us to identify and quantify the trade-offs between maintaining robust security measures and managing system performance effectively.
These experiments are essential for fine-tuning the SecuEdge-DRL to ensure it delivers high security while maintaining optimal performance at the edge of the IoT network. Additionally, we formulated all security policies in Table 3.

7.6.4 Energy Consumption Analysis

Table 7.5 reports the p-values from the Mann-Whitney U test and Cliff's δ effect size for measuring the energy consumption associated with each policy. Additionally, the confidence

Table 7.4 Policies and Their Corresponding Codes for Different Attacks

Attack	**Policy**	**Code**
DoS Hulk	Blocking IP address	P_1
DoS Golden Eyes	Turning on the firewall	P_2
DDoS	Resetting the system	P_3
Port Scanning	Closing the ports	P_4

Table 7.5 Policies and their corresponding p-values, effect sizes, and confidence intervals on energy consumption.

Policy	**p__value**	**Effect size**	**Confidence intervals (95%)**
Blocking IP address (P_0 VS. $P1$)	0.05 >	0.813	0.476, 1.150
Turning on the firewall (P_0 VS. $P2$)	0.05 >	0.969	0.613, 1.325
Resetting the system (P_0 VS. $P3$)	0.05 >	1.000	0.642, 1.358
Closing the ports (P_0 VS. $P4$)	0.05 >	0.862	0.522, 1.202

intervals provide a range for effect size estimation, ensuring statistical reliability and reinforcing the observed impact of security policies on energy consumption. The findings compare the average energy consumption over fifteen independent runs while implementing four distinct policies against their respective baseline conditions (i.e., when the policy is active or deactivated). We reject the null hypothesis based on the p-values in Table 7.5, as all are less than 0.05. This indicates a statistically significant difference in energy consumption for the policies P_1, P_2, P_3, and P_4 when active. The effect sizes provide further insights: P_1 (Blocking IP address) shows an effect size of 0.813, suggesting a moderate increase in energy consumption; P_2 (Turning on the firewall) has an effect size of 0.969, indicating a negligible change in energy consumption, demonstrating minimal impact; P_3 (Resetting the system) reflects an effect size of 1.0, indicating a higher energy requirement due to the resource-intensive nature of system resets; and P_4 (Closing ports) shows an effect size of 0.862, reflecting a moderate increase in energy usage. These results confirm that while each policy affects energy consumption, the increases remain within acceptable limits. All policies significantly enhance system security operations without imposing undue energy burdens, demonstrating the SecuEdge-DRL's suitability for resource-constrained edge gateway. Furthermore, the confidence interval reinforces the reliability of these measurements, accounting for variations across independent runs. This additional statistical validation strengthens the interpretation of the findings, confirming the impact of security policies on energy consumption while maintaining system efficiency.

As shown in Figure 7.8, the energy consumption during regular operation and various cyberattacks (DoS Hulk, DoS GoldenEye, DDoS, and Port scanning) highlights the impact of security policies on the performance of the SecuEdge-DRL. Under normal conditions, energy consumption is around 2.1 j, with minimal fluctuations, as a baseline for detecting cyber threats. During DoS Hulk and DoS GoldenEye attacks, energy usage increases to 2.6 j, indicating that additional resources are required to mitigate these attacks. For DDoS attacks, energy consumption remains around 2.6 j but shows more significant variability, reflecting the complexity of detecting these distributed attacks. In the case of Port scanning, energy consumption rises slightly to about 2.4 j, lower than during DoS attacks, as Port scanning is typically a precursor to more severe threats.

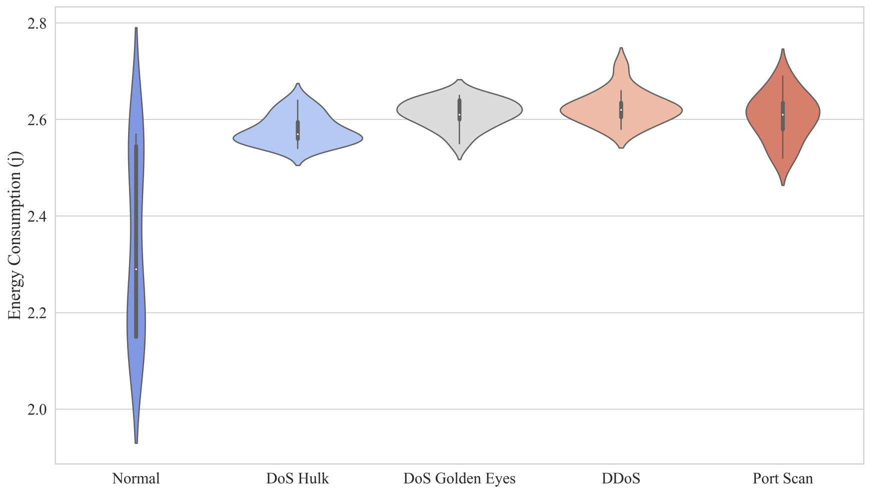

Figure 7.8 Energy consumption under a different policy.

CPU Usage Analysis

Table 7.6 Policies and their corresponding p-values, effect sizes, and confidence intervals on CPU usage.

Policy	**p_value**	**Effect size**	**Confidence intervals (95%)**
Blocking IP address (P_0 VS. $P1$)	0.05 >	0.747	0.68, 0.81
Turning on the firewall (P_0 VS. $P2$)	0.05 >	0.64	0.58, 0.72
Resetting the system (P_0 VS. $P3$)	0.05 >	0.1	0.05, 0.15
Closing the ports (P_0 VS. $P4$)	0.05 >	0.916	0.89, 0.94

Table 7.6 displays the p-values derived from the Mann-Whitney U test along with Cliff's δ effect size, assessing the impact of each security policy on CPU usage during attack detection. Additionally, confidence intervals provide a measure of reliability, offering deeper insight into the variations introduced by different policies. The results contrast the average CPU usage observed during the implementation of four distinct policies against each security policy's

baseline condition (e.g., active or deactivated). According to the p-values in Table 7.6, we reject the null hypothesis. Based on the data, the evidence suggests that there is a statistically significant difference in the average CPU usage by P_1, P_2, P_3, and P_4 when the security policy is active. Moreover, the effect sizes 0.747 for P_1 and 0.916 for P_4 indicate a large impact on CPU usage, while the effect size of 0.64 for P_2 shows a medium impact, and 0.1 for P_3 suggests a small impact on CPU usage by these policies. Moreover, the confidence intervals confirm that policies with larger effect sizes consistently impose more significant computational overhead, while those with moderate effect sizes maintain a more controlled impact. This statistical validation reinforces the reliability of these security measures, demonstrating that the observed trade-off between security enforcement and system performance remains within an acceptable and predictable range.

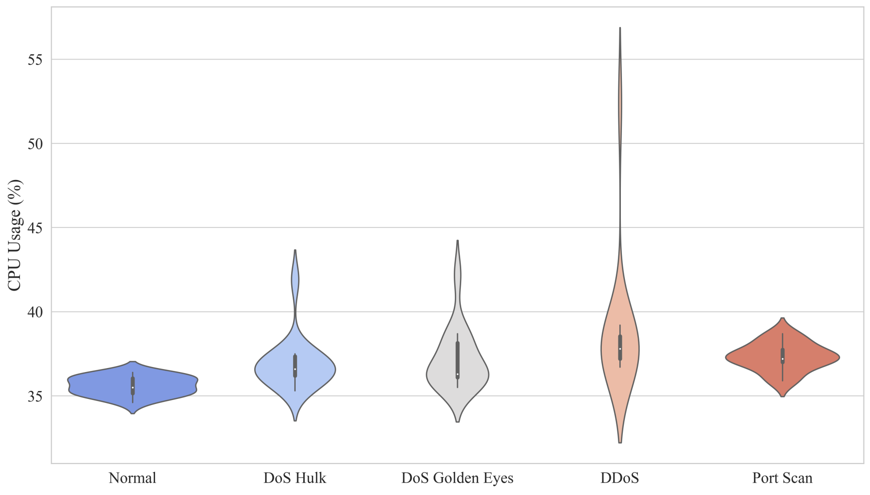

Figure 7.9 CPU Usage under a different policy.

Figure 7.9 illustrates CPU usage during regular operation and various attacks, showcasing the impact of security policies on the efficiency of the SecuEdge-DRL. Under regular operation, CPU usage is at its lowest, around 35%, with minimal fluctuation, serving as a baseline for comparison during attacks. During DoS Hulk and DoS GoldenEye attacks, CPU usage increases to around 38-40%, reflecting additional processing power required to mitigate these threats while maintaining moderate efficiency. In DDoS attacks, CPU usage rises significantly to 45-50%, with a broader distribution, indicating the complexity and demand of handling these attacks. The SecuEdge-DRL effectively allocates resources to manage the increased workload. CPU usage is slightly elevated for Port scanning, around 36-38%, requiring less computational effort than DoS attacks. This demonstrates the system's ability to handle reconnaissance activities efficiently without overloading the CPU.

CPU Load Analysis

Table 7.7 Policies and their corresponding p-values, effect sizes, and confidence intervals on CPU load.

Policy	**p_value**	**Effect size**	**Confidence intervals (95%)**
Blocking IP address (P_0 VS. $P1$)	$0.05 >$	0.507	0.42, 0.58
Turning on the firewall (P_0 VS. $P2$)	$0.05 >$	0.1	0.02, 0.18
Resetting the system (P_0 VS. $P3$)	$0.05 >$	0.431	0.35, 0.51
Closing the ports (P_0 VS. $P4$)	$0.05 >$	0.431	0.35, 0.51

Table 7.7 reports the p-values from the Mann-Whitney U test and Cliff's δ effect size, assessing the influence of each security policy on CPU load. Moreover, the confidence intervals enhance the statistical reliability of these findings by quantifying the range of variation introduced by different policies. The results compare the average CPU load observed when applying four distinct security policies against their respective baseline conditions (i.e., active vs. deactivated). Moreover, we reject the null hypothesis according to the p-values in Table 7.7. Based on the data, the evidence suggests that there is a statistically significant difference in the average CPU load by P_1, P_2, P_3 and P_4 when the security policy is active. Moreover, the effect sizes 0.507 for P_1 indicating a moderate impact on CPU load, 0.1 for P_2 suggesting a small impact, and 0.431 for both P_3 and P_4 also indicating moderate impacts demonstrate varying levels of impact on CPU load by these policies. Additionally, the confidence intervals confirm that policies with larger effect sizes contribute noticeably to CPU load fluctuations, while policies with smaller effect sizes show a minimal and predictable impact. This statistical validation strengthens the reliability of these security mechanisms, indicating that the trade-off between security and performance remains within a well-defined and manageable range.

Figure 7.10 presents CPU load during regular operation and attacks, demonstrating the impact of security policies on CPU load and the efficiency of the DRL-based IDS. During regular operation, CPU load remains steady at around 35-37%, with minimal fluctuation, as a baseline for evaluating CPU load under attack scenarios. In the case of DoS Hulk attacks, CPU load slightly increases to 35-38%, reflecting the additional processing required to mitigate the attack, though the system maintains efficient performance. Similarly, for DoS GoldenEye attacks, CPU load centers around 35-36%, indicating that while the system expends more resources, it efficiently handles these threats with only a moderate increase in load. DDoS attacks lead to a more pronounced rise in CPU load, reaching around 45%, with wider fluctuations, reflecting the complexity of detecting and mitigating distributed attacks. This shows the IDS's ability to handle demanding situations effectively. For Port scanning attacks, CPU load slightly rises to 37-39%, lower than during DoS attacks. Since Port scanning is often a precursor to more significant threats, the system efficiently handles

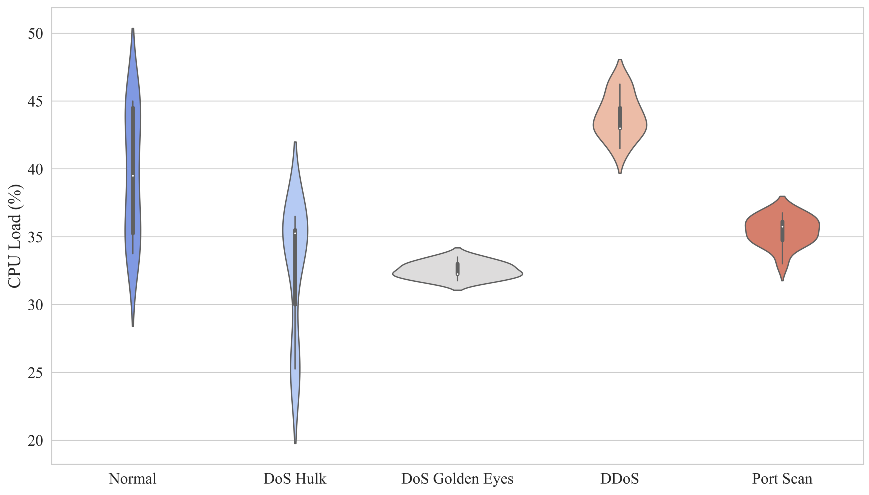

Figure 7.10 CPU Load under a different policy.

the increased load without overtaxing resources.

Evaluation of self-adaptation IDS under real-time experiment

Table 7.8 Performance Comparison of Proposed Model Under Real-Time Attacks

Reference	**Accuracy**	**F1 Score**	**Precision**	**Recall**
Proposed Model	0.9254	0.9388	0.9573	0.9254
Proposed Model (Real-Time)	0.9075	0.9163	0.9245	0.9114

The SecuEdge-DRL demonstrates consistency between the training phase and real-world deployment on edge under real-time attacks. As shown in Table 7.8, the model maintains a high accuracy of 92.54%, along with solid precision (95.73%) and F1-score (93.88%) under regular conditions. When tested under real-time attack conditions, the model still delivers robust performance with an accuracy of 90.75%, F1-score of 91.63%, and a precision of 92.45%. Although there is a slight reduction in precision and F1 scores in real-world conditions, the overall performance remains acceptable. This slight decrease reflects the challenges of real-time attacks and underscores the model's resilience and practical effectiveness at the edge of IoT systems.

7.6.5 Comparative Analysis of Energy Efficiency

This section compares the SecuEdge-DRL framework with Bouhamed et al. [274], which proposed a periodic DRL-based IDS for UAVs. SecuEdge-DRL maintains an average energy consumption of 2.6J per detection while ensuring real-time adaptive learning, allowing it to respond instantly to evolving threats. In contrast, [274] employs periodic offline learning, optimizing power usage by updating IDS policies only during UAV docking intervals, leading to lower energy fluctuations and a 7.4% increase in operational time. However, this approach risks delayed threat detection, leaving systems vulnerable between updates. SecuEdge-DRL mitigates this risk through continuous learning, maintaining proactive defense capabilities while balancing security and energy efficiency. Its DRL-driven decision-making ensures dynamic adaptation, making it more suitable for highly dynamic at the edge where real-time security responses are critical.

7.7 Discussion

SecuEdge-DRL is built on a centralized DRL architecture, where decision-making occurs at the edge server while security policies are enforced at edge gateways. This design enables real-time threat detection with minimal computational overhead. However, multi-edge environments introduce challenges such as inconsistent data distributions, communication delays, and varying resource constraints, requiring further enhancements. Policy randomization dynamically alters security actions for adversarial robustness to prevent attackers from predicting and bypassing defenses. For example, in response to port scanning attempts, SecuEdge-DRL may randomly block different ports, enforce temporary firewalls, or deploy decoy mechanisms, increasing attack resistance. Instead of relying on predefined attack signatures, anomaly-based detection continuously monitors network behavior, authentication failures, protocol misuse, and abnormal resource consumption. Dynamic detection thresholds adapt to evolving attack patterns, enabling real-time identification of zero-day threats. Moreover, DRL-driven anomaly scoring prioritizes high-risk deviations for faster mitigation. For continuous learning in a multi-edge environment, hierarchical DRL ensures local edge models train on real-time data and periodically sync with a global model, preserving learned knowledge while preventing catastrophic forgetting. Transfer learning allows trained models to adapt to new environments without extensive retraining. To maintain efficiency on resource-constrained edge gateways, quantized DRL models optimize inference while minimizing computational costs. Furthermore, this study demonstrates the effectiveness of the SecuEdge-DRL at the IoT edge, balancing security, CPU usage, and energy consumption. Integrating the MAPE-K framework, the SecuEdge-DRL dynamically adapts its policies to various threat scenarios, optimizing security and resource efficiency. The findings show that the SecuEdge-DRL handles diverse cyber threats effectively, adjusting actions in real-time based on the threat level to conserve energy and prevent CPU overload—critical for resource-constrained IoT edge gateway. Also, this research directly aligns with several United Nations Sustainable Development Goals (SDGs): [269] SDG 9 (resilient infrastructure through IoT security), SDG 12 (responsible resource consumption via energy efficiency), and SDG 11 (secure, sustainable smart cities), highlighting the role of secure IoT systems in global sustainability. While promising, the SecuEdge-DRL's effectiveness relies on diverse training interactions, posing challenges in new environments with limited threat exposure. The correlation-based feature selection improves detection accuracy by minimizing interdependent features, though the experimental setup may not fully emulate real-world IoT complexity. Future work will expand training data and optimize the DQN model to enhance energy efficiency further and reduce computational demands, contributing to the long-term sustainability of IoT systems. This approach aligns with global efforts to develop resilient, energy-conscious infrastructure, supporting sustainable development goals. Deploying SecuEdge-DRL in real-world IoT networks involves considerations such as interoperability with diverse platforms, where adapting to different hardware architectures and communication protocols ensures seamless integration. Efficient decision-making at the edge remains essential to maintaining real-time threat detection while optimizing computational efficiency. Additionally, regulatory and privacy frameworks must be considered to align security policies with data protection requirements. These factors highlight practical implementation aspects while reinforcing the adaptability and robustness of SecuEdge-DRL.

