

# **Reinforcement Learning Enhanced Intrusion Detection System (IDS) for Autonomous Network Defence**

Abishik Macherla Vijayakrishna  
B.Eng(Hons) Cybersecurity & Forensics  
Student ID: 40594078

February 24, 2026

## **Abstract**

The aim of this project is to develop and evaluate a proof-of-concept Intrusion Detection System (IDS) enhanced by Reinforcement Learning (RL) agents for autonomous network defence. Traditional IDSs rely on static signatures or supervised machine learning models that lack the ability to adapt to novel, zero-day threats in real-time. This research integrates RL agents into a simulated network defence environment to create adaptive response mechanisms capable of continuous learning from feedback. Leveraging the CIC-IDS2017 dataset and OpenAI Gym-compatible environment design, we implement and compare two RL algorithms of Deep Q-Network (DQN) and Proximal Policy Optimization (PPO) against traditional machine learning baseline including XGBoost. The experimental methodology includes standard classification scenarios and zero-day attack simulations where specific attack types are excluded during training to evaluate generalization capabilities. While the machine learning baselines achieved near-perfect accuracy, the RL agents demonstrated a security-first approach with significantly higher recall rates, effectively prioritizing threat detection at the cost of higher false positives. These results confirm that RL agents can be tuned towards high-security postures through asymmetric reward engineering, offering a promising direction for autonomous network defence where missing an attack carries greater consequence than a false alarm.

# 1 Introduction

Intrusion detection has long been a challenge for cybersecurity professionals of the operations industry [1]. As cyber threats continue to evolve in both scale and sophistication, the need for robust and adaptive Intrusion Detection Systems (IDS) has never been greater. The persistence of high-impact breaches is fundamentally tied to “dwell time” the window between initial system compromise and final containment. Recent experimental evidence indicates that the global average duration to identify and contain a data breach in 2025 has settled at approximately 241 days [2]. These figures underscore a critical failure in traditional, static security architectures and mandate a transition toward autonomous network defence systems capable of reasoning and responding at machine speed.

Traditional IDS approaches, including signature-based and anomaly-based detection methods, have shown significant limitations in identifying novel attacks and adapting to rapidly changing threat landscapes. Signature-based systems maintain databases of known attack fingerprints, achieving high accuracy for documented threats but remaining fundamentally blind to zero-day exploits [3]. Anomaly-based systems model normal behaviour and flag deviations, theoretically enabling novel threat detection, but suffer from high false positive rates leading to “alert fatigue” in Security Operations Centers (SOCs) [4]. Recent years have witnessed a surge in the application of machine learning to cybersecurity, with supervised models achieving near-perfect accuracy on benchmark datasets [5]. However, these solutions are not without limitations: supervised machine learning models, while achieving high accuracy on training distributions, remain fundamentally static after training, unable to adapt to evolving attacker tactics without complete retraining.

This has led researchers to explore more advanced AI techniques, with Reinforcement Learning (RL) emerging as the most promising paradigm for achieving autonomous defence [6]. Unlike traditional machine learning approaches that learn from static datasets of labelled examples, an RL agent learns through trial-and-error interaction with an environment, receiving feedback in the form of rewards for its actions [7]. This feedback-driven learning mechanism offers a compelling property for intrusion detection: the ability to continuously refine detection policies based on the success or failure of interventions. When an RL agent misclassifies a threat, it receives a negative reward and adjusts its behaviour accordingly, potentially enabling adaptation to new attack patterns without complete model retraining.

The motivation behind this study stems from the need to address the limitations of current IDS technologies and leverage the adaptive properties of reinforcement learning in the cybersecurity domain. In conventional SOC environments, human-driven detection and verification processes can result in dwell times measured in hours or even days, providing attackers enough opportunity to establish persistence or exfiltrate sensitive data. Organisations utilising extensive AI and automation in their security operations have demonstrated an ability to shorten the breach lifecycle by up to 80 days, significantly mitigating long-term costs [2]. An autonomous RL agent, by contrast, can evaluate and respond to potential threats in seconds, potentially reducing this window of vulnerability significantly.

On this investigation, the following research objectives were:

1. To implement and compare two Deep Reinforcement Learning algorithms: Deep Q-Network (DQN) and Proximal Policy Optimisation (PPO) for network intrusion detection within a simulated environment.
2. To evaluate the RL agents' performance against traditional supervised machine learning baselines (Random Forest and XGBoost) using the CIC-IDS2017 benchmark dataset.
3. To assess the agents' generalisation capabilities through zero-day attack simulation, where specific attack categories are withheld during training and evaluated during testing.
4. To analyse the agents' response to asymmetric reward structures, examining whether reward engineering can effectively prioritise high recall (threat detection) over raw classification accuracy.

This work contributes to the growing field of Autonomous Cyber Defence (ACD). By implementing an OpenAI Gym-compatible environment inspired by the *CybORG++* framework design principles [8], we address limitations of unstable legacy environments found in previous research. The project demonstrates the viability of RL agents not merely as classifiers, but as adaptive decision-makers whose risk tolerance can be tuned through reward engineering alone. A Streamlit-based dashboard provides real-time training visualisation and hyperparameter tuning capabilities, enabling iterative refinement of agent behaviour.

The structure of this paper is as follows. Section 2 provides background information on traditional intrusion detection approaches, the application of machine learning in network security, and the emerging role of reinforcement learning in autonomous cyber defence. Section 3 outlines the experimental methodology, which includes environment setup, dataset preprocessing, and the implementation of DQN, PPO, and machine learning baseline models. Section 4 presents the experimental results, with a focus on performance comparisons across standard and zero-day scenarios. Section 5 discusses the implications of the findings, addresses limitations and challenges, and outlines directions for future research. Section 6 concludes by summarising the key findings and their significance for adaptive intrusion detection.

## 2 Literature Review

### 3 Intrusion Detection Systems (IDS)

An Intrusion Detection System (IDS) acts like a “digital alarm system” for a computer network. It’s main job is to monitor all the activity on the network traffic and identify any potential security breaches or malicious activities. When it detects something suspicious, it logs the event and alerts a human security analyst. As the volume and complexity of cyber-attacks grow, these automated systems are essential for protecting critical data [1].

There are two main types of IDS, and some modern systems use a hybrid approach [9].

**Signature-Based IDS** acts similarly to antivirus software, maintaining a database of “signatures” or digital footprints belonging to known cyber-attacks. It scans network traffic against this list, making it extremely accurate for catching known threats. However, its significant weakness is inflexibility; it cannot detect “zero-day” attacks or novel threats that lack a pre-existing signature.

**Anomaly-Based IDS** works by building a model of normal network traffic behaviour rather than looking for specific attack signatures. It flags activity that deviates from this baseline. In theory, this approach allows for the detection of zero-day attacks. However, it often suffers from unreliability due to high rates of false positives. Harmless but unusual actions, such as an employee logging in at an unusual time, may be flagged, leading to “alert fatigue” for human analysts who may ignore real threats amidst the noise.

