National College of Ireland

An Active Defense Security Framework using Deep Reinforcement Learning for Container-Based Architectures

MSc Research Project
MSc Cloud Computing

Mahesh Kumar Avula
X23196530

School of Computing
National College of Ireland

Supervisor: Sean Heeney

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Mahesh Kumar Avula

Student ID: x23196530

Programme: MSc Cloud Computing **Year:** 2024 - 2025

Module: MSc Research Project

Supervisor: Sean Heeney

Submission Due Date: 13/12/2024

Project Title: An Active Defense Security Framework using Deep Reinforcement Learning for Container-Based Architectures

Word Count: 7900 **Page Count**: **20**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Mahesh Kumar

Date: 13/12/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

An Active Defense Security Framework using Deep Reinforcement Learning for Container-Based Architectures

Mahesh Kumar Avula
X23196530

Abstract

In this research, we present an active defense security framework for the container environments for overcoming the dynamic security challenges of containerized applications. We employ deep reinforcement learning (DRL) to optimize adaptive threat response and resource utilization within the framework. It presents a dynamic Holistic System Attack Graph (HSAG) model of time varying behavioral analysis of container activities, along with a novel Prioritized Dueling Double Deep Q Network (P3DQN) for security optimization at its foundation. A comprehensive security evaluation system is designed in the framework that is capable of real time monitoring, analysis and automatic response. Detection rates reported are 98.5% for CPU attacks and 96.8% for memory incident with 0.8% false positive rate. During normal operation, average CPU usage never exceeded 3.19%, and peak usage during active incident response was 25.04%. For detected incidents, the framework achieved an average of 17.42 seconds from detection to completion of the action (17.42s onset to action complete with a prevention rate of 100 per cent. This shows the effectiveness of the framework to provide real time security monitoring and response with optimal resource utilization in containerized service environments.

Keywords −Container Security, Deep Reinforcement Learning, HSAG, Cloud Computing

1 Introduction

1.1 Motivation and Problem Background

The dynamic nature of microservices and the complex containerized environment makes deployment and security in containerized applications pretty vulnerable. With the proliferation of container technologies, application deployment has been fundamentally changed, yet new security vulnerabilities and new attack vectors have been introduced (Sethuraman and Khan, 2023). Therefore, traditional security approaches are not up to the task, as they are static in nature and unprepared to evolve to meet new threat landscapes, an issue which container-based environments are plagued with (Rahman et al., 2023). Continuous deployment patterns find more sophisticated security measures in a dynamic environment introduced by container orchestration (Sofia et al., 2023).

Research (Aktolga, 2023) recently shows that containerized applications are particularly vulnerable to resource-based incidents and attacks—the attack vectors of compromised

containers could impact the entire system stability and security. The challenges posed by ephemeral containers as well as the need for keeping service availability while securing these services compound further.

Deep reinforcement learning (DRL) has demonstrated its capability to adapt to changing threat landscapes and optimize security tactical decisions in real time (Nguyen and Reddi, 2021) in the application of DRL to dynamic security optimization. Though, current DRL based solutions mostly miss being aware of container specific environments and particular security requirements (Nkenyereye et al., 2023). To overcome these limitations, this research provides an enhanced framework that combines a P3DQN architecture with a comprehensive container monitoring to construct an adaptive security system.

1.2 Problem Statement

In this research, the spread of container use in enterprise environments has made common container specific security incidents (Adhikari and Baidya, 2024). Due to the unique, highly dynamic containerized environment, traditional security solutions frequently do not fulfill their role of sounding the alarm in regard to dynamic threat response and runtime security. An adaptive security framework that is able to dynamically respond to emerging threats with minimal overhead and with the optimal application performance as (Li et al., 2022) proposed. Traditional reactive security methods do not catch the dynamic nature of container worlds. By combining deep reinforcement learning with whole container monitoring, we can create a more proactive and resource efficient security solution that can adapt to change threat landscapes while best utilizing resources.