7.8 Chapter Summary

This chapter presents the SecuEdge-DRL, a system tailored for IoT edge environments. It combines DRL with the DQN algorithm and the MAPE-K framework for self-adaptive and energy-efficient threat detection. The system dynamically adjusts to real-time conditions and evolving threats, enhancing the detection and mitigation of cyberattacks (e.g., DDoS, DoS GoldenEye, DoS Hulk, and Port scanning) without relying on predefined data models. Real-time evaluations on a Raspberry Pi-based IoT testbed demonstrated the SecuEdge-DRL's robustness. The system achieved high detection accuracy while imposing minimal resource overhead on energy-constrained devices. The system proved effective in real-time scenarios and controlled training environments, maintaining operational efficiency and ensuring resilience against diverse cyber threats. Furthermore, the SecuEdge-DRL's design aligns with sustainability objectives by optimizing energy consumption and computational demands, contributing to the long-term sustainability of IoT ecosystems. Statistical analyses validated its ability to balance robust security with resource efficiency, highlighting its potential to support energy-conscious, resilient infrastructure and advance global sustainable development goals.

CHAPTER 8 ARTICLE 6: DEEP REINFORCEMENT LEARNING-BASED INTRUSION DETECTION SYSTEM: DEFENDING EDGE GATEWAYS AGAINST MIRAI AND GAFGYT

8.1 Article Metadata

- **Title:** Deep Reinforcement Learning-Based Intrusion Detection System: Defending Edge Gateways Against Mirai and Gafgyt

- **Authors:** Saeid Jamshidi, Amin Nikanjam, Kawser Wazed Nafi, Foutse Khomh
- **Affiliations:**
  - SWAT Laboratory, Polytechnique Montréal, Quebec, H3T 1J4, Canada
  - Huawei Distributed Scheduling and Data Engine Lab, Canada

- **Submitted Date:** 10 March 2025
- **Accepted Date:** 24 May 2025
- **Conference:** *IEEE FiCloud 2025*

8.2 Chapter Overview

The rapid growth of IoT has transformed industries, resulting in unprecedented opportunities alongside significant cybersecurity challenges. Malware, such as Mirai and Gafgyt, exploits vulnerabilities in IoT devices, resulting in large-scale attacks. Traditional IDSs struggle to detect these evolving threats due to their reliance on static rule-based or classic ML models, which lack adaptability to zero-day attacks and dynamic traffic patterns. This chapter presents EdgeShield-DRL, a novel DRL-based Intrusion Detection System (IDS) designed for IoT edge gateways. EdgeShield-DRL dynamically detects and mitigates evolving threats in real-time while ensuring efficient operation on resource-constrained edge devices. We evaluated EdgeShield-DRL on the N-BaIoT dataset, achieving a high detection accuracy of 97% during training phases and 96% in real-time detection scenarios. Moreover, the system demonstrates robust resource efficiency, maintaining minimal energy consumption and carbon emissions even under attack conditions. Experiments on a real-world testbed further validate EdgeShield-DRL's effectiveness, showcasing resilience against diverse attack scenarios, including large-scale botnet activity. Furthermore, EdgeShield-DRL effectively balances robust security with resource constraints, making it particularly suitable for critical IoT applications, e.g., smart cities, healthcare, and industrial automation.

8.3 Study Design

The exponential growth of the IoT has driven innovations across industries, e.g., smart cities, healthcare, and industrial automation [275]. Moreover, IoT ecosystems have become integral to modern infrastructure, with billions of interconnected devices exchanging sensitive data. However, this unprecedented connectivity has introduced a vast attack surface, exposing IoT devices to sophisticated cyber threats [276]. Among these, Mirai and Gafgyt attacks [277] [278] [279] stand out because they can compromise IoT devices through weak authentication protocols and software vulnerabilities, enabling the attackers to launch large-scale DDoS attacks [280]. These attacks have caused widespread service disruptions, underlining the urgent need for robust IDS for IoT systems [281].
While traditional ML-based IDS frameworks can effectively detect known attack patterns, they face significant limitations in detecting attacks in dynamic IoT systems [268] [282]. Furthermore, ML models rely on pre-collected datasets, making them less effective against evolving threats and zero-day attacks [283] [284]. Additionally, their static nature necessitates frequent retraining and fine-tuning, which is computationally expensive and impractical for resource-constrained devices, such as edge gateways. To overcome these challenges, DRL offers a dynamic and adaptive approach, enabling IDS to learn optimal detection policies autonomously through continuous interaction with its environment [285]. This makes DRL particularly suitable for real-time detection of evolving attack vectors, e.g., Mirai and Gafgyt.
In this study, we deploy a DRL-based IDS on an edge gateway, a pivotal component in IoT systems. Edge gateways are designed to process data locally, reduce latency, and enhance security by acting as the first line of defense before transmitting data to the cloud [286]. However, security at the edge remains challenging due to limited computational resources, making it difficult for traditional IDS to detect and mitigate evolving threats, e.g., Mirai and Gafgyt.
We deploy DRL-based IDS on the edge gateway to demonstrate its feasibility in constrained environments and provide insights into the system's behavior under specific attack scenarios.
Moreover, focusing on Mirai and Gafgyt, two of the most notorious malware families responsible for large-scale IoT disruptions, this study examines the impact of these attacks on system performance and resource utilization. Beyond conventional performance metrics, e.g., IDS detection accuracy and response time, this study measures additional critical metrics, including energy consumption, CPU usage, memory usage, and carbon emissions, addressing the environmental implications of deploying IDS solutions in IoT systems. This research di-

rectly aligns with several United Nations Sustainable Development Goals (SDGs): [44] SDG 9 (resilient infrastructure through IoT security), SDG 12 (responsible resource consumption via energy efficiency), and SDG 11 (secure, sustainable smart cities), highlighting the role of secure IoT systems in global sustainability. Since edge devices frequently operate in energy-limited environments and must balance security with efficiency, ensuring that IDS solutions remain lightweight and sustainable is critical. This study analyzes the trade-offs among these factors to lay the groundwork for designing secure, resource-efficient, and environmentally sustainable IDS solutions for IoT deployments. To summarize, this study makes the following key contributions:

- **Security-focused DRL-based IDS for edge gateways:** A novel IDS leveraging DRL to dynamically detect and mitigate evolving IoT threats, such as Mirai and Gafgyt, ensuring real-time adaptability and robust security.
- **Evaluation of performance and environmental metrics:** This study measures key metrics, including energy consumption, carbon emissions, detection accuracy, and response time, to assess our proposed DRL-based IDS's efficiency and sustainability at the edge gateway.

8.4 EdgeShield-DRL

EdgeShield-DRL is an adaptive IDS designed to secure IoT edge gateways against dynamic and sophisticated cyber threats, including Mirai and Gafgyt botnet attacks. The system leverages the capabilities of DRL to autonomously detect, mitigate, and adapt to emerging threats while optimizing resource utilization at the edge. By operating locally on the edge gateway, EdgeShield-DRL minimizes response times and reduces reliance on centralized systems, making it suitable for real-time IoT environments. The system architecture is depicted in Fig. 8.1.

8.4.1 Security-Oriented Reinforcement Learning Formulation

EdgeShield-DRL formulates the IDS task as a Markov Decision Process (MDP) [287] to model the dynamic interaction between the IDS agent and the IoT environment. The MDP is defined as:

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma), \tag{8.1}$$

Where:

- $\mathcal{S}$: The state space represents real-time observations of the IoT systems, including network-level features (e.g., packet size, protocol type, and traffic volume) and device-level metrics (e.g., CPU utilization, memory usage, and detected anomalies).
- $\mathcal{A}$: The action space comprises security-specific responses, e.g., allowing traffic, blocking malicious packets, isolating compromised devices, or generating alerts.
- $\mathcal{P}(s'|s,a)$: The state transition probability, describing the likelihood of transitioning from state s to s' when action a is taken.
- $\mathcal{R}(s,a)$: The reward function evaluates the security impact of the chosen action. For instance, correctly blocking malicious traffic yields a positive reward, while false positives incur penalties.
- γ: The discount factor, controlling the trade-off between immediate rewards (e.g., rapid mitigation) and long-term benefits (e.g., system stability).

The objective of the DRL agent is to learn an optimal policy $\pi^*(a|s)$ that maximizes the expected cumulative reward:

$$G_t = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t \right]. \tag{8.2}$$

8.4.2 Deep Q-Learning for Threat Mitigation

EdgeShield-DRL employs Deep Q-Learning (DQL) to approximate the optimal action-value function, $Q^*(s,a)$, which quantifies the long-term value of taking action a in state s:

$$Q^*(s,a) = \max_\pi \mathbb{E}\left[G_t \mid s_t = s, a_t = a, \pi\right]. \tag{8.3}$$

A deep neural network (DNN) parameterized by θ approximates $Q(s,a;\theta)$. The agent selects the optimal action a_t at time t by:

$$a_t = \arg\max_a Q(s_t, a; \theta). \tag{8.4}$$

The DNN is trained by minimizing the loss function:

$$\mathcal{L}(\theta) = \mathbb{E}\left[(y_t - Q(s_t, a_t; \theta))^2 \right], \tag{8.5}$$

Where the target Q-value is:

$$y_t = r_t + \gamma \max_{a'} Q(s_{t+1}, a'; \theta^-). \tag{8.6}$$

Here, θ^- represents the parameters of a target network, which is periodically updated to ensure training stability and convergence.

8.4.3 Workflow at the Edge Gateway

The workflow of EdgeShield-DRL is centered on real-time security and adaptability, consisting of the following steps:

1. **State Observation:** The agent observes the current state s_t of the IoT systems, capturing real-time traffic features (e.g., protocol anomalies, packet sizes) and device metrics (e.g., CPU and memory usage).
2. **Action Selection:** Using the DNN, the agent computes Q-values for all actions and selects the optimal action:
$$a_t = \arg\max_{a} Q(s_t, a; \theta). \tag{8.7}$$
3. **Threat Mitigation:** The selected action a_t is executed to address the identified threat. For example, malicious traffic may be blocked, or a compromised device will be quarantined.
4. **Reward Feedback:** The environment evaluates the effectiveness of the action and provides a reward r_t, which is used to guide the agent's future decisions.
5. **Policy Update:** The agent updates its Q-value network using the observed transition (s_t, a_t, r_t, s_{t+1}), refining its policy over time.

Figure 8.1 illustrates the architecture of EdgeShield-DRL. The system operates on IoT edge gateways by observing network and device states, computing Q-values for security actions, and executing optimal responses. The continuous feedback loop ensures iterative policy improvement, enhancing real-time security and adaptability to new threats.

8.4.4 Algorithm Overview

Algorithm 10 outlines the EdgeShield-DRL method for detecting and mitigating Mirai and Gafgyt attacks using DQN. The algorithm trains the IDS to maximize long-term rewards

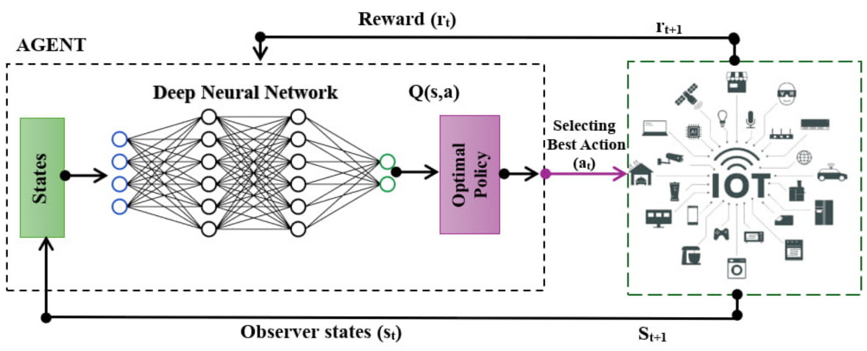

Figure 8.1 Architecture of the EdgeShield-DRL Method for the edge gateway.

through interactions with the edge environment, leveraging an experience replay buffer and a target network for stability.

Initialization

The replay buffer D with capacity N is initialized to store transitions (s_t, a_t, r_t, s_{t+1}). The action-value function $Q(s, a; \theta)$ and target network $Q_{\text{target}}(s, a; \theta^-)$ are initialized with parameters θ and $\theta^- = \theta$, respectively.

Training Process

Each episode begins with an initialization of the edge environment, where the agent observes the initial state s_0 (e.g., network traffic patterns, device metrics). At each time step t:

- **Action Selection:** An action a_t is chosen using an ϵ-greedy policy:

$$a_t = \begin{cases} \text{random action}, & \text{if random}(0,1) < \epsilon, \\ \arg\max_a Q(s_t, a; \theta), & \text{otherwise.} \end{cases}$$

- **Environment Interaction:** Action a_t is executed, yielding a reward r_t and a new state s_{t+1}.

- **Replay Buffer Update:** The transition (s_t, a_t, r_t, s_{t+1}) is stored in D.

Experience Replay

When the replay buffer D contains at least B transitions, a minibatch is sampled to compute target Q-values:

$$y = \begin{cases} r, & \text{if } s' \text{ is terminal,} \\ r + \gamma \max_{a'} Q_{\text{target}}(s', a'; \theta^-), & \text{otherwise.} \end{cases}$$

The network parameters θ are updated by minimizing the loss:

$$\mathcal{L}(\theta) = \frac{1}{B} \sum (y - Q(s, a; \theta))^2 .$$

Target Network and Exploration Decay

The target network Q_{target} is periodically updated as $\theta^- \leftarrow \theta$. The exploration probability ϵ decays over time:

$$\epsilon \leftarrow \max(\epsilon_{\text{min}}, \epsilon \cdot \epsilon_{\text{decay}}).$$

Termination

The training continues until a terminal state is reached (e.g., attack mitigated or device isolated). The trained action-value function $Q(s, a; \theta)$ is then used for real-time detection and mitigation of Mirai and Gafgyt attacks.

8.5 Experimental setup

This section details the experimental approach for assessing the EdgeShield-DRL method deployed on an IoT edge gateway.

8.5.1 Research Questions:

- **How effectively is the EdgeShield-DRL method detecting and mitigating Mirai and Gafgyt attacks?** This question investigates EdgeShield-DRL's ability to accurately detect and mitigate Mirai and Gafgyt attacks, focusing on core metrics such as detection accuracy.
- **How does deploying EdgeShield-DRL impact the edge gateway regarding response time, resource utilization (CPU, memory), and environmental metrics, e.g., energy consumption and carbon emission?** This question explores the

practicality of deploying EdgeShield-DRL on edge gateways, measuring its impact on response time, resource utilization (CPU and memory), and environmental sustainability through energy consumption and carbon emission metrics.

8.5.2 Data Set Evaluation

The N-BaIoT [288] dataset, a benchmark for IoT intrusion detection, was used to evaluate EdgeShield-DRL. This dataset contains regular and attack traffic, including Mirai and Gafgyt patterns, with detailed features like packet sizes and flow durations. This dataset ensures robust evaluation by simulating real-world IoT scenarios, enabling validation against known and evolving attack vectors.

8.5.3 Feature Selection

We selected six key features from the N-BaIoT dataset using correlation analysis and recursive feature elimination (RFE). These features capture statistical and temporal traffic variations, enabling accurate detection of Mirai and Gafgyt attacks. This targeted approach ensures high detection performance while maintaining lightweight computations, making it suitable for resource-constrained edge gateways. The selected features are summarized in Table 8.1.

8.6 Results

To address RQ1, we conducted experiments to evaluate EdgeShield-DRL at the edge under real-time attack scenarios. This section details the experimental setup and methodology used for the evaluation.

8.6.1 Experiments and Evaluation

To evaluate the performance of the EdgeShield-DRL method, we conducted an exploratory study on the impact of different epsilon (ϵ) values $(0.4, 0.5, 1.0)$ to analyze the exploration-exploitation trade-off. This study assessed the model's ability to detect and mitigate Mirai and Gafgyt attacks at the edge gateway, demonstrating its robustness and adaptability in static and dynamic traffic conditions.
Figure 8.2, with reduced exploration ($\epsilon = 0.4$), the model stabilizes more quickly due to its reliance on exploiting existing knowledge. This allows the system to adapt rapidly to established attack signatures, making it particularly effective in environments where traffic

patterns and attack types are relatively static. However, the slower progression in early reward growth suggests that this approach may be less practical for highly variable or evolving attacks, such as new variants of Mirai and Gafgyt. Nonetheless, the final convergence underscores the model's ability to build reliable detection policies for consistent attack patterns.