This project focuses on improving the anomaly-based approach, aiming to retain its ability to detect new threats while mitigating the issues of high false positives and inflexibility.

## 4 Reinforcement Learning for Adaptive Defence

Researchers and organisations have turned to a new kind of artificial intelligence to address these limitations. This section introduces Reinforcement Learning (RL), which is the core technology for this project.

### 4.1 Machine Learning vs Reinforcement Learning

Standard Supervised Machine Learning (ML) trains a model using large, labelled datasets, such as the CIC-IDS2017 dataset used in this work [10]. These models learn patterns by processing examples of “normal” and “attack” traffic, resulting in a static artifact that classifies data based on its training. A significant limitation of this approach is its inability to adapt post-training.

Reinforcement Learning (RL), in contrast, learns from experience rather than a static “answer key” [7]. An RL system learns through trial-and-error, interacting with an environment to maximise a reward signal.

## 4.2 The RL Agent and Agentic Security

The distinction between a static ML “model” and an RL “agent” is central to this research. While an ML model is a static file that makes predictions, an RL agent is a dynamic system capable of action and learning. The agent may use a deep neural network as its “brain,” but acts as an autonomous entity that observes and reacts to its surroundings.

In cybersecurity, “Agentic RL” systems function as autonomous defenders. Unlike static rule-based systems, an agentic system enables continuous monitoring and adaptive responses. The agent observes the **state** of the network traffic, selects an **action** (such as flagging, blocking, or passing traffic), and receives a **reward** based on the outcome (e.g., positive reinforcement for stopping an attack, negative for a false positive). A persistent challenge for these agents is “partial observability,” where the agent must infer threats from limited data, as it may not see the entire network state or hidden attacker movements.

## 4.3 Algorithm Selection: DQN vs. PPO

Choosing the right RL algorithm is critical for the stability and performance of an IDS. Two dominant algorithms in Deep Reinforcement Learning (DRL) are frequently compared:

**Deep Q-Network (DQN)** is a value-based method that learns the value of taking a specific action in a specific state [11]. It is particularly well-suited for discrete action spaces, such as an IDS deciding to simply “Block” or “Allow” a packet. Alavizadeh et al. [12] demonstrated that DQN-based IDS can provide “ongoing auto-learning capability” that detects different types of network intrusions through trial-and-error. While sample-efficient, DQN can sometimes be unstable during training.

**Proximal Policy Optimization (PPO)**, introduced by Schulman et al. [13], is a policy-gradient method. It is often praised for its stability and ease of tuning compared to other policy gradient methods. Liang et al. [14] recently proposed a PPO-based model that allows for dynamic adaptation of defensive strategies in response to evolving environmental patterns, demonstrating that the agent can “learn and optimise detection strategies” through continuous interaction. For simple discrete tasks, PPO may be computationally heavier than DQN, but its “trust region” update mechanism prevents destabilising policy changes [15].

For this project, both algorithms are implemented to compare their performance characteristics: DQN for its sample efficiency in discrete classification, and PPO for its training stability and adaptability.

## 5 Enhancing IDS with RL

The primary advantage of combining an RL agent with an IDS is adaptability. When a brand-new attack appears, traditional Signature-Based IDSs fail due to the lack of a pre-existing signature, and Static ML Baselines often fail because they were not trained on the specific pattern. An RL agent might also fail initially. However, through feedback (simulated or human-in-the-loop), it receives a negative reward for the failure. The agent updates its policy and learns from this mistake, making it more likely to detect similar patterns in the future. This process also reduces false positives; if the agent flags harmless user activity, a negative reward teaches it to adjust its definition of “normal” behaviour for that specific network.

## 5.1 Avoiding Dwell Time

In cyberdefence, “Dwell time”, the time an attack remains undetected in a network is a critical metric. Human detection often yields dwell times measured in hours or days. In contrast, an automated RL agent can flag anomalies in seconds. By automating the initial detection and triage, RL agents can drastically reduce this window of vulnerability, preventing attackers from establishing persistence or exfiltrating data.

## 5.2 Improving the RL Model Accuracy

To make RL-based IDS practical, accuracy must be high. Research focuses on several areas to improve this.

**Feature Engineering** involves selecting the optimal features from network traffic; techniques like ID-RDRL [16] combine recursive feature elimination with Deep RL to improve the detection of unknown attacks.

**Reward Tuning** is also essential, as simple positive/negative rewards can be too sparse to guide learning effectively. Tuning involves providing intermediate rewards, such as penalising false negatives (missed attacks) more heavily than false positives.

**Handling Class Imbalance** is crucial, as attack traffic is rare compared to normal traffic. Shanmugam et al. [4] provide a comprehensive evaluation of class imbalance techniques in IDS, demonstrating that resampling strategies significantly affect model reliability. Techniques such as oversampling, undersampling, or using synthetic attacks in the training environment help the agent learn to identify rare threats.

## 5.3 Zero-Day Attack Detection

A primary strength for RL-based IDS is the identification of zero-day vulnerabilities security weaknesses that have not yet been disclosed or patched. Traditional methods, which rely on historical data, are fundamentally limited in detecting novel threats. Alam et al. [3] propose a Deep Reinforcement Learning-based NIDS designed specifically for zero-day attack detection, “utilising learned features from other known attacks” to identify previously unseen malicious patterns. Their approach treats zero-day attacks as an anomaly detection problem, excluding specific attack categories during training and evaluating the agent’s ability to detect these held-out attacks during testing. This methodology directly informs our experimental design.

## 5.4 The Datasets

A major criticism of many IDS studies is the use of outdated dataset like NSL-KDD. These datasets lack the traffic diversity and modern attack patterns found in today’s networks [17].

This project utilizes the CIC-IDS2017 dataset [18]. Unlike its predecessors, CIC-IDS2017 includes valid pcap data generated from a realistic testbed, containing benign traffic alongside modern attacks such as DDoS, Brute Force, and Web Application attacks. This ensures that the RL agent is trained on patterns relevant to current cybersecurity threats rather than historical artifacts.

## 5.5 Challenges in Real life

Deploying an RL agent into a live network introduces unique risks that supervised models do not face.

**Catastrophic Forgetting:** As an RL agent continues to learn from new attacks, it risks overwriting the knowledge it gained about older attacks as known as catastrophic forgetting. Techniques like Elastic Weight Consolidation (EWC) [19] or continual learning strategies [20] are required to ensure the agent retains its past experience.

**The “Cold Start” Problem:** An untrained RL agent explores by trial-and-error, which is dangerous in a live network (e.g., randomly blocking legitimate users). To mitigate this, agents must undergo a “pre-training” phase in a simulated environment (offline RL) before being deployed to the real world [21].

## 6 Related Research

In today’s AI and Cyber era, RL for cybersecurity is growing. Yang et al. [22] provide a comprehensive survey of Deep RL methods for intrusion detection, highlighting architectures and challenges.