1.3 Research Question

With the proposed adaptive security framework that integrates Holistic System Attack Graph (HSAG) and P3DQN-based deep reinforcement learning, what positive impacts can be gained when aiming to improve defense efficiency, container security and resource optimization in containerized environments?

1.4 Research Objective

This research aims to enhance the security framework of active defense for container-based environments using deep reinforcement learning. With the use of P3DQN, dynamic HSAG model and the dynamic model of the attack, the proposed framework enables adaptive threat response without losing security standards and maximizing resource utilization.

1.5 Research Contributions

- A dynamic HSAG model is developed that integrates real time container activity analysis with attack patterns to provide a holistic view of potential security threats.
- Specifically, we implement a P3DQN algorithm that incorporates prioritized experience replay and dueling network architecture for consistent security policy enforcement.
- A comprehensive monitoring and evaluation system created for real time security assessment and response.
- Resource aware security optimization achieved by implementation of intelligent resource allocation mechanisms, and dynamic scaling algorithms to maintain a trade off between security coverage and resource utilization.
- An automated testing framework for systematic evaluation of security effectiveness and performance impact.

1.6 Thesis Structure

In Chapter 1, we introduce the research problem and motivation of applying deep reinforcement learning to the field of container security, objectives and key contributions. In Chapter 2, we provide a comprehensive literature review of container security challenges, existing DRL application in cybersecurity, existing security frameworks, and last editorial research gaps. The research methodology is detailed in Chapter 3, describing the development of the HSAG model and P3DQN architecture. The technical requirements and system architecture, along with the component interactions are described in Chapter 4, as design specifications. In Chapter 5, implementation is covered, discussing development environment, component implementation, and testing frame setup. In Chapter 6, these results are evaluated; analyzed are detection accuracy, response times, and efficiency in terms of resources used. Chapter 7 then concludes the thesis with key findings, limitations, and possible future research directions regarding the use of deep reinforcement learning in container security.

2 Related Work

2.1 Container Security and Runtime Protection

In cloud computing environments, (Brady et al., 2020) propose a holistic method for securing Docker container security. In particular, they present a series of multi layered continuous integration and continuous deployment (CI/CD) pipeline that ensures the security of Docker images along the development lifecycle. This work addresses the growing concerns with container image vulnerabilities, as containers become more and more critical for the migration of DevOps processes to the cloud. To evaluate the security of Docker image, the study proposes a multi staged CI/CD pipeline to secure against publishing and reuse of images that have been confirmed vulnerable. The pipeline is geared to be flexible such that developers can plug in preferred security analysis tools and set security analysis policy as relevant to their development environments. The study, despite its limitations, is highly relevant to the study of container security, as containerization becomes more and more popular. In (Li et al., 2021), a security risk assessment method of edge computing container based on dynamic game theory is presented. To analyze complex interactions between container intrusion detection systems and malicious attackers, the authors build an Edge Computing Container Risk Assessment Dynamic Game Model (ECC-RA-DGM). Security risks are quantified by the model that takes into account things such as attack probabilities, defense strategies, and system vulnerabilities. The strength of the study is its novel use of dynamic game theory in edge computing security to better represent attacker defender interactions. The model is, however, quite complex, which might make it infeasible to implement in practice, particularly, for resource constrained edge devices. Both the dynamic game theoretic approach and the security strategies for edge computing provide a promising future direction of research.