Table 8.1 Selected Key Features from the N-BaIoT Dataset

Feature Name	Description
MI_dir_L5_variance	Variance of mutual information in traffic (L5)
MI_dir_L3_mean	Mean of mutual information in traffic (L3)
HpHp_L0.01_weight	Weight of histogram probability for flows
HpHp_L0.01_mean	Mean of histogram probability for flows
MI_dir_L1_variance	Variance of mutual information in traffic (L1)
HpHp_L0.01_covariance	Covariance of histogram probabilities

Algorithm 10: EdgeShield-DRL Method for Mirai and Gafgyt Detection

```
Input  : Replay buffer $D$ with capacity $N$, action-value function $Q(s, a; \theta)$,
         target network $Q_{\text{target}}(s, a; \theta^-)$, discount factor $\gamma$,
         exploration probability $\epsilon$, learning rate $\alpha$,
         batch size $B$, target update interval $T$
Output: Trained action-value function $Q(s, a; \theta)$ for real-time detection and mitigation of Mirai and Gafgyt attacks
1  Initialize: Replay buffer $D$
2  Initialize network parameters $\theta$ for $Q(s, a; \theta)$
3  Initialize target network parameters $\theta^- \leftarrow \theta$
4  for each episode do
5      Initialize edge environment and observe initial state $s_0$
6      for each time step $t$ do
7          if random(0, 1) $< \epsilon$ then
8              Select a random action $a_t \in \mathcal{A}$ (e.g., allow, block, isolate, alert)
9          else
10             Select action $a_t = \arg\max_a Q(s_t, a; \theta)$
11         end
12         Execute action $a_t$ and observe:
13             Reward $r_t$: Effectiveness of mitigating Mirai/Gafgyt-related activity
14             Next state $s_{t+1}$: Updated traffic features or device behavior
15         Store $(s_t, a_t, r_t, s_{t+1})$ in $D$
16         if len(D) $\geq B$ then
17             Sample minibatch of $B$ transitions $(s, a, r, s')$ from $D$
18             for each transition in minibatch do
19                 if $s'$ is terminal then
20                     $y = r$
21                 else
22                     $y = r + \gamma \max_{a'} Q_{\text{target}}(s', a'; \theta^-)$
23                 end
24                 Compute loss:
                       $\mathcal{L}(\theta) = \frac{1}{B} \sum (y - Q(s, a; \theta))^2$
                   Perform gradient descent update:
                       $\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}(\theta)$
25             end
26         end
27         if timestep % $T = 0$ then
28             Update target network: $\theta^- \leftarrow \theta$
29         end
30         Decay $\epsilon$:
               $\epsilon \leftarrow \max(\epsilon_{\text{min}}, \epsilon \cdot \epsilon_{\text{decay}})$
31         if $s_{t+1}$ is terminal then
32             break
33         end
34     end
35 end
```

Figure 8.3, the model operates under moderate exploration ($\epsilon = 0.5$), balancing its search for new detection strategies with the exploitation of learned policies. During the initial phases, the system experiences fluctuations in its performance as it generalizes attack patterns across varying traffic scenarios. Over time, the model stabilizes, indicating its success in identifying characteristic behaviors of attacks, e.g., Mirai's distributed DDoS traffic and Gafgyt's malicious payload injections. This stabilization demonstrates the model's capability to classify attack traffic confidently while minimizing false positives.

Figure 8.4 shows the learning process with higher exploration ($\epsilon = 1.0$), where the model prioritizes sampling diverse actions to handle previously unseen or evolving attack patterns better. This exploratory approach results in more significant variability in the early stages as the system evaluates various detection strategies. Despite this, the model converges to a stable policy, effectively detecting known and potentially novel attack behaviors. This highlights the importance of exploration for environments with highly dynamic and unpredictable IoT traffic.

8.6.2 Real-World Testbed Evaluation

Figure 8.5 depicts a testbed simulating a typical IoT edge network. The setup includes six NodeMCUs, two Raspberry Pi 4 Model B units functioning as edge nodes, a laptop, a workstation, a router, a physical server, and a simulated attacker. Each Raspberry Pi 4 Model B is equipped with 8GB of RAM and powered by a 1.5GHz 64-bit quad-core CPU, providing sufficient computational resources to replicate complex real-world networks. The NodeMCUs and the laptop represent IoT devices distributed across two distinct domains, communicating seamlessly with the edge gateways for security processing and data transfer.

The Raspberry Pi devices serve as IoT edge gateways hosting the DRL-based IDS, while the physical server operates as the edge server and runs the DRL-based IDS. A simulated attacker, configured with Kali Linux [185], generates diverse cyberattacks to evaluate the system's resilience. Simultaneously, the workstation captures network traffic using Wireshark, enabling detailed security analysis.

To assess EdgeShield-DRL's resource impact, metrics such as energy consumption, CPU usage, memory usage, IDS response time, and carbon emissions were analyzed using the Mann-Whitney U Test. The null hypothesis for each metric assumed no significant differences between baseline and test states. Rejecting the null hypothesis indicated a statistically significant impact of EdgeShield-DRL on the corresponding metric. This evaluation provided essential insights into the trade-offs between implementing robust security measures and maintaining efficient system performance, ensuring an optimal balance between effective

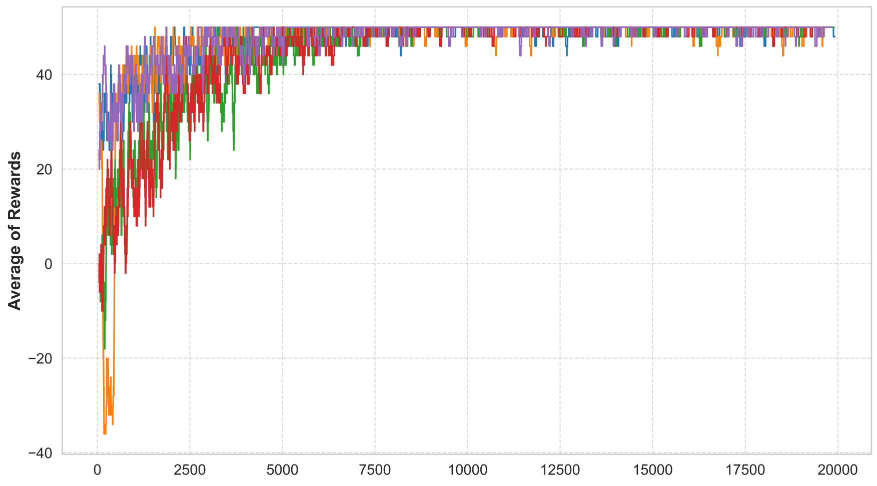

Figure 8.2 Learning performance of EdgeShield-DRL with $\epsilon = 0.4$, showing faster convergence for static attack scenarios.

threat mitigation and resource efficiency.

8.6.3 Dynamic Attack Detection and Pattern-Based Mitigation

The system logs in Figure 8.6 highlight the performance of the EdgeShield-DRL method during normal and attack conditions. In the normal state, key metrics, such as CPU usage (60%), memory usage (1520 MB), and energy consumption (3 Joules), remain stable, which demonstrates efficient operation. During attacks, e.g., Gafgyt and Mirai, the system detects malicious activity in real-time, as indicated by increased resource utilization (CPU up to 70%, energy consumption up to 3.5 Joules). The IDS maintains rapid response times under 0.3 seconds and effectively executes countermeasures, e.g., blocking suspicious IP addresses.

8.6.4 Performance Metrics: Training vs. Real-Time Testing

Table 8.2 presents the performance metrics of the EdgeShield-DRL method during training and real-time testing. The training phase achieved high values across all metrics, with an average detection accuracy of 0.97, an F1 score of 0.97, a recall of 0.98, and a precision of 0.97. In real-time testing, the metrics show a slight reduction due to real-world deployment challenges, such as varying traffic patterns and system constraints, with accuracy at 0.96 and precision at 0.95. This comparison highlights the model's robustness and ability to maintain effective intrusion detection under real-time conditions. The marginal drop in metrics underscores the importance of validating IDS models in practical scenarios.

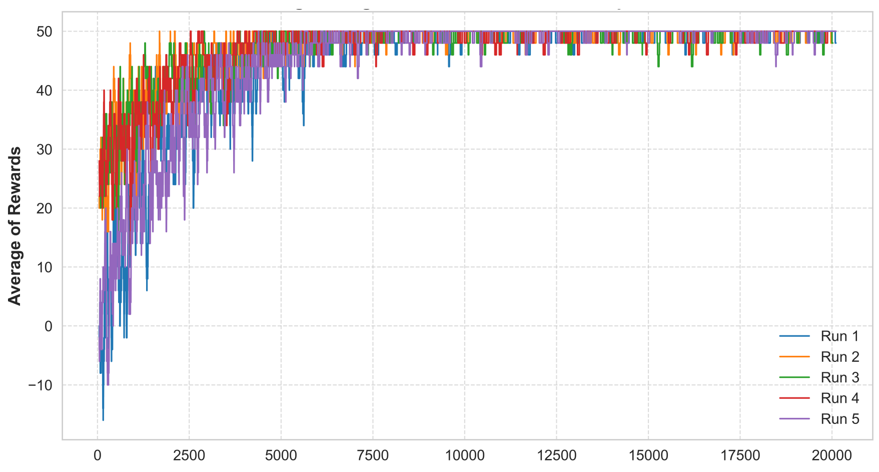

Figure 8.3 Learning performance of EdgeShield-DRL with $\epsilon = 0.5$, balancing exploration and exploitation effectively.

Table 8.2 Performance Metrics Comparison

Metrics	Training	Real-time Test
Accuracy	0.97	0.96
F1 Score	0.97	0.96
Recall	0.98	0.97
Precision	0.97	0.95

8.6.5 Model Performance Evaluation

To address RQ2, we measured resource utilization at the edge under real-time attack scenarios.

Figure 8.7 illustrates the energy consumption of the EdgeShield-DRL method under regular operation and during Mirai and Gafgyt attacks at the edge gateway. The system demonstrates stable energy consumption during normal conditions, reflecting its efficiency in standard operations. However, during attack scenarios, the energy usage increases slightly due to the additional computational load required for real-time detection and mitigation. Gafgyt attacks exhibit marginally higher variability and spikes than Mirai, indicating the system's adaptability to the dynamic nature of different attack patterns.

The statistical analysis shows no significant differences in energy consumption across scenarios (p-values ≥ 0.05), but small effect sizes provide practical insights. Gafgyt attacks slightly increase energy consumption compared to Normal (effect size = 0.027), while Mirai attacks have a marginally lower impact (-0.066). The slight difference between Gafgyt and Mirai (0.074) highlights the consistent energy efficiency of the EdgeShield-DRL method, maintain-

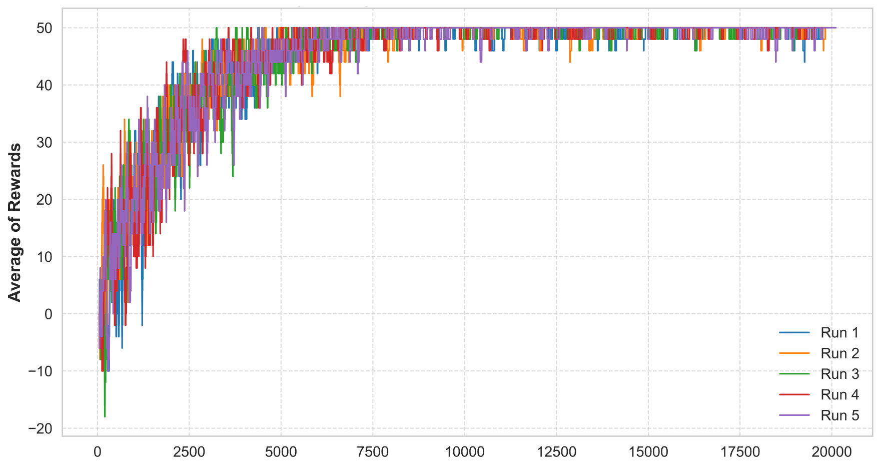

Figure 8.4 Learning performance of EdgeShield-DRL with $\epsilon = 1.0$, emphasizing exploration for evolving attack detection.

ing reliable performance under varying attack conditions.

Figure 8.8 illustrates carbon emissions over time for Gafgyt, Mirai, and Normal scenarios. Gafgyt attacks consistently contribute the least to carbon emissions, demonstrating a minimal environmental impact. On the other hand, Mirai attacks show a moderate contribution, higher than Gafgyt but lower than Normal operations. In contrast, Normal operations exhibit the highest carbon emissions across all time intervals, likely due to routine resource consumption. The trend remains stable over time, with no significant spikes during attack scenarios, reflecting the efficiency of the EdgeShield-DRL method in managing resources.

The statistical analysis shows no significant differences in carbon emissions across scenarios (p-values ≥ 0.05), but small effect sizes provide practical insights. Gafgyt attacks slightly increase carbon emissions compared to Normal (effect size = 0.032), while Mirai attacks have a marginally lower impact (effect size = -0.050), indicating a slightly reduced environmental footprint compared to Normal. The slight difference between Gafgyt and Mirai (effect size = 0.063) underscores the EdgeShield-DRL method's consistent energy efficiency and low environmental impact, even under varying attack conditions.

Figure 8.9 compares the CPU usage across Normal, Mirai, and Gafgyt scenarios. The Gafgyt scenario demonstrates the extensive range of CPU usage, with a slightly higher median than Normal operations, reflecting moderate resource demands during attack mitigation. Mirai attacks show a more consistent CPU usage range, with a median slightly lower than Normal, indicating efficient handling of computational requirements. Normal operations exhibit a narrower range, suggesting stable resource utilization under non-attack conditions.

The statistical analysis shows no significant differences in CPU usage across scenarios (p-values ≥ 0.05), but small negative effect sizes provide insights. Gafgyt and Mirai attacks

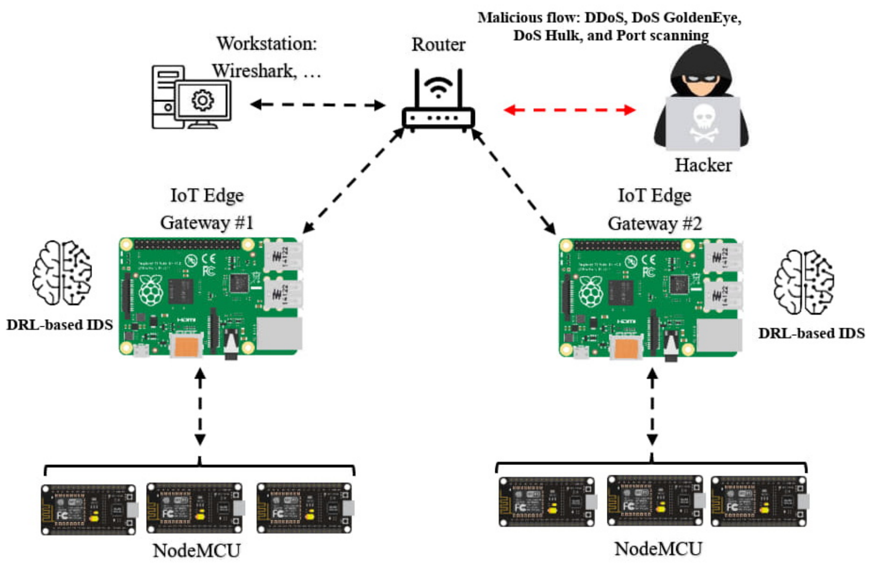
Malicious flow: DDoS, DoS GoldenEye, DoS Hulk, and Port scanning
Workstation: Wireshark, ...
Router
Hacker
IoT Edge Gateway #1
IoT Edge Gateway #2
DRL-based IDS
DRL-based IDS
NodeMCU
NodeMCU

Figure 8.5 Real-world testbed configuration illustrating the deployment of edge gateways.

show slightly lower CPU usage than Normal, with effect sizes of -0.074 and -0.064, respectively. The negligible difference between Gafgyt and Mirai (effect size = -0.014) further highlights the consistent performance of the EdgeShield-DRL method, ensuring efficient CPU utilization under varying attack conditions.

Figure 8.10 compares memory usage across Normal, Mirai, and Gafgyt scenarios. Mirai attacks exhibit the highest memory usage, a higher median than the other scenarios, indicating substantial resource demands during mitigation. Gafgyt attacks show slightly lower memory usage, with a narrower range and a consistent median, suggesting more efficient memory utilization. With a stable and narrow range, regular operations demonstrate the lowest memory usage, reflecting routine operations.

The statistical analysis shows significant differences in memory usage across scenarios (p-values < 0.05), with large effect sizes providing practical insights. Gafgyt and Mirai attacks significantly increase memory usage compared to Normal (effect size = 0.500 for both), indicating higher resource demands during attack mitigation. The difference between Gafgyt and Mirai (effect size = -0.458) reflects slightly lower memory usage during Gafgyt attacks. These findings emphasize the EdgeShield-DRL method's ability to handle resource-intensive scenarios while maintaining efficient memory management under varying attack conditions.