**Anomaly Detection Approaches:** Hsu and Matsuoka [11] demonstrate implementations of Deep Q Networks (DQN) for anomaly detection, proving the concept. Malik and Singh Saini [23] showed that RL agents could outperform traditional ML models. Suwannalai and Polprasert [24] also explored adversarial RL with Deep Q-Networks.

**Multi-Agent and Generalisable Approaches:** Tellache et al. [25] have expanded this to Multi-agent systems. Latest research by Dudman and Bull [26] on Generalisable Cyber Defence Agents shows the push towards agents that can operate in complex, real-world environments.

**Real-World Applications:** Beyond academic prototypes, major tech organisations are investing in RL for security. Use cases include Microsoft’s *CyberBattleSim*, which uses RL agents to simulate lateral movement in a network, helping defenders understand attacker behaviour. Foley et al. [27] explored the practical implementation of PPO for real-time network remediation, focusing on the agent’s ability to balance defensive actions with network availability. This moves RL from purely detection (IDS) to proactive defence and autonomous cyber operations (ACO).

**Comparative Studies:** Recent work by Mondragon Guadarrama et al. [5] presents a benchmark of fourteen preprocessed datasets to address the lack of standardisation in IDS research, evaluating multiple algorithm categories including Random Forest and XGBoost. This comprehensive approach validates the importance of comparing RL agents against strong ML baselines. Fathi et al. [28] provide a taxonomy of modern RL techniques for intrusion detection, highlighting the transition from single-agent to multi-agent and adversarial frameworks.

## 7 Problem Formulation and Threat Model

This section formally defines the intrusion detection problem as a decision-making task amenable to reinforcement learning and specifies the threat model under which the autonomous defence system operates.

## 7.1 Threat Model

A rigorous threat model is essential to justify claims of “autonomous defence.” The following assumptions define the security context:

### 7.1.1 Attacker Capabilities

The adversary is assumed to possess the following capabilities, informed by contemporary threat intelligence [29, 30]:

- **Novel Attack Generation:** Attackers can craft previously unseen attack patterns (zero-day exploits) that lack signatures in traditional databases.
- **Evasion Techniques:** Sophisticated attackers employ polymorphic malware, steganographic techniques, and encrypted payloads to evade pattern-based detection [30].
- **Adversarial Manipulation:** In advanced scenarios, attackers may attempt to poison the learning process or craft adversarial inputs specifically designed to evade ML-based detection [31].
- **Volume Attacks:** Distributed Denial of Service (DDoS) attacks can generate massive traffic volumes to overwhelm both network infrastructure and detection systems.

### 7.1.2 Defender Constraints

The autonomous defence system operates under the following constraints:

- **Limited Visibility:** The agent observes only flow-level features extracted from network traffic, not full packet payloads or system-level indicators. This represents partial observability of the true network state.
- **Analyst Bandwidth:** Human security analysts have limited capacity to review alerts. The system must minimise false positives while maximising true threat detection.
- **Acceptable Disruption Rate:** Blocking legitimate traffic has business costs. The defender must maintain an acceptable false positive rate to avoid disrupting normal operations.
- **Latency Requirements:** Detection and response must occur within milliseconds to prevent attackers from establishing persistence or exfiltrating data [2].

## 7.2 Formal RL Formulation

The intrusion detection task is formalised as a Markov Decision Process (MDP), with acknowledgment of partial observability characteristics:

### 7.2.1 State/Observation Space

The observation  $o_t \in \mathcal{O}$  at timestep  $t$  consists of an 80-dimensional feature vector extracted from a network flow:

$$o_t = [f_1, f_2, \dots, f_{80}] \quad (1)$$

where features include flow duration, packet counts, byte statistics, inter-arrival times, and flag counts. All features are normalised to  $[0, 1]$  through min-max scaling.

### 7.2.2 Action Space

The action space  $\mathcal{A}$  is discrete with two elements:

$$\mathcal{A} = \{0 : \text{Allow}, 1 : \text{Block}\} \quad (2)$$

This binary formulation represents the fundamental classification decision. Extended action spaces (rate-limit, quarantine, escalate) are considered as future work.

### 7.2.3 Reward Function

The reward function  $R(s, a, s')$  encodes the security-first priority through asymmetric penalties:

$$R(a, y) = \begin{cases} +10 & \text{if } a = 1 \text{ and } y = 1 \text{ (True Positive)} \\ +1 & \text{if } a = 0 \text{ and } y = 0 \text{ (True Negative)} \\ -1 & \text{if } a = 1 \text{ and } y = 0 \text{ (False Positive)} \\ -10 & \text{if } a = 0 \text{ and } y = 1 \text{ (False Negative)} \end{cases} \quad (3)$$

where  $y \in \{0, 1\}$  is the ground truth label. This 10:1 asymmetry between missed attacks (FN) and false alarms (FP) reflects the security principle that undetected intrusions pose greater risk than minor user disruption.

### 7.2.4 Justification of RL Framework

A legitimate question is whether reinforcement learning is necessary for this task, or whether cost-sensitive classification would suffice. Several factors justify the RL approach:

1. **Reward Engineering Flexibility:** Unlike loss function weighting in supervised learning, RL rewards can encode complex, multi-objective preferences. Different attack types could receive different reward magnitudes based on threat severity, and the reward structure can be modified without retraining the underlying model architecture.
2. **Foundation for Online Adaptation:** While this study employs offline training on static datasets, the RL framework provides a natural pathway to online learning. Future deployments could update from analyst feedback, incident confirmations, or red-team exercises without architectural changes [32].
3. **Demonstrated Efficacy:** Recent systematic reviews confirm that DRL “greatly improves IDS performance in IoT: it boosts detection accuracy, reduces false alarms, and adapts in real time” [32]. Military and defense organisations have achieved successful proof-of-concepts where RL agents outperform human-defined rule sets [33].
4. **Simulation Environment Compatibility:** The RL formulation enables training in cyber simulation environments like CybORG++ [8] and CyberBattleSim [34], facilitating curriculum learning and multi-agent training scenarios [35].

### 7.2.5 Acknowledgment of Limitations

The current implementation operates on a static, labelled dataset with immediate feedback, which more closely resembles offline RL or contextual bandits than fully online sequential decision-making. The “ground truth” labels provide an idealised reward signal that would not

be available in production deployments, where feedback is delayed, partial, and noisy (e.g., analyst confirmations may arrive hours after initial detection). This limitation is explicitly acknowledged, and the experimental results should be interpreted as a proof-of-concept for the reward engineering approach rather than a claim of production-ready autonomous defence.

## 8 Methodology

### 8.1 Environment Design

A custom OpenAI Gymnasium-compatible environment (`IdsEnv`) was implemented to simulate network traffic processing as a sequential decision-making task. The environment was designed following principles from the CybORG++ framework [8], adapted for intrusion detection classification.