In their work (Jin et al., 2021) introduce DSEOM, a framework for dynamic evaluation and optimization of Moving Target Defense (MTD) in container-based cloud environments. The authors tackle the problem of the effectiveness drift in defense mechanisms when exposed to highly dynamic cloud environments, specifically for MTD techniques. MTD evaluations are done rapidly with DSEOM, which automatically perceives and respond to updates within the environment of the cloud. This research proposes multiple dimensions model attack graphs, holistic attack graph (HAG), and security and performance metrics definition. The framework can monitor update events, update the effectiveness of MTD, and update strategies using a formulation of a knapsack problem. The paper is limited in that they only discuss certain types of update events, and their work does not address the actual overhead of continuous monitoring and optimization. This research has a direct application to cloud security with the proliferation

of container-based environments and microservices. The study of security attacks in Docker container communications presented by (Lee et al., 2023) is a study of security attacks in Docker container communications. Through practical experiments focusing on ARP spoofing, DDoS and privilege escalation attacks we demonstrate the vulnerabilities of Docker networking. In this study, an overview of different methods of communication in Docker containers and discuss security challenges in container networks was presented. Some of the strengths of the research are in their practical approach; they provide detailed experimental setups and result for all attack scenarios considered. It does not, however, propose new defense mechanisms or solutions for mitigating these vulnerabilities. Finally, the experiments are conducted in a controlled environment that likely doesn't fully reflect the complexities of large-scale production Docker deployments.

Moric et al., (2024) investigate security frameworks for container orchestration and microservices, specifically security mechanisms are needed in these technologies. The study focuses on container security in all microservice and container deployment lifecycle stages using virtual configurations, Grype, and Anchore. It assesses the security tool performance and evaluates the tradeoff between the security and cost in containerized settings. Employing a comprehensive security architecture, the authors bring up the important issues of data encryption, network security, and access control. This practical study suggests actual application in the security measures of containerized systems, which are integrated into the cycle of software development. Unfortunately, there are caveats to this study such as deep analysis and more quantitative data. In a comprehensive analysis of Docker container attacks and defense mechanisms, (Haq et al., 2024) categorize Docker container attacks and defense mechanisms into nine types and propose a detailed taxonomy. It points out that container-based applications are becoming more and more popular, and that we need strong defense strategies. Building on an extensive dataset of 51 real world vulnerabilities, the authors reevaluate current state of the art anomaly detection techniques, as well as providing value in understanding real world applications of current defense mechanisms. In addition, they point to deficiencies in current defenses, such as high false positive rates and insufficient training data that render existing approaches to anomaly detection impractical. While its limitations are its main limitations, it serves as extremely relevant research on container security and points out the dire necessity for additional research in order to improve container application security. Empirical evaluation as well as a systematic approach make the study a very useful resource for container security researchers and practitioners.

2.2 Machine Learning Approaches in Security Applications

In this work, the authors of (Li et al., 2021) present an optimal defensive deception framework in container-based cloud environments based on deep reinforcement learning (DRL) that can neutralize stealthy attackers. To address both time and range dimensions, the authors propose a System Risk Graph, SRG, a set of stimuli and associated risks and vulnerabilities for cloud application vulnerability and risk analysis, which is also a very precise adversarial model. Quantitatively, they define a coefficient of deception (C) to evaluate the effectiveness of decoy placement strategies. The framework's most salient strengths are its ability to respond to the dynamic characteristics of containerized clouds as well as the discipline that it brings to threat modeling. Compared to current approaches simulations show a 30.22% increase in coefficients of deception as well as improved detection ratios for random-walker (30.69%) and persistent (51.10%) attackers. This research is highly relevant to the field of cloud security as container-based architectures and microservices start to take on high prevalence in the industry. (Kommula et al., 2023) paper looks into the security challenges of container technology when used in cloud computing environments. The authors show that existing implementations are adequate at preventing data leakage in shared memory and though these implementations are

already secure from data leakage they flag security vulnerabilities in container networking especially in Docker Compose clusters where inter container communication can be intercepted. To remedy the weakness of these vulnerabilities, the authors suggest an apparatus, named ldocker0, that enhances security by learning and enforcing network policies, thereby preventing non-authorized access or malicious attack, such as through ARP spoofing. The limitations of the paper include not having empirical evidence of effectiveness of the proposed solution and lack of performance impact analysis. Nonetheless, the study provides insights into container security research by identifying key problems and offering novel solutions to these problems.