Figure 8.11 illustrates the IDS response time for Mirai and Gafgyt attacks over time. Both attack scenarios show consistent response times with occasional spikes. Gafgyt attacks exhibit slightly higher variability, with some peaks reaching above 0.34 seconds, while Mirai attacks remain comparatively stable, with fewer extreme values. Despite these spikes, the overall

```
Timestamp:  2024-12-14 15:00:00
System Status:  NORMAL
CPU Usage:  60%, Memory Usage:  1520 MB
Energy Consumption:  3 Joules, Carbon Emissions:  0.001 Kg
IDS Response Time:  0.24 seconds
The system is operating normally.
```

```
Timestamp:  2024-12-14 15:00:05
System Status:  ATTACK DETECTED (Gafgyt)
CPU Usage:  70%, Memory Usage:  1580 MB
Energy Consumption:  3.2 Joules, Carbon Emissions:  0.002 Kg
IDS Response Time:  0.24 seconds
ALERT: Gafgyt attack detected.  Initiating countermeasures.
Mitigation:  Blocking suspicious IP addresses.
```

```
Timestamp:  2024-12-14 15:00:10
System Status:  ATTACK DETECTED (Mirai)
CPU Usage:  66%, Memory Usage:  1600 MB
Energy Consumption:  3.5 Joules, Carbon Emissions:  0.0025 Kg
IDS Response Time:  0.26 seconds
ALERT: Mirai attack detected.  Initiating countermeasures.
Mitigation:  Blocking suspicious IP addresses.
```

Figure 8.6 System logs highlighting normal and attack states for Mirai and Gafgyt.

response times for both attacks stay within a manageable range, indicating the EdgeShield-DRL method's ability to maintain efficient real-time detection and mitigation. These results highlight the IDS's reliability and adaptability under different attack conditions.
The statistical analysis of IDS response time shows no significant differences across scenarios (p-values ≥ 0.05), but small effect sizes provide practical insights. Gafgyt attacks slightly increase the IDS response time compared to Normal (effect size $= 0.071$), while Mirai attacks have a negligible effect size (-0.007), indicating near-identical response times to Normal operations. The slight difference between Gafgyt and Mirai (effect size $= 0.085$) suggests a marginally longer response time for Gafgyt. These results underscore the EdgeShield-DRL method's ability to maintain rapid and efficient detection and mitigation capabilities across varying attack scenarios.

8.7 Discussion

The discussion highlights the effectiveness of the EdgeShield-DRL system in tackling critical challenges in IoT security, resource efficiency, and sustainability. The analysis of energy consumption and carbon emissions (Figures 8.7 and 8.8) shows that the system imposes minimal environmental impact during Gafgyt and Mirai attacks, with slight increases compared to

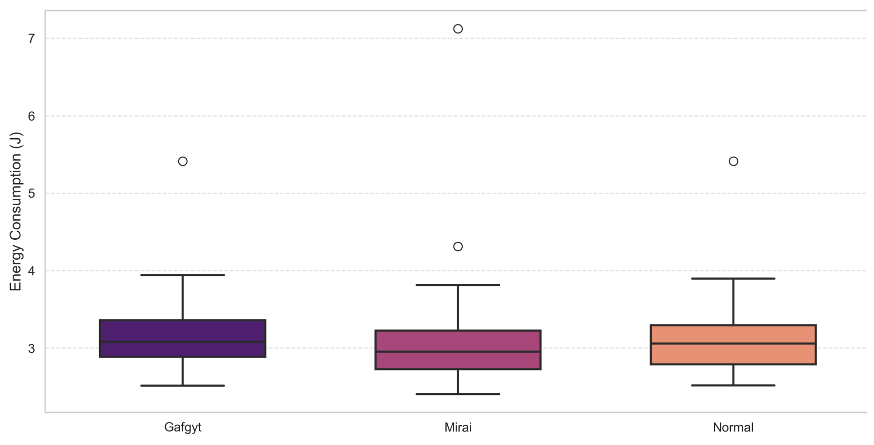

Figure 8.7 Energy consumption across normal, Mirai, and Gafgyt scenarios.

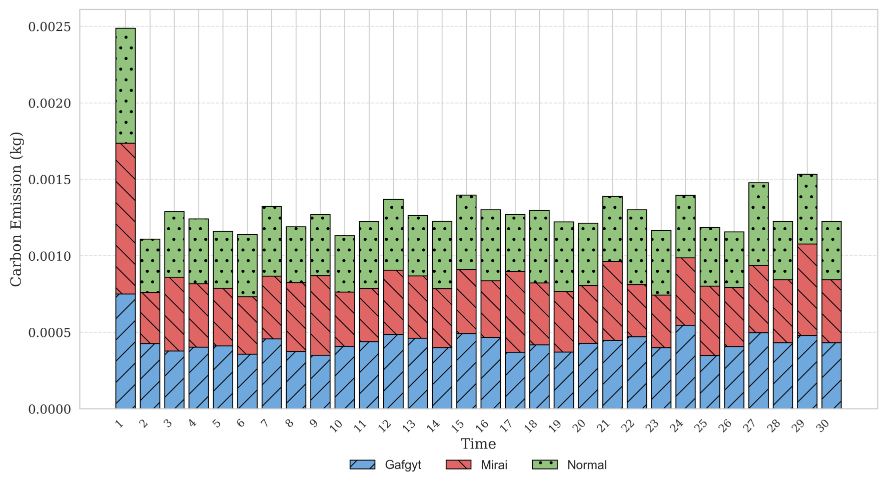

Figure 8.8 Carbon emissions under normal, Mirai, and Gafgyt scenarios over time.

Normal operations. This reflects its alignment with sustainable computing principles, crucial for edge gateways in resource-constrained environments. CPU and memory usage (Figures 8.9 and 8.10) demonstrate efficient resource management, ensuring stable performance without overloading the edge devices. Moreover, Gafgyt attacks exhibit more variability in CPU usage, while Mirai attacks demand higher memory resources, reflecting the differing computational complexities of these threats. IDS response time (Figure 8.11) remains low and stable across both attack scenarios, meeting the real-time detection and mitigation requirements in latency-sensitive IoT applications. While the system demonstrates robust performance, its evaluation was conducted on Raspberry Pi devices under specific conditions, which may differ from other environments. Variations in hardware configurations or workloads could affect energy consumption, memory usage, and response times. Therefore, considering the

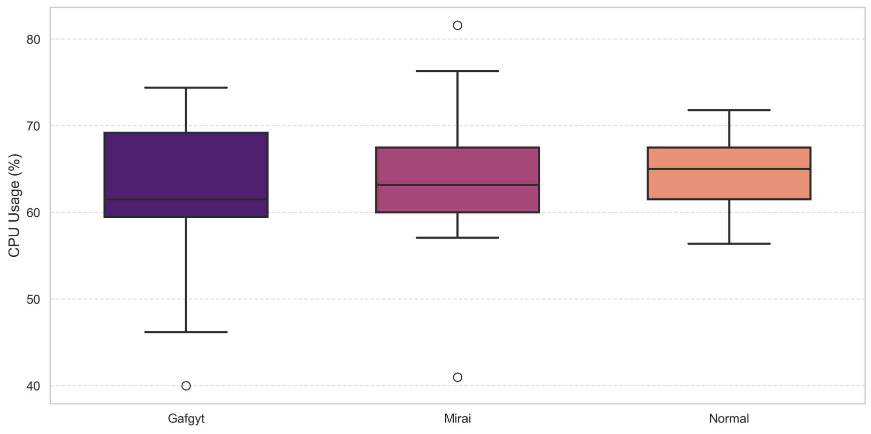

Figure 8.9 CPU usage comparison for normal, Mirai, and Gafgyt scenarios.

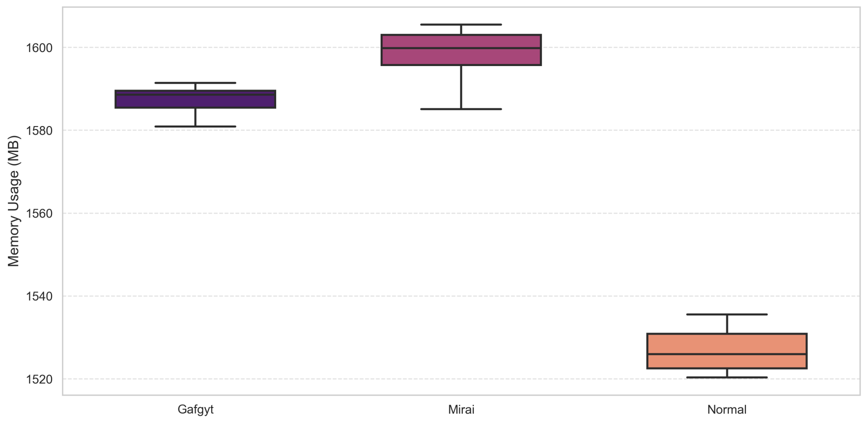

Figure 8.10 Memory usage comparison for normal, Mirai, and Gafgyt scenarios.

testing conditions' potential limitations, these results should be interpreted. These insights emphasize the system's potential for scalable, secure, and environmentally conscious IoT deployments.

8.8 Chapter Summary

This study presented the EdgeShield-DRL system, a DRL-based IDS for securing IoT edge gateways against threats like Gafgyt and Mirai. The system demonstrated robust security, low response times, and efficient resource utilization, ensuring real-time detection and mitigation in resource-constrained environments. Minimal energy and carbon emissions increases align with sustainable computing, making it suitable for critical applications such as smart cities and healthcare.

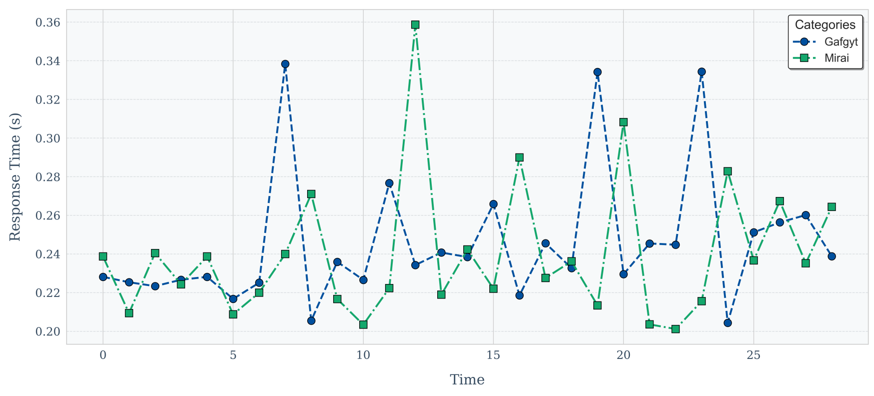

Figure 8.11 IDS response time for Mirai and Gafgyt attacks over time.

CHAPTER 9 CONCLUSION

This dissertation presents a DRL-based IDS framework to secure IoT systems against evolving cyber threats while optimizing energy efficiency and computational performance. Through a systematic analysis of traditional IDS techniques and IoT security patterns, this research identified key scalability, adaptability, and resource constraint limitations, necessitating the development of intelligent, self-adaptive cybersecurity mechanisms. To overcome these challenges, multiple DRL-based IDS models—DeepEdgeIDS, AutoDRL-IDS, EdgeShield-DRL, and SecuEdge-DRL—were introduced, each designed to enhance threat detection, anomaly mitigation, and resource efficiency. A dynamic security pattern selection framework was proposed, autonomously optimizing CPU load, memory consumption, and energy usage by selecting appropriate defense mechanisms in real-time.
The SecuEdge-DRL framework was further developed to integrate the MAPE-K model, enabling real-time adaptation of security policies. A plugin-based IDS test suite was designed to facilitate benchmarking and reproducibility of DRL-driven IDS models in various IoT and SDN environments. Extensive real-world testbed evaluations demonstrated that DRL-based IDS significantly improves attack detection accuracy, reduces false positive rates, and optimizes computational efficiency, outperforming conventional IDS solutions. These advancements contribute to sustainable cybersecurity, aligning with the United Nations SDGs, particularly SDG 9 (Industry, Innovation, and Infrastructure), SDG 11 (Sustainable Cities and Communities), and SDG 13 (Climate Action). This dissertation lays a foundation for scalable, adaptive, and sustainable IoT cybersecurity solutions by bridging the gap between security robustness and energy-efficient computing.

9.1 Summary of Works

The research presented in this dissertation systematically investigated the limitations of traditional ML-based IDS approaches in handling dynamic cyber threats in IoT networks. To address these limitations, the study introduced multiple DRL-based IDS frameworks that improve detection accuracy while balancing computational efficiency and evaluating widely used IoT security patterns, providing insights into optimizing IDS performance through dynamic security pattern selection.
Additionally, the research developed a novel plugin-based IDS test suite that allows for benchmarking ML- and DRL-based IDS models under various IoT network conditions. The real-world testbed experiments validated the efficiency of the proposed solutions, demonstrating

their ability to detect and mitigate cyber threats while maintaining low energy consumption. Integrating reinforcement learning with SDN-based architectures further improved IDS adaptability and performance in real-time scenarios.

9.2 Limitations

Despite the significant contributions of this dissertation, several limitations remain, highlighting areas for further improvement of the proposed DRL-based IDS frameworks. One key limitation is the susceptibility of DRL models to adversarial ML attacks. Although the models achieved high detection accuracy, they remain vulnerable to subtle traffic manipulations that can mislead decision-making and increase false positives, particularly against attacks, e.g., DoS Hulk. Integrating adversarial training, policy randomization, and adaptive confidence scoring could help address this vulnerability. Computational efficiency is another concern, especially in resource-constrained environments. While lightweight designs were adopted—particularly in DeepEdgeIDS and SecuEdge-DRL—real-world testing (e.g., on Raspberry Pi) revealed CPU load spikes under heavy traffic, e.g, DDoS simulations. Optimizations like model compression, hybrid IDS strategies, and dynamic anomaly thresholds are needed to balance detection performance and resource usage. Some models, such as AutoDRL-IDS and EdgeShield-DRL, rely on centralized training, posing scalability and privacy challenges. Aggregating data across edge nodes introduces communication overhead and potential exposure of sensitive information. Decentralized learning via FL is a promising alternative, offering privacy preservation and improved scalability. In addition, intermittent connectivity in IoT networks can disrupt real-time updates, which may be mitigated using edge caching mechanisms. The scope of validation also presents a limitation. While the models were evaluated in general and smart home scenarios, their performance in domains such as industrial IoT, healthcare, autonomous systems, and critical infrastructure remains to be explored. Broader validation is essential to ensure robustness and adaptability across diverse environments.

Furthermore, adaptive policy enforcement remains an open challenge. Although SecuEdge-DRL employs the MAPE-K loop for feedback-driven adaptation, more sophisticated mechanisms are needed to adjust detection thresholds and responses dynamically in fast-changing networks. Finally, while energy efficiency was a core design goal, long-term sustainability—especially in continuously operating systems—requires further attention. DRL models may incur cumulative energy costs, underscoring the need for ongoing research into green computing, energy profiling, and load-aware training techniques.

9.3 Future Research

This dissertation lays the foundation for adaptive, energy-efficient IDS in IoT edge networks. However, several open challenges and limitations point to meaningful avenues for future research. A critical area is enhancing the robustness of DRL-based IDS against adversarial ML. While DRL enables adaptability, it remains vulnerable to evasion tactics and adversarial inputs. Future efforts should explore adversarial training, policy randomization, and ensemble-based DRL approaches. Additionally, hybrid architectures that combine supervised and unsupervised learning can improve generalization and resilience to novel threats. Integrating FL into DRL frameworks is essential to supporting scalable and privacy-preserving IDS. FL allows collaborative model training across multiple edge devices without sharing raw data, reducing communication overhead and enhancing user privacy. Coupling FL with blockchain technologies further enhances decentralized trust management, secure communication, and tamper-proof logging, which are key for distributed IoT security. Moreover, optimizing resource consumption remains a priority. Techniques such as model compression, edge caching, knowledge distillation, and quantization can significantly reduce latency and energy usage. As energy efficiency becomes a growing concern amid global sustainability trends, IDS design must account for emerging energy constraints and green computing practices. Incorporating dynamic power management and context-aware activation of detection modules can further align IDS with energy-aware operation. In parallel, aligning IDS frameworks with SDN in IoT environments offers flexible traffic management, improved network visibility, and centralized policy enforcement. Future work should explore tighter integration of IDS with SDN controllers for real-time threat mitigation, automated policy updates, and programmable security workflows. Research must also address challenges around SDN scalability, controller security, and resilient communication between edge nodes and control planes. Furthermore, the evolution of 5G networks introduces both opportunities and challenges for IDS systems. Ultra-low latency, massive device connectivity, and network slicing require IDS to adapt to high-speed, highly dynamic environments. Investigating how IDS can operate efficiently over 5G-enabled infrastructures—possibly by leveraging edge-native architectures and network slicing-aware security policies- will be essential to supporting real-time threat detection at scale. Expanding IDS applicability to domain-specific environments such as smart cities, autonomous vehicles, industrial IoT, and healthcare brings further research opportunities. These contexts demand ultra-reliable, low-latency detection, domain-specific threat modeling, and strict privacy guarantees. Developing more context-aware and adaptive detection strategies is crucial for deployment in such sensitive or mission-critical environments. Advanced AI techniques may also enhance IDS capabilities. Moreover, GNNs, transformers, and LLMs can support improved threat contextualization, semantic reasoning, and model interpretability. Real-time threat intelligence, predictive analytics, adaptive confidence scoring, and ensemble anomaly detection methods could help boost detection accuracy while reducing false positives. Further research should also focus on orchestrating security patterns and adaptive policies at runtime. Extending the MAPE-K control loop with capabilities, e.g., real-time resource allocation, threat prioritization, and cross-layer coordination, could significantly improve network resilience under evolving attack landscapes. In addition, large-scale and long-term evaluation is essential to validate IDS performance under realistic conditions. Future studies should employ extended testbeds with heterogeneous IoT devices, introduce simulated concept drift, and continuously emulate evolving cyberattacks to assess the sustainability of proposed solutions. Moreover, ethical and regulatory dimensions must be considered. Fairness-aware detection mechanisms, explainable AI approaches, and policy frameworks for autonomous cybersecurity systems are crucial for responsible and transparent IDS deployment, especially in safety-critical or privacy-sensitive settings.