- **State Space:** The observation space consisted of 78 normalised network flow features extracted from the CIC-IDS2017 dataset. Features included packet size statistics, inter-arrival times, flow duration, and payload characteristics. All features were scaled to the range  $[0, 1]$  using min-max normalisation to ensure stable neural network training.
- **Action Space:** A discrete action space of size 2 was defined, where action 0 represented “Allow” (classify as benign) and action 1 represented “Block” (classify as malicious).
- **Reward Function:** A configurable reward structure was implemented to allow systematic investigation of how reward asymmetry affects agent behaviour. The default configuration used:
  - True Positive (correctly blocking an attack): +10
  - True Negative (correctly allowing benign traffic): +1
  - False Negative (missing an attack): -10
  - False Positive (incorrectly blocking benign traffic): -1

Five alternative reward configurations were subsequently tested as part of the hyperparameter experiments (Section 8.5).

- **Episode Structure:** Each episode began at a random position in the dataset to prevent the agent from overfitting to initial samples. The episode terminated after processing a fixed number of samples (1,000 steps) or upon reaching the dataset boundary.

### 8.2 Dataset Preprocessing

Two benchmark datasets were used in this study:

#### 8.2.1 CIC-IDS2017 (Primary Dataset)

The CIC-IDS2017 benchmark dataset [18] was selected for its realistic traffic generation methodology and inclusion of modern attack types (DDoS, PortScan, Web Attacks, Brute Force, Infiltration). The following preprocessing pipeline was applied:

- **Data Cleaning:** Infinite values and NaN entries were removed. Duplicate entries were dropped to reduce redundancy.
- **Feature Selection:** 78 features were retained after removing identifier columns (source/destination IP, ports) that could cause data leakage. Features included flow-level statistics such as packet lengths, inter-arrival times, and flag counts.

- **Label Encoding:** Multi-class attack labels were encoded into binary format (0 = Benign, 1 = Attack) for the primary classification task. Raw labels were preserved separately for zero-day simulation experiments.
- **Train/Test Split:** An 80/20 stratified random split was applied, maintaining class distribution across both sets. The training set contained approximately 2.27 million samples, with the test set containing approximately 565,576 samples.
- **Normalisation:** Min-max scaling was applied to all features independently to ensure values fell within the  $[0, 1]$  range required by the RL environment.

### 8.2.2 CIC-IoT-2023 (Cross-Dataset Evaluation)

The CIC-IoT-2023 dataset was used exclusively for Scenario 4 (cross-dataset generalisation). This dataset contains IoT network traffic with a different feature schema than CIC-IDS2017. To enable cross-dataset evaluation, a feature mapping was developed that identified 12 common features present in both datasets:

- Flow duration, total forward/backward packets
- Forward/backward packet length statistics (mean, max, min, std)
- Flow bytes per second, flow packets per second
- Average packet size, packet length variance

Both datasets were preprocessed using the same pipeline and saved as Parquet files for efficient loading during training.

### 8.2.3 Data Leakage Controls

To ensure evaluation validity and address concerns regarding inflated performance metrics common in IDS research [5], the following controls were implemented:

1. **Identifier Removal:** Source/destination IP addresses, port numbers, and flow identifiers were excluded from the feature set.
2. **Stratified Splitting:** The train/test split was stratified by class label to maintain consistent attack-to-benign ratios.
3. **Normalisation After Splitting:** Min-max scaling was fitted only on the training set. The test set was transformed using the training set statistics to prevent information leakage from future observations.
4. **Zero-Day Exclusion Protocol:** For zero-day experiments (Scenarios 2 and 3), entire attack categories were excluded from training rather than individual samples. This ensures the agent has received no exposure to the attack class being evaluated.

## 8.3 Algorithm Implementations

### 8.3.1 Deep Q-Network (DQN)

The DQN agent was implemented using a custom PyTorch neural network architecture to approximate Q-values for the discrete action space. The implementation followed the original DQN algorithm [36] with the following specifications:

- **Network Architecture:** A feedforward neural network with two hidden layers of 64 neurons each, using ReLU activation functions. The input layer accepted 78 features, and the output layer produced Q-values for 2 actions.
- **Experience Replay:** A replay buffer of 100,000 transitions was maintained to break temporal correlations and improve sample efficiency. Mini-batches of 64 transitions were sampled uniformly for training updates.
- **Target Network:** A separate target network was used with soft updates ( $\tau = 0.001$ ) to stabilise training.
- **Exploration Strategy:** An  $\varepsilon$ -greedy policy was employed with  $\varepsilon$  decaying from 1.0 to 0.01 over training using a decay factor of 0.999 per episode.
- **Training Parameters:** The agent was trained for 2,000 episodes (baseline) or up to 5,000 episodes (extended experiments) with a maximum of 1,000 steps per episode. Learning rate was set to  $5 \times 10^{-4}$  with the Adam optimiser and discount factor  $\gamma = 0.99$ .

### 8.3.2 Proximal Policy Optimization (PPO)

A PPO agent was implemented using the Stable-Baselines3 library [37] to leverage its policy-gradient stability:

- **Policy Network:** A shared actor-critic architecture with two hidden layers of 64 neurons each.
- **Training Parameters:** The agent was initially trained for 200,000 timesteps and subsequently retrained with 1,000,000 timesteps. Key hyperparameters included:
  - Learning rate:  $3 \times 10^{-4}$
  - N steps per update: 2,048
  - Batch size: 64
  - Number of epochs per update: 10
  - Discount factor ( $\gamma$ ): 0.99
  - GAE lambda: 0.95
  - Clip range: 0.2
  - Entropy coefficient: 0.01
- **Clipping Mechanism:** The PPO clipping objective constrains the policy ratio within  $[1 - \varepsilon, 1 + \varepsilon]$  where  $\varepsilon = 0.2$ , preventing destabilising policy updates.

### 8.3.3 Machine Learning Baselines

Random Forest and XGBoost classifiers were trained as supervised learning baselines:

- **Random Forest:** Implemented using scikit-learn with 100 estimators, maximum depth of 20, and minimum samples split of 5.
- **XGBoost:** Implemented using the XGBoost library with 100 estimators, maximum depth of 6, and learning rate of 0.1. Both models were trained on the same 80% training split and evaluated on the held-out 20% test set.

## 8.4 Evaluation Scenarios

Four experimental scenarios were designed to comprehensively evaluate detection capabilities and adaptability:

- **Scenario 1 (Standard Classification):** All models were trained and tested on the full CIC-IDS2017 dataset using the 80/20 split to evaluate baseline detection capabilities on known attack types.
- **Scenario 2 (Zero-Day DDoS Simulation):** The RL agents were trained on all traffic *excluding* DDoS attacks, then tested specifically on the held-out DDoS samples (128,025 samples). This simulated encountering a novel high-volume threat not seen during training. ML baselines trained *with* DDoS were also evaluated for comparison.
- **Scenario 3 (Zero-Day Web Attack Simulation):** Similar to Scenario 2, but holding out Web Application attacks (SQL Injection, XSS, Brute Force) — 2,180 test samples. This tested detection of stealthier application-layer threats.
- **Scenario 4 (Cross-Dataset Generalisation):** All models were trained on CIC-IDS2017 using only the 12 common features mapped to CIC-IoT-2023. Evaluation was performed on the CIC-IoT-2023 test set (937,393 samples) to assess transferability across different network environments and time periods.