SecCPS, secure container placement strategy based on deep reinforcement learning was developed by (Deng et al., 20202). Co-resident attacks are a problem that is addressed in the strategy in containerized environments where container may share a VM or physical server. Finally, the research looks at the more complex case of containers running on VMs that reside on physical machines, which is a more realistic view of the real-world cloud setup. The container placement optimization is achieved within the deep reinforcement learning model using risk minimization by balancing workloads across tenants. The results of security constraints placement experiments show significant superior of SecCPS compared to well-known strategies such as Spread, Binpack, Random, and PSSF in terms of minimizing co-resident risks and workload balancing. The study has limitations, however, in the assumption of uniform resource requirements and fixed capacity limits for VMs, phenomena which may not exist in reality. In (Tunde-Onadele et al., 2024), the authors introduce a self-supervised hybrid learning (SHIL) framework to automatically detect security attacks in containerized applications. The framework employs unsupervised and supervised machine learning techniques to discover attacks without dependence on manual, labeled training data. It consists of three components: Autoencoder neural networks for unsupervised anomaly detection, hybrid alert validation for filtering false alarms, and self-supervised model creation for automatic labeling of the data. Its strength is in the innovative solution it adopts to acquire labelled training data in dynamic container environments. It significantly reduced false alarms relative to existing methods, on 46 real world security attacks on 29 server applications. Because SHIL is so light weight, it's practical for large scale deployments. But the study also has limitations, including not discussing how varying types of attacks will affect it, or how adept the study is at identifying advanced threats.

In their work (Aly et al., 2024), the researchers study multi-class threat detection in Kubernetes environments using neural network and machine learning techniques. Based on this, the authors introduce an advanced detection method that combines the Naive Bayes algorithm with the feature engineering and dimensionality reduction techniques like Principal Component Analysis (PCA) and Autoencoders. This study fills a large gap in Kubernetes security by moving beyond binary classification of normal vs abnormal activities. The F1-Score finally achieved using their approach of combining PCA and Autoencoders with Naive Bayes classifier, was 0.95 and an accuracy of 91%. The relevance of this study to Kubernetes security is profound as it provides an alternative approach towards handling complex security issues within container environments. And being a multi-class classification and to tackle more sophisticated pre-processing techniques, this study established a new benchmark for threat detection in Kubernetes infrastructures. In Li et al., (2023), they present an optimal active defensive security framework (OADSF) for container-based cloud environments using DRL. The authors address challenges of complex attack scenarios and evolving microservice states in container-based clouds, which can weaken active defense strategies. Using a Holistic System Attack Graph (HSAG) model of security threats and attack paths at both application and container layers, they build a comprehensive threat model and an adaptive active defense deployment strategy. The optimization of moving target defense (MTD) strategy is modeled as

a Markov decision process, which reduces to concrete optimization problems by discarding a large class of feasible solutions in large scale cloud applications with high state space explosion. Finally, the proposed framework provides defense efficiency (versus both existing defense methods including DSEOM and SmartSCR) under the same set of plausible assumptions.