REFERENCES

[1] L. Reinfurt, U. Breitenbücher, M. Falkenthal, P. Fremantle, and F. Leymann, “Internet of Things security patterns,” in *Proc. PLoP*, 2017, p. 20.

[2] S. Jamshidi, A. Amirnia, A. Nikanjam, and F. Khomh, “Enhancing security and energy efficiency of cyber-physical systems using deep reinforcement learning,” *Procedia Computer Science*, vol. 238, pp. 1074–1079, 2024.

[3] R. Chataut, A. Phoummalayvane, and R. Akl, “Unleashing the power of iot: A comprehensive review of iot applications and future prospects in healthcare, agriculture, smart homes, smart cities, and industry 4.0,” *Sensors*, vol. 23, no. 16, p. 7194, 2023.

[4] K. O. M. Salih, T. A. Rashid, D. Radovanovic, and N. Bacanin, “A comprehensive survey on the internet of things with the industrial marketplace,” *Sensors*, vol. 22, no. 3, p. 730, 2022.

[5] T. M. Attia, “Challenges and opportunities in the future applications of iot technology,” 2019.

[6] K. Shafique, B. A. Khawaja, F. Sabir, S. Qazi, and M. Mustaqim, “Internet of things (iot) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5g-iot scenarios,” *Ieee Access*, vol. 8, pp. 23 022–23 040, 2020.

[7] A. N. Lone, S. Mustajab, and M. Alam, “A comprehensive study on cybersecurity challenges and opportunities in the iot world,” *Security and Privacy*, vol. 6, no. 6, p. e318, 2023.

[8] I. Stellios, P. Kotzanikolaou, M. Psarakis, C. Alcaraz, and J. Lopez, “A survey of iot-enabled cyberattacks: Assessing attack paths to critical infrastructures and services,” *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3453–3495, 2018.

[9] Q. Li, H. Huang, R. Li, J. Lv, Z. Yuan, L. Ma, Y. Han, and Y. Jiang, “A comprehensive survey on ddos defense systems: New trends and challenges,” *Computer Networks*, p. 109895, 2023.

[10] P. Kumari and A. K. Jain, “A comprehensive study of ddos attacks over iot network and their countermeasures,” *Computers & Security*, vol. 127, p. 103096, 2023.

[11] R. Vishwakarma and A. K. Jain, “A survey of ddos attacking techniques and defence mechanisms in the iot network,” *Telecommunication systems*, vol. 73, no. 1, pp. 3–25, 2020.

[12] D. Kshirsagar and S. Kumar, “An ontology approach for proactive detection of http flood dos attack,” *International Journal of System Assurance Engineering and Management*, vol. 14, no. Suppl 3, pp. 840–847, 2023.

[13] M. Frustaci, P. Pace, G. Aloi, and G. Fortino, “Evaluating critical security issues of the iot world: Present and future challenges,” *IEEE Internet of things journal*, vol. 5, no. 4, pp. 2483–2495, 2017.

[14] P. I. R. Grammatikis, P. G. Sarigiannidis, and I. D. Moscholios, “Securing the internet of things: Challenges, threats and solutions,” *Internet of Things*, vol. 5, pp. 41–70, 2019.

[15] O. I. Abiodun, E. O. Abiodun, M. Alawida, R. S. Alkhawaldeh, and H. Arshad, “A review on the security of the internet of things: Challenges and solutions,” *Wireless Personal Communications*, vol. 119, pp. 2603–2637, 2021.

[16] T. Bhattacharya, A. V. Peddi, S. Ponaganti, and S. T. Veeramalla, “A survey on various security protocols of edge computing,” *The Journal of Supercomputing*, vol. 81, no. 1, p. 310, 2025.

[17] D. W. Chadwick, W. Fan, G. Costantino, R. De Lemos, F. Di Cerbo, I. Herwono, M. Manea, P. Mori, A. Sajjad, and X.-S. Wang, “A cloud-edge based data security architecture for sharing and analysing cyber threat information,” *Future generation computer systems*, vol. 102, pp. 710–722, 2020.

[18] F. El Husseini, H. Noura, O. Salman, and A. Chehab, “Advanced machine learning approaches for zero-day attack detection: A review,” in *2024 8th Cyber Security in Networking Conference (CSNet)*. IEEE, 2024, pp. 297–304.

[19] R. Ahmad, I. Alsmadi, W. Alhamdani, and L. Tawalbeh, “Zero-day attack detection: a systematic literature review,” *Artificial Intelligence Review*, vol. 56, no. 10, pp. 10 733–10 811, 2023.

[20] S. Ali, S. U. Rehman, A. Imran, G. Adeem, Z. Iqbal, and K.-I. Kim, “Comparative evaluation of ai-based techniques for zero-day attacks detection,” *Electronics*, vol. 11, no. 23, p. 3934, 2022.

[21] N. Jeffrey, Q. Tan, and J. R. Villar, "A review of anomaly detection strategies to detect threats to cyber-physical systems," *Electronics*, vol. 12, no. 15, p. 3283, 2023.

[22] S. Ennaji, F. De Gaspari, D. Hitaj, A. Kbidi, and L. V. Mancini, "Adversarial challenges in network intrusion detection systems: Research insights and future prospects," *arXiv preprint arXiv:2409.18736*, 2024.

[23] F. Louati, F. B. Ktata, and I. Amous, "Big-ids: a decentralized multi agent reinforcement learning approach for distributed intrusion detection in big data networks," *Cluster Computing*, pp. 1–19, 2024.

[24] S. A. Bakhsh, M. A. Khan, F. Ahmed, M. S. Alshehri, H. Ali, and J. Ahmad, "Enhancing iot network security through deep learning-powered intrusion detection system," *Internet of Things*, vol. 24, p. 100936, 2023.

[25] U. Inayat, M. F. Zia, S. Mahmood, H. M. Khalid, and M. Benbouzid, "Learning-based methods for cyber attacks detection in iot systems: A survey on methods, analysis, and future prospects," *Electronics*, vol. 11, no. 9, p. 1502, 2022.

[26] Y. K. Saheed, A. I. Abiodun, S. Misra, M. K. Holone, and R. Colomo-Palacios, "A machine learning-based intrusion detection for detecting internet of things network attacks," *Alexandria Engineering Journal*, vol. 61, no. 12, pp. 9395–9409, 2022.

[27] M. Bansal, A. Goyal, and A. Choudhary, "A comparative analysis of k-nearest neighbor, genetic, support vector machine, decision tree, and long short term memory algorithms in machine learning," *Decision Analytics Journal*, vol. 3, p. 100071, 2022.

[28] Z. Azam, M. M. Islam, and M. N. Huda, "Comparative analysis of intrusion detection systems and machine learning based model analysis through decision tree," *IEEE Access*, 2023.

[29] G. Kumar and H. Alqahtani, "Machine learning techniques for intrusion detection systems in sdn-recent advances, challenges and future directions." *CMES-Computer Modeling in Engineering & Sciences*, vol. 134, no. 1, 2023.

[30] Y. Wu, B. Zou, and Y. Cao, "Current status and challenges and future trends of deep learning-based intrusion detection models," *Journal of Imaging*, vol. 10, no. 10, p. 254, 2024.

[31] M. A. Shihab, H. A. Marhoon, S. R. Ahmed, A. D. Radhi, and R. Sekhar, "Towards resilient machine learning models: Addressing adversarial attacks in wireless sensor network," *Journal of Robotics and Control (JRC)*, vol. 5, no. 5, pp. 1599–1617, 2024.

[32] K. Aryal, M. Gupta, M. Abdelsalam, P. Kunwar, and B. Thuraisingham, "A survey on adversarial attacks for malware analysis," *IEEE Access*, 2024.

[33] A. Guesmi, M. A. Hanif, B. Ouni, and M. Shafique, "Physical adversarial attacks for camera-based smart systems: Current trends, categorization, applications, research challenges, and future outlook," *IEEE Access*, 2023.

[34] B. R. Maddireddy and B. R. Maddireddy, "The role of reinforcement learning in dynamic cyber defense strategies," *International Journal of Advanced Engineering Technologies and Innovations*, vol. 1, no. 2, pp. 267–292, 2024.

[35] M. Sewak, S. K. Sahay, and H. Rathore, "Deep reinforcement learning in the advanced cybersecurity threat detection and protection," *Information Systems Frontiers*, vol. 25, no. 2, pp. 589–611, 2023.

[36] F. Arif, N. A. Khan, J. Iqbal, F. K. Karim, N. Innab, S. M. Mostafa *et al.*, "Dqqs: Deep reinforcement learning based technique for enhancing security and performance in sdn-iot environments," *IEEE Access*, 2024.

[37] H. Kheddar, D. W. Dawoud, A. I. Awad, Y. Himeur, and M. K. Khan, "Reinforcement-learning-based intrusion detection in communication networks: A review," *IEEE Communications Surveys & Tutorials*, 2024.

[38] P. Sharma, S. Jain, S. Gupta, and V. Chamola, "Role of machine learning and deep learning in securing 5g-driven industrial iot applications," *Ad Hoc Networks*, vol. 123, p. 102685, 2021.

[39] V. G. da Silva Ruffo, D. M. B. Lent, M. Komarchesqui, V. F. Schiavon, M. V. O. de Assis, L. F. Carvalho, and M. L. Proença Jr, "Anomaly and intrusion detection using deep learning for software-defined networks: A survey," *Expert Systems with Applications*, p. 124982, 2024.

[40] S. U. Qureshi, J. He, S. Tunio, N. Zhu, A. Nazir, A. Wajahat, F. Ullah, and A. Wadud, "Systematic review of deep learning solutions for malware detection and forensic analysis in iot," *Journal of King Saud University-Computer and Information Sciences*, p. 102164, 2024.

[41] P. Malhotra, Y. Singh, P. Anand, D. K. Bangotra, P. K. Singh, and W.-C. Hong, "Internet of things: Evolution, concerns and security challenges," *Sensors*, vol. 21, no. 5, p. 1809, 2021.

[42] M. A. Khan and K. Salah, “Iot security: Review, blockchain solutions, and open challenges,” *Future generation computer systems*, vol. 82, pp. 395–411, 2018.

[43] H. HaddadPajouh, A. Dehghantanha, R. M. Parizi, M. Aledhari, and H. Karimipour, “A survey on internet of things security: Requirements, challenges, and solutions,” *Internet of Things*, vol. 14, p. 100129, 2021.

[44] U. Nations, “United nations goals: Sustainable development,” 2023, accessed: September 3, 2024. [Online]. Available: https://sdgs.un.org/goals

[45] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, “Edge computing: Vision and challenges,” *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[46] M. Satyanarayanan, “The emergence of edge computing,” *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[47] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, “Fog computing and its role in the internet of things,” in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, 2012, pp. 13–16.

[48] W. AlAqqad, M. Nijim, U. Onyeakazi, and H. Albataineh, “Cyber edge: Mitigating cyber-attacks in edge computing using intrusion detection system,” in *International Conference on Advances in Computing Research.* Springer, 2024, pp. 292–305.

[49] A. Nag, M. M. Hassan, A. Das, A. Sinha, N. Chand, A. Kar, V. Sharma, and A. Alkhayyat, “Exploring the applications and security threats of internet of thing in the cloud computing paradigm: A comprehensive study on the cloud of things,” *Transactions on Emerging Telecommunications Technologies*, vol. 35, no. 4, p. e4897, 2024.

[50] F. A. Alaba, “Iot architecture layers,” in *Internet of Things: A Case Study in Africa.* Springer, 2024, pp. 65–85.

[51] M. Adam, M. Hammoudeh, R. Alrawashdeh, and B. Alsulaimy, “A survey on security, privacy, trust, and architectural challenges in iot systems,” *IEEE Access*, 2024.

[52] O. Arshi, A. Rai, G. Gupta, J. K. Pandey, and S. Mondal, “Iot in energy: a comprehensive review of technologies, applications, and future directions,” *Peer-to-Peer Networking and Applications*, pp. 1–40, 2024.

[53] E. N. Amachaghi, M. Shojafar, C. H. Foh, and K. Moessner, “A survey for intrusion detection systems in open ran,” *IEEE Access*, 2024.

[54] S. J. Soheli, N. Jahan, M. B. Hossain, A. Adhikary, A. R. Khan, and M. Wahiduzzaman, “Smart greenhouse monitoring system using internet of things and artificial intelligence,” *Wireless Personal Communications*, vol. 124, no. 4, pp. 3603–3634, 2022.

[55] M. Ahmid and O. Kazar, “A comprehensive review of the internet of things security,” *Journal of Applied Security Research*, vol. 18, no. 3, pp. 289–305, 2023.

[56] S. N. G. Aryavalli and H. Kumar, “Top 12 layer-wise security challenges and a secure architectural solution for internet of things,” *Computers and Electrical Engineering*, vol. 105, p. 108487, 2023.

[57] V. V. Vegesna, “Methodology for mitigating the security issues and challenges in the internet of things (iot) framework for enhanced security,” *Asian Journal of Basic Science & Research*, vol. 5, no. 1, pp. 85–102, 2023.

[58] B. Kaur, S. Dadkhah, F. Shoeleh, E. C. P. Neto, P. Xiong, S. Iqbal, P. Lamontagne, S. Ray, and A. A. Ghorbani, “Internet of things (iot) security dataset evolution: Challenges and future directions,” *Internet of Things*, vol. 22, p. 100780, 2023.

[59] M. A. Jamshed, K. Ali, Q. H. Abbasi, M. A. Imran, and M. Ur-Rehman, “Challenges, applications, and future of wireless sensors in internet of things: A review,” *IEEE Sensors Journal*, vol. 22, no. 6, pp. 5482–5494, 2022.

[60] M. Agoramoorthy, A. Ali, D. Sujatha, M. R. TF, and G. Ramesh, “An analysis of signature-based components in hybrid intrusion detection systems,” in *2023 Intelligent Computing and Control for Engineering and Business Systems (ICCEBS)*. IEEE, 2023, pp. 1–5.

[61] D. Ramakrishna and M. A. Shaik, “A comprehensive analysis of cryptographic algorithms: Evaluating security, efficiency, and future challenges,” *IEEE Access*, 2024.

[62] J. M. Kizza, “System intrusion detection and prevention,” in *Guide to computer network security*. Springer, 2024, pp. 295–323.

[63] S. Chell, S. Chakare, P. Sohan, and S. Sandosh, “Real-time threat detection and mitigation in web api development,” in *2024 International Conference on Electrical Electronics and Computing Technologies (ICEECT)*, vol. 1. IEEE, 2024, pp. 1–9.

[64] P. Bajaj, S. Mishra, and A. Paul, “Comparative analysis of stack-ensemble-based intrusion detection system for single-layer and cross-layer dos attack detection in iot,” *SN Computer Science*, vol. 4, no. 5, p. 562, 2023.

[65] K. Lian, L. Zhang, G. Yang, S. Mao, X. Wang, Y. Zhang, and M. Yang, “Component security ten years later: An empirical study of cross-layer threats in real-world mobile applications,” *Proceedings of the ACM on Software Engineering*, vol. 1, no. FSE, pp. 70–91, 2024.

[66] H. Azzaoui, A. Z. E. Boukhamla, P. Perazzo, M. Alazab, and V. Ravi, “A lightweight cooperative intrusion detection system for rpl-based iot,” *Wireless Personal Communications*, vol. 134, no. 4, pp. 2235–2258, 2024.

[67] A. Fausto, G. Gaggero, F. Patrone, and M. Marchese, “Reduction of the delays within an intrusion detection system (ids) based on software defined networking (sdn),” *IEEE Access*, vol. 10, pp. 109 850–109 862, 2022.

[68] A. H. Shamsan and A. R. Faridi, “Sdn-assisted iot architecture: a review,” in *2018 4th International Conference on Computing Communication and Automation (ICCCA).* IEEE, 2018, pp. 1–7.

[69] Á. L. Valdivieso Caraguay, A. Benito Peral, L. I. Barona Lopez, and L. J. Garcia Villalba, “Sdn: Evolution and opportunities in the development iot applications,” *International Journal of Distributed Sensor Networks*, vol. 10, no. 5, p. 735142, 2014.

[70] O. Flauzac, C. González, A. Hachani, and F. Nolot, “Sdn based architecture for iot and improvement of the security,” in *2015 IEEE 29th international conference on advanced information networking and applications workshops.* IEEE, 2015, pp. 688–693.