## 8.5 DQN Hyperparameter Experiments

To systematically investigate how reward structure, network architecture, and training duration affect agent behaviour, 8 DQN experiments were conducted across two phases:

**Phase 1 (Reward Structure Tuning):** Five reward configurations were tested with the same base architecture (64-64 network, 2,000 episodes):

Table 1: Phase 1: Reward Structure Configurations

Experiment	TP	TN	FN	FP
Exp 1: Baseline (10:1)	+10	+1	-10	-1
Exp 2: Lower FN Penalty	+10	+1	-5	-1
Exp 3: Higher FP Penalty	+10	+1	-10	-3
Exp 4: 5:1 Ratio	+5	+1	-5	-1
Exp 5: Symmetric	+1	+1	-1	-1

**Phase 2 (Architecture and Training):** Using the best reward configuration from Phase 1 (symmetric), three additional experiments varied network size, training duration, and learning rate:

- Exp 6: Larger network (128-128 neurons)
- Exp 7: Extended training (5,000 episodes)
- Exp 8: Slower learning rate ( $1 \times 10^{-4}$ )

## 8.6 Evaluation Metrics

- Standard metrics including Accuracy, Precision, Recall, and F1-Score were recorded for all models.
- Emphasis was placed on **Recall** (Detection Rate), as missing an attack in a security context is more critical than a false positive.
- Confusion matrices were generated to visualise the breakdown of True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN).
- For Scenario 4, ROC-AUC scores and inference latency were additionally measured.

## 9 Results

This section presents the experimental results from training and evaluating the RL agents and ML baselines across all four scenarios, followed by the DQN hyperparameter experiment findings.

### 9.1 Scenario 1: Standard Classification

Table 2 presents the performance of all models on the standard 80/20 train/test split of the CIC-IDS2017 dataset (78 features, 565,576 test samples).

Table 2: Standard Classification Performance on CIC-IDS2017

Model	Accuracy	Precision	Recall	F1 Score
Random Forest	99.90%	99.64%	99.86%	99.75%
XGBoost	99.95%	99.83%	99.90%	99.87%
DQN (Exp 7)	96.62%	95.93%	86.49%	90.97%

The supervised ML baselines achieved near-perfect classification accuracy, with XGBoost marginally outperforming Random Forest across all metrics. The DQN agent achieved 96.62% accuracy and 90.97% F1 score. While this is lower than the ML baselines, the agent learned to classify network traffic with over 90% harmonic accuracy using only reward signals, without access to labelled data during inference.

The DQN agent’s 86.49% recall indicates it missed approximately 15,039 attacks out of 111,311 in the test set, while generating 4,080 false positives. This precision-recall balance reflects the best reward configuration identified through the hyperparameter experiments (symmetric rewards, Section 9.5).

## 9.2 Scenario 2: Zero-Day DDoS Detection

Table 3 presents results where the DQN agent was trained *without* any DDoS attack samples, then tested on 128,025 DDoS-only samples. ML baselines trained *with* DDoS exposure are shown for comparison.

Table 3: Zero-Day DDoS Detection Performance

Model	Accuracy	Precision	Recall	F1 Score
<i>Trained WITHOUT DDoS (Zero-Day Scenario)</i>				
DQN (No DDoS)	56.77%	100.00%	56.77%	72.42%
<i>Trained WITH DDoS (Reference Baseline)</i>				
DQN (Standard)	98.59%	100.00%	98.59%	99.29%
Random Forest	99.97%	100.00%	99.97%	99.99%
XGBoost	99.92%	100.00%	99.92%	99.96%

The zero-day DQN agent successfully detected 72,674 out of 128,025 DDoS attacks (56.77% recall) that it had *never encountered during training*, while maintaining 100% precision — every sample it flagged as malicious was indeed a DDoS attack. This suggests the agent learned generalised indicators of anomalous network behaviour (e.g., volumetric patterns) that partially overlap with DDoS traffic characteristics, rather than memorising specific attack signatures.

The gap between 56.77% (zero-day) and 98.59% (standard DQN) quantifies the generalisation cost of encountering truly novel high-volume attacks. The ML baselines trained with DDoS exposure achieved near-perfect detection, confirming that DDoS patterns are highly learnable when present in training data.

## 9.3 Scenario 3: Zero-Day Web Attack Detection

Table 4 presents results where the DQN agent was trained without web attacks (SQL Injection, XSS, Brute Force), then tested on 2,180 web attack samples.

Table 4: Zero-Day Web Attack Detection Performance

Model	Accuracy	Precision	Recall	F1 Score
<i>Trained WITHOUT Web Attacks (Zero-Day Scenario)</i>				
DQN (No Web)	2.29%	100.00%	2.29%	4.48%
DQN (Standard)	3.67%	100.00%	3.67%	7.08%
<i>Trained WITH Web Attacks (Reference Baseline)</i>				
Random Forest	98.21%	100.00%	98.21%	99.10%
XGBoost	97.25%	100.00%	97.25%	98.60%

The DQN agent trained without web attacks achieved only 2.29% recall (50 out of 2,180 samples), constituting near-total failure. Even the standard DQN trained *with* web attacks achieved only 3.67% recall on these specific test samples. This result has important implications: web attacks (SQL injection, XSS) operate at the *application layer* and their distinguishing characteristics lie in HTTP payload content, which is not captured by the 78 flow-level network statistics in CIC-IDS2017. No amount of reward tuning can overcome a fundamental feature representation limitation.

The ML baselines (trained with web attacks) achieved 97–98% recall, likely by learning correlations between web attack patterns and *other* flow-level features present in the specific

dataset. However, even these models cannot generalise web attack detection beyond the dataset distribution.

## 9.4 Scenario 4: Cross-Dataset Generalisation

Table 5 presents results where all models were trained on CIC-IDS2017 using 12 mapped features and tested on the CIC-IoT-2023 dataset (937,393 samples).

Table 5: Cross-Dataset Generalisation (CIC-IDS2017  $\rightarrow$  CIC-IoT-2023)

Model	Accuracy	Precision	Recall	F1	Latency ( $\mu$ s)
Random Forest	38.26%	4.67%	0.03%	0.07%	1.05
XGBoost	38.65%	0.00%	0.00%	0.00%	0.55
DQN	12.44%	3.44%	1.58%	2.16%	<b>0.20</b>

All models failed catastrophically when evaluated on CIC-IoT-2023 traffic. The Random Forest achieved only 0.03% recall (detecting 191 out of 575,051 attacks), while XGBoost detected zero attacks entirely. The DQN agent, while achieving only 2.16% F1, was the only model to detect any meaningful number of attacks (9,079), though at the cost of 254,797 false positives.