[71] M. B. Yassein, S. Aljawarneh, M. Al-Rousan, W. Mardini, and W. Al-Rashdan, “Combined software-defined network (sdn) and internet of things (iot),” in *2017 international conference on electrical and computing technologies and applications (ICECTA).* IEEE, 2017, pp. 1–6.

[72] N. Bizanis and F. A. Kuipers, “Sdn and virtualization solutions for the internet of things: A survey,” *IEEE Access*, vol. 4, pp. 5591–5606, 2016.

[73] A. Al Hayajneh, M. Z. A. Bhuiyan, and I. McAndrew, “Improving internet of things (iot) security with software-defined networking (sdn),” *Computers*, vol. 9, no. 1, p. 8, 2020.

[74] M. A. Bouke, A. Abdullah, S. H. ALshatebi, and M. T. Abdullah, “E2ids: An enhanced intelligent intrusion detection system based on decision tree algorithm,” *Journal of Applied Artificial Intelligence*, vol. 3, no. 1, pp. 1–16, 2022.

[75] L. A. C. Ahakonye, C. I. Nwakanma, J.-M. Lee, and D.-S. Kim, "Scada intrusion detection scheme exploiting the fusion of modified decision tree and chi-square feature selection," *Internet of Things*, vol. 21, p. 100676, 2023.

[76] M. Hammad, N. Hewahi, and W. Elmedany, "Mmm-rf: A novel high accuracy multinomial mixture model for network intrusion detection systems," *Computers & Security*, vol. 120, p. 102777, 2022.

[77] K. Albulayhi, Q. Abu Al-Haija, S. A. Alsuhibany, A. A. Jillepalli, M. Ashrafuzzaman, and F. T. Sheldon, "Iot intrusion detection using machine learning with a novel high performing feature selection method," *Applied Sciences*, vol. 12, no. 10, p. 5015, 2022.

[78] H. Yang, S. Liang, J. Ni, H. Li, and X. S. Shen, "Secure and efficient k nn classification for industrial internet of things," *IEEE Internet of Things Journal*, vol. 7, no. 11, pp. 10 945–10 954, 2020.

[79] A. D. Afifaturahman and M. Firmansyah, "Perbandingan algoritma k-nearest neighbour (knn) dan naive bayes pada intrusion detection system (ids)," *Innovation in Research of Informatics (INNOVATICS)*, vol. 3, no. 1, 2021.

[80] F. Z. Belgrana, N. Benamrane, M. A. Hamaida, A. M. Chaabani, and A. Taleb-Ahmed, "Network intrusion detection system using neural network and condensed nearest neighbors with selection of nsl-kdd influencing features," in *2020 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)*. IEEE, 2021, pp. 23–29.

[81] Y. Yan, L. Qi, J. Wang, Y. Lin, and L. Chen, "A network intrusion detection method based on stacked autoencoder and lstm," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[82] M. D. Hossain, H. Inoue, H. Ochiai, D. Fall, and Y. Kadobayashi, "Lstm-based intrusion detection system for in-vehicle can bus communications," *IEEE Access*, vol. 8, pp. 185 489–185 502, 2020.

[83] A. El-Ghamry, A. Darwish, and A. E. Hassanien, "An optimized cnn-based intrusion detection system for reducing risks in smart farming," *Internet of Things*, vol. 22, p. 100709, 2023.

[84] S. Jamshidi, A. Nikanjam, M. A. Hamdaqa, and F. Khomh, "Attack detection by using deep learning for cyber-physical system," in *Artificial Intelligence for Cyber-Physical Systems Hardening*. Springer, 2022, pp. 155–179.

[85] P. Sun, P. Liu, Q. Li, C. Liu, X. Lu, R. Hao, and J. Chen, "Dl-ids: Extracting features using cnn-lstm hybrid network for intrusion detection system," *Security and communication networks*, vol. 2020, pp. 1–11, 2020.

[86] A. Halbouni, T. S. Gunawan, M. H. Habaebi, M. Halbouni, M. Kartiwi, and R. Ahmad, "Cnn-lstm: hybrid deep neural network for network intrusion detection system," *IEEE Access*, vol. 10, pp. 99 837–99 849, 2022.

[87] O. Elnakib, E. Shaaban, M. Mahmoud, and K. Emara, "Eidm: deep learning model for iot intrusion detection systems," *The Journal of Supercomputing*, pp. 1–21, 2023.

[88] I. Sorokin, A. Seleznev, M. Pavlov, A. Fedorov, and A. Ignateva, "Deep attention recurrent q-network," *arXiv preprint arXiv:1512.01693*, 2015.

[89] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[90] R. Bellman, "Dynamic programming," *science*, vol. 153, no. 3731, pp. 34–37, 1966.

[91] J. Finnegan and S. Brown, "An analysis of the energy consumption of lpwa-based iot devices," in *2018 International Symposium on Networks, Computers and Communications (ISNCC)*. IEEE, 2018, pp. 1–6.

[92] A. S. Abdul-Qawy, P. Pramod, E. Magesh, and T. Srinivasulu, "The internet of things (iot): An overview," *International Journal of Engineering Research and Applications*, vol. 5, no. 12, pp. 71–82, 2015.

[93] E. B. Fernandez, H. Washizaki, N. Yoshioka, and T. Okubo, "The design of secure IoT applications using patterns: State of the art and directions for research," *Internet of Things*, vol. 15, p. 100408, 2021.

[94] C. Orellana, E. B. Fernandez, and H. Astudillo, "A pattern for a secure sensor node," in *Proceedings of the 27th Conference on Pattern Languages of Programs (USA)(PLOP'20). Association for Computing Machinery, USA*, 2020.

[95] U. B. Ahmad, M. A. Akram, and A. N. Mian, "Low-latency intrusion detection using a deep neural network," *IT Professional*, vol. 24, no. 3, pp. 67–72, 2022.

[96] Y. Liu, K.-F. Tsang, C. K. Wu, Y. Wei, H. Wang, and H. Zhu, "Ieee p2668-compliant multi-layer iot-ddos defense system using deep reinforcement learning," *IEEE Transactions on Consumer Electronics*, vol. 69, no. 1, pp. 49–64, 2022.

[97] X. Liu, W. Yu, F. Liang, D. Griffith, and N. Golmie, “On deep reinforcement learning security for industrial internet of things,” *Computer Communications*, vol. 168, pp. 20–32, 2021.

[98] T. Ramana, M. Thirunavukkarasan, A. S. Mohammed, G. G. Devarajan, and S. M. Nagarajan, “Ambient intelligence approach: Internet of things based decision performance analysis for intrusion detection,” *Computer Communications*, vol. 195, pp. 315–322, 2022.

[99] S. Vadigi, K. Sethi, D. Mohanty, S. P. Das, and P. Bera, “Federated reinforcement learning based intrusion detection system using dynamic attention mechanism,” *Journal of Information Security and Applications*, vol. 78, p. 103608, 2023.

[100] X. Ma, Y. Li, and Y. Gao, “Decision model of intrusion response based on markov game in fog computing environment,” *Wireless Networks*, vol. 29, no. 8, pp. 3383–3392, 2023.

[101] R. Zhang, H. Xia, C. Liu, R.-b. Jiang, and X.-g. Cheng, “Anti-attack scheme for edge devices based on deep reinforcement learning,” *Wireless Communications and Mobile Computing*, vol. 2021, pp. 1–9, 2021.

[102] M. Al-Fawa’reh, J. Abu-Khalaf, P. Szewczyk, and J. J. Kang, “Malbot-drl: Malware botnet detection using deep reinforcement learning in iot networks,” *IEEE Internet of Things Journal*, 2023.

[103] M. Lopez-Martin, B. Carro, and A. Sanchez-Esguevillas, “Application of deep reinforcement learning to intrusion detection for supervised problems,” *Expert Systems with Applications*, vol. 141, p. 112963, 2020.

[104] M. M. Almasri and A. M. Alajlan, “A novel-cascaded anfis-based deep reinforcement learning for the detection of attack in cloud iot-based smart city applications,” *Concurrency and Computation: Practice and Experience*, vol. 35, no. 22, p. e7738, 2023.

[105] J. Wang, Y. Liu, W. Zhang, X. Yan, N. Zhou, and Z. Jiang, “Relfa: Resist link flooding attacks via renyi entropy and deep reinforcement learning in sdn-iot,” *China Communications*, vol. 19, no. 7, pp. 157–171, 2022.

[106] N. Kandhoul and S. K. Dhurandher, “Deep q learning based secure routing approach for oppiot networks,” *Internet of Things*, vol. 20, p. 100597, 2022.

[107] J. Wang and J. Liu, “Deep learning for securing software-defined industrial internet of things: attacks and countermeasures,” *IEEE Internet of Things Journal*, vol. 9, no. 13, pp. 11 179–11 189, 2021.

[108] K. Ren, Y. Zeng, Y. Zhong, B. Sheng, and Y. Zhang, "Mafsids: a reinforcement learning-based intrusion detection model for multi-agent feature selection networks," *Journal of Big Data*, vol. 10, no. 1, p. 137, 2023.

[109] B. Yang, M. H. Arshad, and Q. Zhao, "Packet-level and flow-level network intrusion detection based on reinforcement learning and adversarial training," *Algorithms*, vol. 15, no. 12, p. 453, 2022.

[110] Y. Feng, W. Zhang, S. Yin, H. Tang, Y. Xiang, and Y. Zhang, "A collaborative stealthy ddos detection method based on reinforcement learning at the edge of the internet of things," *IEEE Internet of Things Journal*, 2023.

[111] H. Karthikeyan and G. Usha, "Real-time ddos flooding attack detection in intelligent transportation systems," *Computers and Electrical Engineering*, vol. 101, p. 107995, 2022.

[112] F. Mesadieu, D. Torre, and A. Chennameneni, "Leveraging deep reinforcement learning technique for intrusion detection in scada infrastructure," *IEEE Access*, 2024.

[113] H. H. R. Sherazi, R. Iqbal, F. Ahmad, Z. A. Khan, and M. H. Chaudary, "Ddos attack detection: A key enabler for sustainable communication in internet of vehicles," *Sustainable Computing: Informatics and Systems*, vol. 23, pp. 13–20, 2019.

[114] R. Heartfield, G. Loukas, A. Bezemskij, and E. Panaousis, "Self-configurable cyber-physical intrusion detection for smart homes using reinforcement learning," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1720–1735, 2020.

[115] P. Radoglou-Grammatikis, K. Rompolos, P. Sarigiannidis, V. Argyriou, T. Lagkas, A. Sarigiannidis, S. Goudos, and S. Wan, "Modeling, detecting, and mitigating threats against industrial healthcare systems: a combined software defined networking and reinforcement learning approach," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 2041–2052, 2021.

[116] C. Hu, J. Yan, and X. Liu, "Reinforcement learning-based adaptive feature boosting for smart grid intrusion detection," *IEEE Transactions on Smart Grid*, 2022.

[117] L. Nie, W. Sun, S. Wang, Z. Ning, J. J. Rodrigues, Y. Wu, and S. Li, "Intrusion detection in green internet of things: a deep deterministic policy gradient-based algorithm," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 2, pp. 778–788, 2021.

[118] N. M. Al-Maslamani, B. S. Ciftler, M. Abdallah, and M. M. Mahmoud, "Toward secure federated learning for iot using drl-enabled reputation mechanism," *IEEE Internet of Things Journal*, vol. 9, no. 21, pp. 21 971–21 983, 2022.

[119] J. Parras, A. Almodóvar, P. A. Apellániz, and S. Zazo, "Inverse reinforcement learning: a new framework to mitigate an intelligent backoff attack," *IEEE Internet of Things Journal*, vol. 9, no. 24, pp. 24 790–24 799, 2022.

[120] S. Tharewal, M. W. Ashfaque, S. S. Banu, P. Uma, S. M. Hassen, and M. Shabaz, "Intrusion detection system for industrial internet of things based on deep reinforcement learning," *Wireless Communications and Mobile Computing*, vol. 2022, pp. 1–8, 2022.

[121] N. S. Alotaibi, H. I. Ahmed, and S. O. M. Kamel, "Dynamic adaptation attack detection model for a distributed multi-access edge computing smart city," *Sensors*, vol. 23, no. 16, p. 7135, 2023.

[122] S. Dong, Y. Xia, and T. Peng, "Network abnormal traffic detection model based on semi-supervised deep reinforcement learning," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4197–4212, 2021.

[123] M. Alauthman, N. Aslam, M. Al-Kasassbeh, S. Khan, A. Al-Qerem, and K.-K. R. Choo, "An efficient reinforcement learning-based botnet detection approach," *Journal of Network and Computer Applications*, vol. 150, p. 102479, 2020.

[124] V. Praveena, A. Vijayaraj, P. Chinnasamy, I. Ali, R. Alroobaea, S. Y. Alyahyan, and M. A. Raza, "Optimal deep reinforcement learning for intrusion detection in uavs," *Computers, Materials & Continua*, vol. 70, no. 2, pp. 2639–2653, 2022.

[125] V. Juneja, S. K. Dinkar, and D. V. Gupta, "An anomalous co-operative trust & pg-drl based vampire attack detection & routing," *Concurrency and Computation: Practice and Experience*, vol. 34, no. 3, p. e6557, 2022.

[126] A. Pashaei, M. E. Akbari, M. Z. Lighvan, and A. Charmin, "Early intrusion detection system using honeypot for industrial control networks," *Results in Engineering*, vol. 16, p. 100576, 2022.

[127] A. K. Shukla, "Detection of anomaly intrusion utilizing self-adaptive grasshopper optimization algorithm," *Neural Computing and Applications*, vol. 33, no. 13, pp. 7541–7561, 2021.

[128] D. K. Dake, J. D. Gadze, G. S. Klogo, and H. Nunoo-Mensah, “Multi-agent reinforcement learning framework in sdn-iot for transient load detection and prevention,” *Technologies*, vol. 9, no. 3, p. 44, 2021.

[129] X. Feng, J. Han, R. Zhang, S. Xu, and H. Xia, “Security defense strategy algorithm for internet of things based on deep reinforcement learning,” *High-Confidence Computing*, vol. 4, no. 1, p. 100167, 2024.

[130] S. Janakiraman and M. Deva Priya, “A deep reinforcement learning-based ddos attack mitigation scheme for securing big data in fog-assisted cloud environment,” *Wireless Personal Communications*, vol. 130, no. 4, pp. 2869–2886, 2023.

[131] H. Nandanwar and R. Katarya, “Deep learning enabled intrusion detection system for industrial iot environment,” *Expert Systems with Applications*, vol. 249, p. 123808, 2024.

[132] “Intrusion detection evaluation dataset (iscxids2012),” https://www.unb.ca/cic/datasets/ids.html, Canadian Institute for Cybersecurity, University of New Brunswick.

[133] “Unsw-nb15 dataset,” https://research.unsw.edu.au/projects/unsw-nb15-dataset, University of New South Wales.

[134] “Intrusion detection evaluation dataset (cic-ids2017),” https://www.unb.ca/cic/datasets/ids-2017.html, Canadian Institute for Cybersecurity, University of New Brunswick.

[135] “Industrial internet of things (iiot) research 2018,” https://www.cse.wustl.edu/~jain/iiot/index.html, Department of Computer Science, Washington University in St. Louis.

[136] “Advanced industrial internet of things (iiot) research 2021,” https://www.cse.wustl.edu/~jain/iiot2/index.html, Department of Computer Science, Washington University in St. Louis.

[137] “N-baiot dataset,” https://www.kaggle.com/datasets/mkashifn/nbaiot-dataset/code, Kaggle, accessed: 2024-05-07.

[138] “Hogzilla ids dataset,” https://ids-hogzilla.org/dataset/, Hogzilla IDS.

[139] “Iotid20 dataset,” https://www.kaggle.com/datasets/rohulaminlabid/iotid20-dataset, Kaggle.

[140] P. Radoglou-Grammatikis, K. Rompolos, T. Lagkas, V. Argyriou, and P. Sarigiannidis, "Iec 60870-5-104 intrusion detection dataset," 2022. [Online]. Available: https://dx.doi.org/10.21227/fj7s-f281

[141] "Awid intrusion detection dataset," https://github.com/Bee-Mar/AWID-Intrusion-Detection, GitHub.

[142] "Bot-iot dataset," https://research.unsw.edu.au/projects/bot-iot-dataset, University of New South Wales.

[143] "Ddos 2019 dataset," https://www.unb.ca/cic/datasets/ddos-2019.html, Canadian Institute for Cybersecurity, University of New Brunswick.

[144] "Medbiot dataset," https://cs.taltech.ee/research/data/medbiot/, Tallinn University of Technology.

[145] U. Singh and M. Rizwan, "Scada system dataset exploration and machine learning based forecast for wind turbines," *Results in Engineering*, vol. 16, p. 100640, 2022.

[146] J. F. Cevallos Moreno, A. Rizzardi, S. Sicari, and A. Coen-Porisini, "Deep reinforcement learning for intrusion detection in internet of things: Best practices, lessons learnt, and open challenges," *Lessons Learnt, and Open Challenges.*

[147] A. K. Zamani and A. Chapnevis, "Botnet intrusion detection system in internet of things with developed deep learning," *arXiv preprint arXiv:2207.04503*, 2022.

[148] S. Anbazhagan and R. Mugelan, "Next-gen resource optimization in nb-iot networks: Harnessing soft actor–critic reinforcement learning," *Computer Networks*, vol. 252, p. 110670, 2024.

[149] S. Yang, L. Zhuang, J. Zhang, J. Lan, and B. Li, "A multi-policy deep reinforcement learning approach for multi-objective joint routing and scheduling in deterministic networks," *IEEE Internet of Things Journal*, 2024.

[150] "Kdd cup 1999 data," https://www.kaggle.com/datasets/galaxyh/kdd-cup-1999-data, Kaggle, 1999.

[151] "Edge iiotset pre-processing," https://www.kaggle.com/code/mohamedamineferrag/edge-iiotset-pre-processing, Kaggle.

[152] U. of New Brunswick, "Iot dataset 2023," 2023, accessed: 2025-01-09. [Online]. Available: https://www.unb.ca/cic/datasets/iotdataset-2023.html

[153] B. Saeed, S. Arshad, S. M. U. Saeed, and M. A. Azam, "Enhancing iot security: Federated learning with gans for effective attack detection," in *2023 20th International Bhurban Conference on Applied Sciences and Technology (IBCAST).* IEEE, 2023, pp. 570–575.

[154] ns 3 Project, "ns-3 network simulator." [Online]. Available: https://www.nsnam.org/

[155] C. O. Project, "An introduction to cooja simulator." [Online]. Available: https://github.com/contiki-os/contiki/wiki/An-Introduction-to-COOJA

[156] O. Project, "Omnet++ network simulator." [Online]. Available: https://omnetpp.org/

[157] M. Adam and U. Baroud, "Federated learning for iot: Applications, trends, taxonomy, challenges, current solutions, and future directions," *IEEE Open Journal of the Communications Society*, 2024.

[158] F. Chen, L. Wang, J. Hong, J. Jiang, and L. Zhou, "Unmasking bias in artificial intelligence: a systematic review of bias detection and mitigation strategies in electronic health record-based models," *Journal of the American Medical Informatics Association*, vol. 31, no. 5, pp. 1172–1183, 2024.

[159] M. Tabassum, S. Mahmood, A. Bukhari, B. Alshemaimri, A. Daud, and F. Khalique, "Anomaly-based threat detection in smart health using machine learning," *BMC Medical Informatics and Decision Making*, vol. 24, no. 1, p. 347, 2024.

[160] D. G. Chowdhry, R. Verma, and M. Mathur, *The Evolution of Business in the Cyber Age: Digital Transformation, Threats, and Security.* CRC Press, 2020.

[161] S. Hadzovic, S. Mrdovic, and M. Radonjic, "A path towards an internet of things and artificial intelligence regulatory framework," *IEEE Communications Magazine*, 2023.

[162] K. L. M. Ang, J. K. P. Seng, and E. Ngharamike, "Towards crowdsourcing internet of things (crowd-iot): Architectures, security, and applications," *Future Internet*, vol. 14, no. 2, p. 49, 2022.

[163] P. Mall, R. Amin, A. K. Das, M. T. Leung, and K.-K. R. Choo, "Puf-based authentication and key agreement protocols for iot, wsns, and smart grids: a comprehensive survey," *IEEE Internet of Things Journal*, vol. 9, no. 11, pp. 8205–8228, 2022.

[164] A. Lakhan, M. A. Mohammed, K. H. Abdulkareem, M. M. Jaber, J. Nedoma, R. Martinek, and P. Zmij, "Delay optimal schemes for internet of things applications in heterogeneous edge cloud computing networks," *Sensors*, vol. 22, no. 16, p. 5937, 2022.

[165] A. Djenna, S. Harous, and D. E. Saidouni, “Internet of things meet the internet of threats: New concern cyber security issues of critical cyber infrastructure,” *Applied Sciences*, vol. 11, no. 10, p. 4580, 2021.

[166] M. Almiani, A. AbuGhazleh, A. Al-Rahayfeh, S. Atiewi, and A. Razaque, “Deep recurrent neural network for iot intrusion detection system,” *Simulation Modelling Practice and Theory*, vol. 101, p. 102031, 2020.

[167] T. Rajmohan, P. H. Nguyen, and N. Ferry, “Research landscape of patterns and architectures for iot security: a systematic review,” in *2020 46th Euromicro conference on software engineering and advanced applications (SEAA)*. IEEE, 2020, pp. 463–470.

[168] N. Tekin, A. Acar, A. Aris, A. S. Uluagac, and V. C. Gungor, “Energy consumption of on-device machine learning models for iot intrusion detection,” *Internet of Things*, vol. 21, p. 100670, 2023.

[169] A. Hakiri, A. Gokhale, P. Berthou, D. C. Schmidt, and T. Gayraud, “Software-defined networking: Challenges and research opportunities for future internet,” *Computer Networks*, vol. 75, pp. 453–471, 2014.

[170] K. H. K. Reddy, A. K. Luhach, V. V. Kumar, S. Pratihar, D. Kumar, and D. S. Roy, “Towards energy efficient smart city services: A software defined resource management scheme for data centers,” *Sustainable Computing: Informatics and Systems*, vol. 35, p. 100776, 2022.

[171] A. Montazerolghaem, “Software-defined internet of multimedia things: Energy-efficient and load-balanced resource management,” *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 2432–2442, 2021.

[172] J. Liu, H. Shen, H. S. Narman, W. Chung, and Z. Lin, “A survey of mobile crowdsensing techniques: A critical component for the internet of things,” *ACM Transactions on Cyber-Physical Systems*, vol. 2, no. 3, pp. 1–26, 2018.

[173] B. B. Gupta and M. Quamara, “An overview of internet of things (iot): Architectural aspects, challenges, and protocols,” *Concurrency and Computation: Practice and Experience*, vol. 32, no. 21, p. e4946, 2020.

[174] D. Stiawan, M. Y. B. Idris, A. M. Bamhdi, R. Budiarto *et al.*, “Cicids-2017 dataset feature analysis with information gain for anomaly detection,” *IEEE Access*, vol. 8, pp. 132 911–132 921, 2020.

[175] R. Panigrahi and S. Borah, “A detailed analysis of cicids2017 dataset for designing intrusion detection systems,” *International Journal of Engineering & Technology*, vol. 7, no. 3.24, pp. 479–482, 2018.

[176] S. M. Kasongo and Y. Sun, “A deep learning method with wrapper-based feature extraction for wireless intrusion detection system,” *Computers & Security*, vol. 92, p. 101752, 2020.

[177] T. Gaber, A. El-Ghamry, and A. E. Hassanien, “Injection attack detection using machine learning for smart iot applications,” *Physical Communication*, vol. 52, p. 101685, 2022.

[178] A. A. Alsulami, Q. Abu Al-Haija, A. Tayeb, and A. Alqahtani, “An intrusion detection and classification system for iot traffic with improved data engineering,” *Applied Sciences*, vol. 12, no. 23, p. 12336, 2022.

[179] L. Yang, A. Moubayed, I. Hamieh, and A. Shami, “Tree-based intelligent intrusion detection system in internet of vehicles,” in *2019 IEEE global communications conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[180] Great Learning, “Label encoding in python,” https://www.mygreatlearning.com/blog/label-encoding-in-python/#:~:text=Label%20encoding%20is%20a%20simple,input%20into%20machine%20learning%20algorithms., n.d., accessed: 2024-03-21.

[181] Analytics Vidhya, “Overcoming class imbalance using smote techniques,” https://www.analyticsvidhya.com/blog/2020/10/overcoming-class-imbalance-using-smote-techniques/#:~:text=SMOTE%3A%20Synthetic%20Minority%20Oversampling%20Technique,-SMOTE%20is%20an&text=This%20algorithm%20helps%20to%20overcome,positive%20instances%20that%20lie%20together., 2020, accessed: 2024-03-21.

[182] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, “Convolutional, long short-term memory, fully connected deep neural networks,” in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. Ieee, 2015, pp. 4580–4584.

[183] L. Muhammad, A. A. Haruna, U. S. Sharif, and M. B. Mohammed, “Cnn-lstm deep learning based forecasting model for covid-19 infection cases in nigeria, south africa and botswana,” *Health and technology*, vol. 12, no. 6, pp. 1259–1276, 2022.

[184] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: concepts, cnn architectures, challenges, applications, future directions," *Journal of big Data*, vol. 8, pp. 1–74, 2021.

[185] G. Najera-Gutierrez and J. A. Ansari, *Web Penetration Testing with Kali Linux: Explore the methods and tools of ethical hacking with Kali Linux*. Packt Publishing Ltd, 2018.

[186] S. Asadollahi, B. Goswami, and M. Sameer, "Ryu controller's scalability experiment on software defined networks," in *2018 IEEE international conference on current trends in advanced computing (ICCTAC)*. IEEE, 2018, pp. 1–5.

[187] K. Kaur, J. Singh, and N. S. Ghumman, "Mininet as software defined networking testing platform," in *International conference on communication, computing & systems (ICCCS)*, 2014, pp. 139–42.

[188] L. St, S. Wold *et al.*, "Analysis of variance (anova)," *Chemometrics and intelligent laboratory systems*, vol. 6, no. 4, pp. 259–272, 1989.

[189] D. Breitenbacher, I. Homoliak, Y. L. Aung, N. O. Tippenhauer, and Y. Elovici, "Hades-iot: A practical host-based anomaly detection system for iot devices," in *Proceedings of the 2019 ACM Asia conference on computer and communications security*, 2019, pp. 479–484.

[190] B. Chen, Y. Zhang, G. Iosifidis, and M. Liu, "Reinforcement learning on computational resource allocation of cloud-based wireless networks," in *2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*. IEEE, 2020, pp. 1–6.

[191] R. D. Corin, A. Costanzo, F. Callegati, and D. Siracusa, "Methods and techniques for dynamic deployability of software-defined security services," *CoRR*, 2020.

[192] A. van de Ven. Powertop. https://github.com/fenrus75/powertop.

[193] N. F. Syed, Z. Baig, A. Ibrahim, and C. Valli, "Denial of service attack detection through machine learning for the iot," *Journal of Information and Telecommunication*, vol. 4, no. 4, pp. 482–503, 2020.

[194] K. Sonar and H. Upadhyay, "A survey: Ddos attack on internet of things," *International Journal of Engineering Research and Development*, vol. 10, no. 11, pp. 58–63, 2014.

[195] M. M. Raikar and S. Meena, “Ssh brute force attack mitigation in internet of things (iot) network: An edge device security measure,” in *2021 2nd international conference on secure cyber computing and communications (ICSCCC)*. IEEE, 2021, pp. 72–77.

[196] Q. A. Al-Haija, E. Saleh, and M. Alnabhan, “Detecting port scan attacks using logistic regression,” in *2021 4th International symposium on advanced electrical and communication technologies (ISAECT)*. IEEE, 2021, pp. 1–5.

[197] Z. Campbell, A. Bray, A. Ritz, and A. Groce, “Differentially private anova testing,” in *2018 1st International Conference on Data Intelligence and Security (ICDIS)*. IEEE, 2018, pp. 281–285.

[198] H. Wei and X. Song, “Smooth tests for normality in anova,” *arXiv preprint arXiv:2110.04849*, 2021.

[199] E. Frimpong, “A performance study of the snort ids,” 2008.

[200] D. Fadhilah and M. I. Marzuki, “Performance analysis of ids snort and ids suricata with many-core processor in virtual machines against dos/ddos attacks,” in *2020 2nd International Conference on Broadband Communications, Wireless Sensors and Powering (BCWSP)*. IEEE, 2020, pp. 157–162.

[201] M. Hawedi, C. Talhi, and H. Boucheneb, “Multi-tenant intrusion detection system for public cloud (mtids),” *The Journal of Supercomputing*, vol. 74, pp. 5199–5230, 2018.

[202] S. M. Raza, J. Jeong, M. Kim, B. Kang, and H. Choo, “Empirical performance and energy consumption evaluation of container solutions on resource constrained iot gateways,” *Sensors*, vol. 21, no. 4, p. 1378, 2021.

[203] W. Park and S. Ahn, “Performance comparison and detection analysis in snort and suricata environment,” *Wireless Personal Communications*, vol. 94, pp. 241–252, 2017.

[204] E. Ozturk Kiyak, B. Ghasemkhani, and D. Birant, “High-level k-nearest neighbors (hlknn): A supervised machine learning model for classification analysis,” *Electronics*, vol. 12, no. 18, p. 3828, 2023.

[205] E. Altulaihan, M. A. Almaiah, and A. Aljughaiman, “Anomaly detection ids for detecting dos attacks in iot networks based on machine learning algorithms,” *Sensors*, vol. 24, no. 2, p. 713, 2024.

[206] F. Khomh and S. A. Abtahizadeh, "Understanding the impact of cloud patterns on performance and energy consumption," *Journal of Systems and Software*, vol. 141, pp. 151–170, 2018.

[207] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering.* Springer Science & Business Media, 2012.

[208] S. Jamshidi. Experimental packages. https://1drv.ms/f/s!AsjiVVuJWVG6mvUazqzhsSNa6w3-1Q?e=w69LHJ.

[209] A. A. Laghari, K. Wu, R. A. Laghari, M. Ali, and A. A. Khan, "A review and state of art of Internet of Things (IoT)," *Archives of Computational Methods in Engineering*, pp. 1–19, 2021.

[210] B. K. Mohanta, D. Jena, U. Satapathy, and S. Patnaik, "Survey on iot security: Challenges and solution using machine learning, artificial intelligence and blockchain technology," *Internet of Things*, vol. 11, p. 100227, 2020.

[211] E. Schiller, A. Aidoo, J. Fuhrer, J. Stahl, M. Ziörjen, and B. Stiller, "Landscape of IoT security," *Computer Science Review*, vol. 44, p. 100467, 2022.

[212] A. Javadpour, F. Ja'fari, T. Taleb, M. Shojafar, and C. Benzaïd, "A comprehensive survey on cyber deception techniques to improve honeypot performance," *Computers & Security*, p. 103792, 2024.

[213] V. Patchava, H. B. Kandala, and P. R. Babu, "A smart home automation technique with Raspberry pi using IoT," in *2015 International conference on smart sensors and systems (IC-SSS).* IEEE, 2015, pp. 1–4.

[214] C. Toma, A. Alexandru, M. Popa, and A. Zamfiroiu, "IoT solution for smart cities' pollution monitoring and the security challenges," *Sensors*, vol. 19, no. 15, p. 3401, 2019.

[215] A. Rahman, T. Rahman, N. H. Ghani, S. Hossain, and J. Uddin, "IoT-based patient monitoring system using ECG sensor," in *2019 international conference on robotics, electrical and signal processing techniques (ICREST).* IEEE, 2019, pp. 378–382.

[216] D. Hasan and A. Ismaeel, "Designing ecg monitoring healthcare system based on Internet of Things(IoT) blink application," *Journal of applied science and technology trends*, vol. 1, no. 3, pp. 106–111, 2020.

[217] Flask, https://palletsprojects.com/p/flask/, 2023.

[218] Nginx, “NGINX: Advanced load balancer, web server, & reverse proxy,” https://www.nginx.com/, 2023.

[219] G. Chart, https://developers.google.com/chart/, 2023.

[220] Plotly, https://plotly.com/python/, 2023.

[221] E. Ahvar, A.-C. Orgerie, and A. Lebre, “Estimating energy consumption of cloud, fog, and edge computing infrastructures,” *IEEE Transactions on Sustainable Computing*, vol. 7, no. 2, pp. 277–288, 2019.

[222] H. K. Yugank, R. Sharma, and S. H. Gupta, “An approach to analyse energy consumption of an iot system,” *International Journal of Information Technology*, vol. 14, no. 5, pp. 2549–2558, 2022.

[223] D. J. Sheskin, *Handbook of parametric and nonparametric statistical procedures.* Chapman and Hall/CRC, 2003.

[224] A. Dmitrienko, G. Molenberghs, C. Chuang-Stein, and W. Offen, “Analysis of clinical trials using SAS: A practical guide. SAS institute, 2005,” *http://www.google.ca/books*.

[225] A. F. Otoom, E. E. Abdallah *et al.*, “Deep learning for accurate detection of brute force attacks on iot networks,” *Procedia Computer Science*, vol. 220, pp. 291–298, 2023.

[226] A. K. Sekar, R. Ramakrishnan, A. Ganesh, and T. Kiruthiga, “Emerging cyber security and brute force attacks in hospital management information systems,” in *2023 Second International Conference On Smart Technologies For Smart Nation (SmartTechCon)*. IEEE, 2023, pp. 421–426.

[227] O. Salem, K. Alsubhi, A. Shaafi, M. Gheryani, A. Mehaoua, and R. Boutaba, “Man-in-the-middle attack mitigation in the Internet of Medical Things,” *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 2053–2062, 2021.

[228] M. Shafiq, Z. Gu, O. Cheikhrouhou, W. Alhakami, and H. Hamam, “The rise of “internet of things”: Review and open research issues related to detection and prevention of iot-based security attacks,” *Wireless Communications and Mobile Computing*, vol. 2022, no. 1, p. 8669348, 2022.