This universal failure is attributed to the significant distribution shift between datasets: CIC-IDS2017 (2017). enterprise network traffic) and CIC-IoT-2023 (IoT device traffic) have fundamentally different traffic characteristics despite sharing 12 common feature names. This confirms that IDS models trained on one network environment cannot be directly transferred to another without retraining or domain adaptation.

Notably, the DQN agent achieved the fastest inference time at 0.20  $\mu$ s per sample (4.9 million samples per second), compared to 1.05  $\mu$ s for Random Forest and 0.55  $\mu$ s for XGBoost.

## 9.5 DQN Hyperparameter Experiments

Table 6 presents the results of 8 systematic DQN experiments across two phases.

Table 6: DQN Hyperparameter Experiment Results

Experiment	Acc%	Prec%	Rec%	F1%	FP	FN
<i>Phase 1: Reward Structure Tuning (64-64, 2000 ep)</i>						
E1: Baseline (10:1)	80.52	52.33	98.18	68.26	93,800	2,028
E2: Lower FN Penalty	80.74	52.61	97.84	68.44	92,692	2,407
E3: Higher FP Penalty	90.00	66.78	94.58	78.28	49,218	6,033
E4: 5:1 Ratio	79.43	51.25	98.47	67.42	98,004	1,706
E5: Symmetric (1:1)	96.08	94.56	84.85	89.44	5,144	16,865
<i>Phase 2: Architecture &amp; Training (symmetric rewards)</i>						
E6: 128-128 network	96.29	94.70	85.92	90.10	5,077	15,673
E7: 5000 episodes	96.62	95.93	86.49	90.97	4,080	15,039
E8: Slower LR ( $10^{-4}$ )	96.18	95.78	84.32	89.69	3,776	17,449

### 9.5.1 Phase 1 Findings: Reward Structure as a Policy Control Mechanism

The Phase 1 results demonstrate that reward structure is the *primary determinant* of agent behaviour. Across the five configurations:

- **Recall range:** 84.85% (symmetric) to 98.47% (5:1 ratio) — a 13.62 percentage point range controlled entirely by reward ratios.
- **FP range:** 4,080 (Exp 7, symmetric) to 98,004 (Exp 4, 5:1) — a  $24\times$  difference in false positive volume.
- **F1 Score:** Higher reward asymmetry (Exp 1, 4) achieved high recall ( $>98\%$ ) but poor F1 ( $\sim 68\%$ ) due to excessive false positives. The symmetric configuration (Exp 5) achieved the best F1 (89.44%) by balancing precision and recall.

This finding has practical implications: a network administrator could *tune* the IDS security posture by adjusting a single hyperparameter (reward ratio) rather than retraining or changing the model architecture.

### 9.5.2 Phase 2 Findings: Architecture and Training

With the symmetric reward structure fixed, three architectural variations were tested:

- **Larger network (Exp 6):** The 128-128 architecture provided minimal improvement ( $+0.66\%$  F1) over the 64-64 baseline, suggesting the bottleneck is not model capacity.
- **Extended training (Exp 7):** Training for 5,000 episodes yielded the best overall F1 (90.97%) with the lowest false positive count (4,080), representing the optimal configuration.
- **Slower learning rate (Exp 8):** Reducing the learning rate from  $5 \times 10^{-4}$  to  $1 \times 10^{-4}$  slightly reduced recall (84.32%) without improving F1, suggesting the original learning rate was already near-optimal.

## 10 Discussion

This section discusses the implications of the experimental findings, compares results with published research, addresses limitations, and outlines future directions.

### 10.1 Implications for Autonomous Network Defence

#### 10.1.1 RL as Tunable Risk Management

The 8 DQN hyperparameter experiments provide the strongest evidence for RL’s value proposition in IDS: the reward structure functions as a direct *policy control mechanism*. Across the five Phase 1 configurations, recall ranged from 84.85% to 98.47%, while false positives ranged from 4,080 to 98,004 — controlled entirely by modifying four reward parameters.

This tunability has practical significance for different deployment contexts:

- **High-Security Environments:** Critical infrastructure operators could deploy the asymmetric configuration (Exp 1/4, 98%+ recall) accepting  $\sim 94,000$  false positives as the cost of near-total attack detection.
- **User-Sensitive Contexts:** Consumer-facing services could deploy the symmetric configuration (Exp 5/7, 90%+ F1) with only  $\sim 4,000$  false positives.

This level of risk-posture adjustment through a single hyperparameter is not achievable with standard supervised learning, where cost-sensitive reweighting provides limited and less interpretable control.

### 10.1.2 Zero-Day Detection: Volumetric vs. Application-Layer Attacks

The contrasting zero-day results between DDoS (56.77% recall) and web attacks (2.29% recall) reveal a fundamental boundary condition for flow-level RL-IDS:

- **DDoS (56.77% recall):** DDoS attacks generate volumetric anomalies — high packet counts, unusual byte rates, abnormal flow durations — that share statistical characteristics with other attack types the agent learned during training. The agent’s 100% precision confirms it identified genuine anomalous patterns rather than making random guesses.
- **Web attacks (2.29% recall):** SQL injection, XSS, and brute force attacks are distinguished by their HTTP payload content, not by flow-level statistics. The 78 CIC-IDS2017 features capture network-layer behaviour but not application-layer semantics, making these attacks effectively invisible to any flow-level classifier — RL or ML.

This finding has implications for IDS architecture: effective detection of both volumetric and application-layer threats requires a multi-modal approach combining flow-level analysis with deep packet inspection (DPI).

### 10.1.3 Cross-Dataset Generalisation

The catastrophic failure of all models on CIC-IoT-2023 ( $F1 \leq 2.16\%$ ) demonstrates a well-documented challenge in IDS research: *distribution shift*. Despite mapping 12 common features, the statistical distributions of CIC-IDS2017 (2017 enterprise traffic) and CIC-IoT-2023 (IoT device traffic) are fundamentally different. This result is consistent with findings by Mondragon et al. [5], who documented similar cross-dataset failures.

However, the DQN agent’s inference speed (0.20  $\mu$ s per sample, 4.9M samples/sec) makes it suitable for high-throughput environments where real-time classification is required, even if it requires per-network retraining.

## 10.2 Comparison with Published Research

Table 7 compares this study’s results with published RL-IDS research.