[229] N. Mishra and S. Pandya, “Internet of things applications, security challenges, attacks, intrusion detection, and future visions: A systematic review,” *IEEE Access*, vol. 9, pp. 59 353–59 377, 2021.

[230] R. Johari, I. Kaur, R. Tripathi, and K. Gupta, “Penetration testing in iot network,” in *2020 5th International Conference on Computing, Communication and Security (ICCCS).* IEEE, 2020, pp. 1–7.

[231] A. Roets and B. L. Tait, “Iot-penn: A security penetration tester for mqtt in the iot environment,” in *Cybersecurity in the Age of Smart Societies: Proceedings of the 14th International Conference on Global Security, Safety and Sustainability, London, September 2022.* Springer, 2023, pp. 141–157.

[232] I. Cvitić, D. Peraković, M. Periša, A. Jevremović, and A. Shalaginov, “An overview of smart home iot trends and related cybersecurity challenges,” *Mobile Networks and Applications*, vol. 28, no. 4, pp. 1334–1348, 2023.

[233] R. O. Andrade, S. G. Yoo, L. Tello-Oquendo, and I. Ortiz-Garcés, “A comprehensive study of the iot cybersecurity in smart cities,” *IEEE Access*, vol. 8, pp. 228 922–228 941, 2020.

[234] A. J. Cartwright, “The elephant in the room: cybersecurity in healthcare,” *Journal of Clinical Monitoring and Computing*, vol. 37, no. 5, pp. 1123–1132, 2023.

[235] K. Doshi, Y. Yilmaz, and S. Uludag, “Timely detection and mitigation of stealthy ddos attacks via iot networks,” *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 5, pp. 2164–2176, 2021.

[236] P. Kumar, H. Bagga, B. S. Netam, and V. Uduthalapally, “Sad-IoT: Security analysis of DDoS attacks in IoT networks,” *Wireless Personal Communications*, vol. 122, no. 1, pp. 87–108, 2022.

[237] J. Thomas, S. Cherian, S. Chandran, and V. Pavithran, “Man in the middle attack mitigation in lorawan,” in *2020 International Conference on Inventive Computation Technologies (ICICT).* IEEE, 2020, pp. 353–358.

[238] E. D. Saputro, Y. Purwanto, and M. F. Ruriawan, “Medium interaction honeypot infrastructure on the Internet of Things(IoT),” in *2020 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS).* IEEE, 2021, pp. 98–102.

[239] D. Stiawan, M. Idris, R. F. Malik, S. Nurmaini, N. Alsharif, R. Budiarto *et al.*, “Investigating brute force attack patterns in IoT network,” *Journal of Electrical and Computer Engineering*, vol. 2019, 2019.

[240] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[241] A. Kumari and S. Tanwar, "Reinforcement learning for multiagent-based residential energy management system. in 2021 ieee globecom workshops (gc wkshps)(pp. 1–6)," *IEEE.(2021, December)*, 2021.

[242] C. I. for Cybersecurity, "CICIDS 2017 dataset," https://www.unb.ca/cic/datasets/ids-2017.html.

[243] A. Kumari, R. Kakkar, S. Tanwar, D. Garg, Z. Polkowski, F. Alqahtani, and A. Tolba, "Multi-agent-based decentralized residential energy management using deep reinforcement learning," *Journal of Building Engineering*, vol. 87, p. 109031, 2024.

[244] A. Kumari and S. Tanwar, "Al-based peak load reduction approach for residential buildings using reinforcement learning," in *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS).* IEEE, 2021, pp. 972–977.

[245] A. Shahidinejad and J. Abawajy, "An all-inclusive taxonomy and critical review of blockchain-assisted authentication and session key generation protocols for iot," *ACM Computing Surveys*, vol. 56, no. 7, pp. 1–38, 2024.

[246] R. Feldt and A. Magazinius, "Validity threats in empirical software engineering research-an initial survey." in *Seke*, 2010, pp. 374–379.

[247] S. Jamshidi, "Codes," 2024. [Online]. Available: https://www.dropbox.com/scl/fo/jay5jo3ksmsjni3rc9tr9/ACyvr23WjU_x-9kodBa-3Fk?rlkey=fdldg4j65m5j6z2hdgnawo0op&st=5v9rak3x&dl=0

[248] A. Kumari and I. Sharma, "Securing the internet of things using ai-enabled detection of attacks via port scans in iot networks," in *2023 International Conference on Power Energy, Environment & Intelligent Control (PEEIC).* IEEE, 2023, pp. 348–352.

[249] N. Ahmed, F. Hassan, K. Aurangzeb, A. H. Magsi, and M. Alhussein, "Advanced machine learning approach for dos attack resilience in internet of vehicles security," *Heliyon*, vol. 10, no. 8, 2024.

[250] C. S. Kalutharage, X. Liu, C. Chrysoulas, N. Pitropakis, and P. Papadopoulos, "Explainable ai-based ddos attack identification method for iot networks," *Computers*, vol. 12, no. 2, p. 32, 2023.

[251] S. Chamoli and V. Mittal, “Ddos landscape: Insight into attacks and tools,” in *2024 International Conference on Electrical Electronics and Computing Technologies (ICEECT)*, vol. 1. IEEE, 2024, pp. 1–6.

[252] K. Sahu, R. Kshirsagar, S. Vasudeva, T. Alzahrani, and N. Karimian, “Leveraging timing side-channel information and machine learning for iot security,” in *2021 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, 2021, pp. 1–6.

[253] S. A. A. Abir, A. Anwar, J. Choi, and A. Kayes, “Iot-enabled smart energy grid: Applications and challenges,” *IEEE access*, vol. 9, pp. 50 961–50 981, 2021.

[254] S. Jamshidi, A. Nikanjam, N. Kawser, F. Khomh, and M.-A. Hamdaqa, “Understanding the impact of iot security patterns on cpu usage and energy consumption on iot devices,” *Authorea Preprints*, 2024.

[255] UNICEF Canada, “Global goals: Sustainable development for every child’s future,” 2024. [Online]. Available: https://www.unicef.ca/en

[256] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani *et al.*, “Toward generating a new intrusion detection dataset and intrusion traffic characterization.” *ICISSp*, vol. 1, pp. 108–116, 2018.

[257] “Cicflowmeter,” https://github.com/ahlashkari/CICFlowMeter.

[258] X. Mu and M. F. Antwi-Afari, “The applications of internet of things (iot) in industrial management: a science mapping review,” *International Journal of Production Research*, vol. 62, no. 5, pp. 1928–1952, 2024.

[259] M. C. Yahia and D. Yahia, “Intrusion detection system (ids) in iot environment using machine learning techniques,” in *Advances in Intelligent Systems and Computing*. Cham: Springer, 2021, pp. 469–482.

[260] Z. Shah, I. Ullah, H. Li, A. Levula, and K. Khurshid, “Blockchain based solutions to mitigate distributed denial of service (ddos) attacks in the internet of things (iot): A survey,” *Sensors*, vol. 22, no. 3, p. 1094, 2022.

[261] E. Džaferović, A. Sokol, A. Abd Almisreb, and S. M. Norzeli, “Dos and ddos vulnerability of iot: a review,” *Sustainable Engineering and Innovation*, vol. 1, no. 1, pp. 43–48, 2019.

[262] L. Markowsky and G. Markowsky, "Scanning for vulnerable devices in the internet of things," in *2015 IEEE 8th International conference on intelligent data acquisition and advanced computing systems: technology and applications (IDAACS)*, vol. 1. IEEE, 2015, pp. 463–467.

[263] W. H. Hassan *et al.*, "Current research on internet of things (iot) security: A survey," *Computer networks*, vol. 148, pp. 283–294, 2019.

[264] M. u Nisa and K. Kifayat, "Detection of slow port scanning attacks," in *2020 International Conference on Cyber Warfare and Security (ICCWS).* IEEE, 2020, pp. 1–7.

[265] R. Christopher, "Port scanning techniques and the defense against them," 2002.

[266] A. A. Alahmadi, M. Aljabri, F. Alhaidari, D. J. Alharthi, G. E. Rayani, L. A. Marghalani, O. B. Alotaibi, and S. A. Bajandouh, "Ddos attack detection in iot-based networks using machine learning models: a survey and research directions," *Electronics*, vol. 12, no. 14, p. 3103, 2023.

[267] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis *et al.*, "Understanding the mirai botnet," in *26th USENIX security symposium (USENIX Security 17)*, 2017, pp. 1093–1110.

[268] M. A. Shyaa, N. F. Ibrahim, Z. Zainol, R. Abdullah, M. Anbar, and L. Alzubaidi, "Evolving cybersecurity frontiers: A comprehensive survey on concept drift and feature dynamics aware machine and deep learning in intrusion detection systems," *Engineering Applications of Artificial Intelligence*, vol. 137, p. 109143, 2024.

[269] United Nations, "Sustainable development goals," 2015, accessed: 2024-12-08. [Online]. Available: https://www.un.org/sustainabledevelopment/

[270] A. Marzano, D. Alexander, O. Fonseca, E. Fazzion, C. Hoepers, K. Steding-Jessen, M. H. Chaves, Í. Cunha, D. Guedes, and W. Meira, "The evolution of bashlite and mirai iot botnets," in *2018 IEEE Symposium on Computers and Communications (ISCC).* IEEE, 2018, pp. 00 813–00 818.

[271] G. Settanni, F. Skopik, A. Karaj, M. Wurzenberger, and R. Fiedler, "Protecting cyber physical production systems using anomaly detection to enable self-adaptation," in *2018 IEEE Industrial Cyber-Physical Systems (ICPS).* IEEE, 2018, pp. 173–180.

[272] A. Uprety and D. B. Rawat, “Reinforcement learning for iot security: A comprehensive survey,” *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8693–8706, 2020.

[273] R. Hertzog, J. O’Gorman, and M. Aharoni, “Kali linux revealed,” *Mastering the Penetration Testing Distribution*, 2017.

[274] O. Bouhamed, O. Bouachir, M. Aloqaily, and I. Al Ridhawi, “Lightweight ids for uav networks: A periodic deep reinforcement learning-based approach,” in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*. IEEE, 2021, pp. 1032–1037.

[275] A. Salam, “Internet of things for sustainable forestry,” in *Internet of Things for sustainable community development: Wireless communications, sensing, and systems*. Springer, 2024, pp. 147–181.

[276] T. Alam, “A reliable communication framework and its use in internet of things (iot),” *Authorea Preprints*, 2023.

[277] I. Gulatas, H. H. Kilinc, A. H. Zaim, and M. A. Aydin, “Malware threat on edge/fog computing environments from internet of things devices perspective,” *IEEE Access*, vol. 11, pp. 33 584–33 606, 2023.

[278] B. Esmaeili, A. Azmoodeh, A. Dehghantanha, G. Srivastava, H. Karimipour, and J. C.-W. Lin, “A gnn-based adversarial internet of things malware detection framework for critical infrastructure: Studying gafgyt, mirai and tsunami campaigns,” *IEEE Internet of Things Journal*, 2023.

[279] M. S. Essa and S. K. Guirguis, “Evaluation of tree-based machine learning algorithms for network intrusion detection in the internet of things,” *IT Professional*, vol. 25, no. 5, pp. 45–56, 2023.

[280] T. Zhukabayeva, L. Zholshiyeva, K. Ven-Tsen, A. Adamova, N. Karabayev, and E. Mardenov, “Comprehensive study on detecting multi-class classification of iot attack using machine learning methods,” *Journal of Robotics and Control (JRC)*, vol. 5, no. 6, pp. 1943–1956, 2024.

[281] L. R. Hazim, A. A. Jasim, O. Ata, and M. Ilyas, “Intrusion detection system (ids) of multiclassification iot by using pipelining and an efficient machine learning,” in *2023 International Conference on Engineering and Emerging Technologies (ICEET)*. IEEE, 2023, pp. 1–6.

[282] B. Bala and S. Behal, "Ai techniques for iot-based ddos attack detection: Taxonomies, comprehensive review and research challenges," *Computer science review*, vol. 52, p. 100631, 2024.

[283] F. Deldar and M. Abadi, "Deep learning for zero-day malware detection and classification: A survey," *ACM Computing Surveys*, vol. 56, no. 2, pp. 1–37, 2023.

[284] P. R. Agbedanu, S. J. Yang, R. Musabe, I. Gatare, and J. Rwigema, "A scalable approach to internet of things and industrial internet of things security: Evaluating adaptive self-adjusting memory k-nearest neighbor for zero-day attack detection," *Sensors*, vol. 25, no. 1, p. 216, 2025.

[285] A. Hazra, V. M. R. Tummala, N. Mazumdar, D. K. Sah, and M. Adhikari, "Deep reinforcement learning in edge networks: Challenges and future directions," *Physical Communication*, p. 102460, 2024.

[286] A. Ometov, O. L. Molua, M. Komarov, and J. Nurmi, "A survey of security in cloud, edge, and fog computing," *Sensors*, vol. 22, no. 3, p. 927, 2022.

[287] L. N. Steimle, D. L. Kaufman, and B. T. Denton, "Multi-model markov decision processes," *IISE Transactions*, vol. 53, no. 10, pp. 1124–1139, 2021.

[288] M. K. Nisar, "N-BaIoT Dataset: Network-based Detection of IoT Malware," https://www.kaggle.com/datasets/mkashifn/nbaiot-dataset, 2018.

APPENDIX A APPENDIX A

Table A.1 Guideline for selecting the six patterns.

Applications most important non-functional requirement	**OOC**	**PZH**	**WL**	**BL**	**SSN**
Security	😊	☹	😊	😊	😊
Energy efficiency	😐	😐	😐	😐	😐
CPU usage	😐	😐	😐	😐	😐

APPENDIX B REAL-TIME SYSTEM LOG: ADAPTIVE HARDWARE RESILIENCE / APPENDIX B

B.1 Real-Time System Log: Adaptive Hardware Resilience

The system logs in Figure B.1 demonstrate SecuEdge-DRL's adaptive resilience against cyber threats and hardware failures. The log captures real-time events where the system detects DoS, DDoS, and Port scanning attacks, dynamically applying mitigation policies. Additionally, it highlights hardware anomaly detection, where SecuEdge-DRL identifies CPU overheating and memory pressure, triggering preventive cooling mechanisms and dynamic load redistribution to maintain stability. To enhance clarity and emphasize critical events, the log is color-coded, where warnings, alerts, and critical failures are distinctly highlighted, ensuring that security responses and hardware resilience actions are easily identifiable. The final entries confirm that adaptive measures restore regular operation, ensuring long-term reliability in resource-constrained edge gateways.

```
2025-01-28 08:00:05 - INFO: System State: NORMAL OPERATION
2025-01-28 08:00:05 - INFO: CPU Usage: 35.2%, CPU Load: 38.1%, Energy: 2.3J

2025-01-28 14:15:30 - WARNING: DoS Hulk Detected – High Packet Volume
2025-01-28 14:15:30 - INFO: CPU Usage: 38.5%, CPU Load: 35.3%, Energy: 2.6J
2025-01-28 14:15:30 - POLICY: Rate Limiting Enabled

2025-01-28 15:45:18 - WARNING: DoS Golden Eyes Attack Detected – Anomalous HTTP
Requests
2025-01-28 15:45:18 - INFO: CPU Usage: 38.1%, CPU Load: 33.5%, Energy: 2.6J
2025-01-28 15:45:18 - POLICY: Filtering Suspicious Packets

2025-01-28 16:10:52 - CRITICAL: DDoS Attack Confirmed – High CPU Overload
2025-01-28 16:10:52 - INFO: CPU Usage: 50.8%, CPU Load: 46.5%, Energy: 2.6J
2025-01-28 16:10:52 - POLICY: Adaptive Filtering & IP Blacklisting Activated

2025-01-28 17:00:45 - ALERT: Edge Node Overheating Detected – Preventive Cooling Applied
2025-01-28 17:00:45 - INFO: CPU Temperature: 85°C (Critical Threshold Approaching)
2025-01-28 17:00:45 - POLICY: Cooling System Activated, Dynamic Load Redistribution

2025-01-28 18:22:33 - ALERT: Port Scan Activity Detected – Rapid Connection Attempts
2025-01-28 18:22:33 - INFO: CPU Usage: 36.4%, CPU Load: 36.2%, Energy: 2.5J
2025-01-28 18:22:33 - POLICY: Blocking Malicious IPs

2025-01-28 19:15:10 - WARNING: High Memory Pressure – System Optimization Required
2025-01-28 19:15:10 - INFO: Available Memory: 9.5% (Threshold Alert)
2025-01-28 19:15:10 - POLICY: Process Cleanup and Memory Reallocation Initiated

2025-01-28 20:30:21 - SUCCESS: Adaptive Measures Implemented – System Stabilizing
2025-01-28 20:30:21 - INFO: CPU Usage: 38.1%, CPU Load: 37.4%, Energy: 2.4J

2025-01-29 06:00:55 - SUCCESS: System Fully Recovered – Normal Operation Restored
2025-01-29 06:00:55 - INFO: CPU Usage: 35.0%, CPU Load: 38.0%, Energy: 2.3J
```

Figure B.1 Compact Real-Time System Log of SecuEdge-DRL Cybersecurity System