Table 7: Comparison with Published RL-IDS Research

Study	Method	F1/Acc	Notes
Hsu & Matsuoka (2020) [11]	DRL	90% Acc	Binary, CIC-IDS2017
Suwannalai et al. (2020) [24]	Adversarial DQN	91% Acc	With adversarial training
Alavizadeh et al. (2022) [12]	DQN	84% F1	CIC-IDS2017
Ren et al. (2022) [16]	DDQN+FS	97.8% F1	Feature selection DRL
Sharma (2025) [38]	Rainbow DQN	99.8% Acc	7 DQN enhancements
<b>This study (Exp 7)</b>	<b>DQN</b>	<b>90.97% F1</b>	Binary, 78 features
<b>This study (ML)</b>	<b>XGBoost</b>	<b>99.87% F1</b>	Same dataset

The comparison reveals several insights:

1. **Competitive with early work:** The DQN agent’s 90.97% F1 exceeds or matches early published DQN results (Hsu & Matsuoka 2020, Alavizadeh et al. 2022), demonstrating competent implementation.
2. **Gap explained by enhancements:** Higher-performing systems (Rainbow DQN, DDQN+FS) employ additional techniques — prioritised replay, distributional estimation, feature selection — that were deliberately excluded to maintain a clear proof-of-concept demonstration.
3. **ML baselines match literature:** The XGBoost baseline’s 99.87% F1 aligns with published ML benchmarks on CIC-IDS2017 [5, 10], confirming evaluation validity.
4. **Unique contribution:** No comparable study systematically evaluates reward structure impact across 8 configurations on CIC-IDS2017, nor tests flow-level RL agents on zero-day web attacks, making these contributions novel.

### 10.3 Limitations

Several limitations constrain the generalisability of these findings:

1. **Offline Training on Static Data:** The RL agents were trained on labelled data with immediate feedback. While framed as reinforcement learning, the training process more closely resembles contextual bandits or offline RL, as the agent receives ground-truth rewards rather than delayed, partial, or noisy signals typical of production environments.
2. **Binary Action Space:** The “Allow/Block” action space, while appropriate for proof-of-concept, does not capture the full range of responses available in real IDS deployments (rate limiting, quarantine, escalation, honeypot redirection).
3. **Single Environment Configuration:** The IDS environment used a fixed episode structure (random start, 1,000 steps, independent samples) throughout all experiments. Alternative designs — sliding window observations, sequential episode ordering, or feature subset selection — were not evaluated but could yield different results.
4. **Dataset Limitations:** CIC-IDS2017 represents 2017 network conditions. Real-world implementations would face concept drift as attack patterns evolve. The cross-dataset experiment (Scenario 4) confirmed this limitation.
5. **Feature Representation:** The flow-level feature set cannot capture application-layer attacks, as demonstrated by Scenario 3. This is a fundamental limitation of network flow analysis, not specific to the RL approach.

### 10.4 Justification of the RL Framework

A valid question is whether the RL framing is necessary. We argue it provides distinct advantages:

1. **Interpretable Risk Tuning:** The 8-experiment reward analysis demonstrates that RL provides a mechanism to encode organisational risk preferences directly into detection policy. The recall range of 84.85%–98.47% via reward adjustment alone is more interpretable than adjusting class weights or decision thresholds in supervised learning.

2. **Foundation for Online Learning:** While this study used offline training, the RL framework provides a natural extension path to online learning, where agents could update from analyst feedback, honeypot data, or red-team exercises.
3. **Inference Speed:** The DQN agent’s 0.20  $\mu$ s inference time (4.9M samples/sec) makes it suitable for real-time high-throughput environments.

## 10.5 Future Research Directions

Based on the findings and limitations identified, several directions for future work are proposed:

- **Environment Variations:** Testing alternative environment designs including sliding window observations (temporal context), sequential episode ordering, and feature subset selection could reveal whether environment structure affects agent performance as significantly as reward structure.
- **Advanced DQN Variants:** Implementing Rainbow DQN enhancements (prioritised replay, dueling networks, noisy exploration) could close the gap with supervised learning baselines.
- **Multi-Modal IDS:** Combining flow-level RL with deep packet inspection could address the application-layer detection gap identified in Scenario 3.
- **Extended PPO Training:** Increasing PPO training beyond 1,000,000 timesteps and investigating policy gradient methods better suited to discrete action spaces.
- **Continual Learning:** Addressing concept drift through techniques like Elastic Weight Consolidation [19] or progressive network training.
- **CybORG++ Integration:** Migrating the environment to the CybORG++ framework [8] for proactive defence scenarios beyond detection.
- **Adversarial Robustness:** Evaluating agent behaviour under adversarial evasion attacks [31] to assess deployment readiness.

## 11 Conclusion

This study investigated the viability of Reinforcement Learning agents for autonomous network intrusion detection, comparing a custom DQN implementation and PPO agent against Random Forest and XGBoost baselines across four evaluation scenarios on the CIC-IDS2017 and CIC-IoT-2023 datasets.

### 11.1 Summary of Contributions

The key contributions of this work are:

1. **Reward Engineering as Policy Control:** Eight systematic DQN experiments demonstrated that reward structure is the primary determinant of agent behaviour. Recall was tuneable from 84.85% to 98.47% via reward ratios alone, enabling direct encoding of organisational risk preferences into detection policy.

2. **Custom RL Environment:** A Gymnasium-compatible IDS environment was implemented, supporting configurable reward structures and zero-day simulation through attack category exclusion.
3. **Four-Scenario Evaluation:** A comprehensive evaluation across standard classification, two zero-day simulations, and cross-dataset generalisation — providing an honest assessment of both capabilities and limitations.
4. **Boundary Condition Identification:** The contrasting zero-day results (56.77% DDoS recall vs. 2.29% web attack recall) identify precisely where flow-level RL succeeds (volumetric anomalies) and fails (application-layer attacks).

## 11.2 Key Findings

- **Standard Classification:** ML baselines achieved near-perfect accuracy (XGBoost: 99.87% F1), while the best DQN configuration achieved 90.97% F1 — competitive with early published RL-IDS results.
- **Zero-Day DDoS:** The DQN agent detected 56.77% of DDoS attacks never seen during training with 100% precision, demonstrating that reward-driven learning develops partially transferable anomaly detection capabilities.
- **Zero-Day Web Attacks:** Near-total failure (2.29% recall) confirmed that flow-level features cannot capture application-layer attack semantics.
- **Cross-Dataset:** Universal failure (all models  $\leq 2.16\%$  F1) confirmed dataset distribution shift as an unsolved challenge for IDS generalisation.
- **Inference Speed:** The DQN agent achieved  $0.20 \mu\text{s}$  per sample (4.9M samples/sec), making it suitable for high-throughput real-time environments.

## 11.3 Limitations

The primary limitations include: (1) offline training on static labelled data, which does not fully exploit RL’s potential for online adaptation; (2) a binary action space that oversimplifies real-world response options; (3) feature representation restricted to flow-level statistics, precluding application-layer attack detection; and (4) evaluation on benchmark datasets that may not reflect contemporary network conditions.

## 11.4 Future Work

Promising research directions include: environment design experiments (sliding window observations, sequential episodes, feature subset selection), advanced DQN variants (Rainbow DQN), multi-modal IDS combining flow-level and payload analysis, continual learning for concept drift adaptation [19], and deployment in network simulation environments such as CybORG++ [8].

## 11.5 Concluding Remarks

Reinforcement Learning offers a compelling paradigm for autonomous network defence, not as a replacement for traditional machine learning, but as a complementary approach that enables security practitioners to encode complex, context-dependent risk preferences directly into detection systems. The reward-shapeable nature of RL agents provides a control mechanism absent from supervised learning, while the sub-microsecond inference time ensures real-time viability. While the current proof-of-concept demonstrates clear performance gaps relative to ML baselines on known threats, the zero-day DDoS result and the systematic reward engineering analysis contribute novel findings to the RL-IDS literature and provide a foundation for future adaptive defence systems.

## References

- [1] A. L. Buczak and E. Guven, “A survey of data mining and machine learning methods for cyber security intrusion detection,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.
- [2] IBM Security, “Cost of a data breach report 2025.” <https://www.ibm.com/reports/data-breach>, 2025. Accessed: January 2026.
- [3] K. Alam, M. Monir, M. Hossain, M. Uddin, and M. Habib, “Adaptive defense: Zero-day attack detection in nids with deep reinforcement learning,” *IEEE Access*, vol. 13, pp. 1–15, 2025.
- [4] V. Shanmugam, R. Razavi-Far, and E. Hallaji, “Addressing class imbalance in intrusion detection: A comprehensive evaluation,” *Electronics*, vol. 14, no. 1, p. 69, 2025.
- [5] J. Mondragon Guadarrama, P. Branco, G. Jourdan, A. Gutierrez-Rodriguez, and R. Biswal, “Advanced ids: a comparative study of datasets and machine learning algorithms for network flow-based intrusion detection systems,” *Applied Intelligence*, vol. 55, no. 7, 2025.
- [6] T. T. Nguyen and V. J. Reddi, “Deep reinforcement learning for cyber security,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 6, pp. 2554–2569, 2021.
- [7] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*. The MIT Press, second ed., 2020.
- [8] H. Emerson, L. Bates, C. Hicks, and V. Mavroudis, “Cyborg++: An enhanced gym for the development of autonomous cyber agents,” *arXiv preprint arXiv:2410.16324*, 2024.
- [9] Y. Wang *et al.*, “Research on traditional and deep learning strategies based on optical flow estimation - a review,” *Journal of King Saud University - Computer and Information Sciences*, vol. 36, no. 4, p. 102029, 2024.
- [10] Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mustafa, and K. A. Kasmiran, “Benchmarking of machine learning for anomaly-based intrusion detection systems in the cids2017 dataset,” *IEEE Access*, vol. 9, pp. 22351–22370, 2021.
- [11] Y.-F. Hsu and M. Matsuoka, “A deep reinforcement learning approach for anomaly network intrusion detection system,” in *2020 IEEE 9th International Conference on Cloud Networking (CloudNet)*, pp. 1–6, IEEE, 2020.
- [12] H. Alavizadeh *et al.*, “Deep q-learning based reinforcement learning approach for network intrusion detection,” *Computers & Security*, vol. 118, p. 102752, 2022.
- [13] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.
- [14] S. Liang, H. Feng, and C. He, “Proximal policy optimization-based intrusion detection model for dynamic adaptation,” in *2024 IEEE ISCEIC*, pp. 1–6, IEEE, 2024.

- [15] E. Bates, V. Mavroudis, and C. Hicks, “Reward shaping for happier autonomous cyber security agents,” in *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security*, pp. 89–100, ACM, 2023.
- [16] K. Ren, Y. Zeng, Z. Cao, and Y. Zhang, “Id-rdrl: a deep reinforcement learning-based feature selection intrusion detection model,” *Scientific Reports*, vol. 12, no. 1, p. 15370, 2022.
- [17] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, “A survey of network-based intrusion detection data sets,” *Computers & Security*, vol. 86, pp. 147–167, 2019.
- [18] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)*, pp. 108–116, SciTePress, 2018.
- [19] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, *et al.*, “Overcoming catastrophic forgetting in neural networks,” *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.
- [20] K. Khetarpal, M. Riemer, I. Rish, and D. Precup, “A survey of continual learning for reinforcement learning,” *arXiv preprint arXiv:2010.14481*, 2020.
- [21] C. Pan *et al.*, “A survey of continual reinforcement learning,” *arXiv preprint arXiv:2506.21872*, 2025.
- [22] W. Yang, A. Acuto, Y. Zhou, and D. Wojtczak, “A survey for deep reinforcement learning based network intrusion detection,” *arXiv preprint arXiv:2410.07612*, 2024.
- [23] M. Malik and K. Singh Saini, “Network intrusion detection system using reinforcement learning,” in *2023 4th International Conference for Emerging Technology (INCET)*, pp. 1–4, IEEE, 2023.
- [24] E. Suwannalai and C. Polprasert, “Network intrusion detection systems using adversarial reinforcement learning with deep q-network,” in *2020 18th International Conference on ICT and Knowledge Engineering (ICT&KE)*, pp. 1–7, IEEE, 2020.
- [25] A. Tellache *et al.*, “Multi-agent reinforcement learning-based network intrusion detection system,” in *NOMS 2024-2024 IEEE Network Operations and Management Symposium*, pp. 1–9, IEEE, 2024.
- [26] T. Dudman and M. Bull, “Towards a generalisable cyber defence agent for real-world computer networks,” *arXiv preprint arXiv:2511.09114*, 2025.
- [27] M. Foley, C. Hicks, H. Kate, and V. Mavroudis, “Autonomous network defence using reinforcement learning,” in *Proceedings of the 2022 ACM Asia Conference on Computer and Communications Security*, pp. 1–15, ACM, 2022.
- [28] D. Fathi, A. Attiah, A. Hakeem, and A. Cherif, “Reinforcement learning for intrusion detection: Recent advances and datasets,” *EasyChair Preprint*, no. 15665, 2025.

- [29] A. A. Syairozi and Arizal, “Challenges and limitations of ids: A comprehensive assessment and future perspectives,” in *Proceedings of the 1st International Conference on Research and Innovations in Information and Engineering Technology (RITECH 2025)*, SciTePress, 2025.
- [30] Mambwe *et al.*, “Limitations of machine learning in network intrusion detection systems,” *Digitus*, 2024.
- [31] M. A. Merzouk *et al.*, “Susceptibility of ml-nids to adversarial settings,” *IEEE Access*, 2025.
- [32] S. Jamshidi *et al.*, “Application of deep reinforcement learning for intrusion detection in internet of things: A systematic review,” *Internet of Things*, vol. 31, no. 8, 2025.
- [33] Miles-Farmer *et al.*, “Reinforcement learning for autonomous resilient cyber defence (arcd),” tech. rep., Frazer-Nash Consultancy / Dstl, 2024.
- [34] T. Kunz *et al.*, “A multiagent cyberbattlesim for rl cyber operation agents,” in *International Conference on Computational Science*, 2022.
- [35] J. Claypoole *et al.*, “Training rl agents for multi-objective network defense tasks,” *arXiv preprint*, 2025.
- [36] 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.
- [37] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-baselines3: Reliable reinforcement learning implementations,” *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021.
- [38] V. Sharma, “Rainbow dqn for intrusion detection: A unified deep reinforcement learning approach across benchmark datasets,” *Article*, 2025.