2.3 Cloud-Native Security Architectures

Proposed in (Di Stefano et al., 2020), Ananke is a framework to monitor and orchestrate cloud native applications (CNAs) following a microservices architecture.). To profile microservices and their interactions in a platform as a service environment, the authors propose a time-varying multi-layer graph model. The goal of the model is to aid the optimization strategies in which cloud providers can increase service quality and service level agreements. The main strength of this study is that in modeling CNAs both cluster and application layers are taken into account. The proposed multi-layer network model covers the essence of how microservices interact and deployment on physical resources, which enables fine granularity analysis of application performance and resource consumption. The limitations to the paper include no empirical validation, without ongoing practical implementation and lack of comparative analysis with existing monitoring and orchestration solutions. In (Bhowmik et al., 2020), a framework was proposed for building security into container based on premises cloud orchestration. They present a model that quantifies container image security risk prior to deployment to reduce cloud infrastructure security risk. The paper talks about the state of cloud orchestration and container technologies, noting that cloud orchestration and container technologies need to be better secured because of the increasing attacks. The researchers then develop an experimental model that leverages image scanning tools to analyze container images in terms of Common Vulnerability and Exposures (CVE) metrics, Common Platform Enumeration (CPE) metrics, and Common Vulnerability Scoring System. A quantitative risk assessment method is created by weighting different type of issues that are identified during the scanning. These scores are then used by the framework to select the most secure image for deployment. The authors admit their study has limitations, however: a small sample size and a lack of discussion of possible countermeasures to vulnerabilities discovered.

The paper by (Nascimento et al., 2024) focuses on the application of availability, scalability, and security in transition from container to cloud native applications. They explain in detail the drawbacks of container-based application and how cloud native architectures help in overcoming them. To show this technical aspect of this migration, the proof of concept (PoC) implementation uses Kubernetes in Azure. The research methodology features how DevOps and DevSecOps approaches can be done while increasing the reproducibility and reliability. A complete analysis of served workload, and system health under different conditions is given to conclude that employing the cloud native approach was indeed effective for coping with increased workload load, therefore resulting in the tremendous performance of the PoC. While the study does have its limits, which include dependence on Azure and Kubernetes, more in depth analysis of security challenges and solutions in cloud native environment is necessary. Vaño et al., (2023) paper covers a comprehensive view of cloud native workload orchestration at the edge, combating deployment strategies, and possible trends. It studies the transfer of the principles of cloud-native computing to the edge computer, exploring the challenges and opportunities. The virtualization techniques, commercial approaches, container deployment option, and orchestration framework e.g Kubernetes are there it covers all of these. By integrating the academic literature in combination with insights from open-source development projects and use cases from the industry, this paper approaches the topic of edge computing in a practical manner and provides a more up to date view of this rapidly evolving field. Additionally, the authors examine commercial solutions from the major cloud providers, and

discuss their approaches and limitations. It reviews the trends in edge computing, such as WebAssembly modules and orchestrating the heterogeneous workloads. Though also limited itself, the paper serves as a valuable resource for other researchers or practitioners working on edge computing by providing a comprehensive coverage of current deployment option and its future.

To improve security and resource optimization in cloud environments, (Prasadu et al., 2024) propose layering container orchestrators. The architecture enables advanced security automation and AI driven resource management to address the challenges of managing containerized applications. The strength of the study includes a complete way of approaching container orchestration with a single system taking care of security, resource management and scalability. While it is surely far from a complete research effort in an ideal scenario, the research does have a very timely and relevant contribution to provide a nice foundation for future research in the area of cloud computing and container orchestration. (Joshi et al., 2024) present a cloud native system for optimizing and delivering cloud native Large Language Models (LLMs) to Kubernetes based infrastructure as a solution to the problem of deploying and operating LLMs in production. Using Kubernetes declarative resource management capability, the framework is easy to train and deploy LLMs on cloud providers such as KIND, Google Kubernetes Engine and Amazon Elastic Kubernetes service. The intent of this research is to improve the performance and cost effectiveness of LLM deployments using the scalability, flexibility, and resource allocation provided by Kubernetes. The paper presents a detailed discussion of challenges in deploying inference services in the cloud, referring to the complexity of hardware configurations required for the model serving containers. This part of the study is very appropriate for optimization of resource utilization and cost management using dynamic scaling in the presence of growing demand for the AI and ML applications.

2.4 Critical Analysis

The research gaps in container security, DRL applications, and cloud native security architectures are identified by doing the literature review. Models today do not consider vulnerabilities of containers and cloud native vectors of attack. To fill these gaps, the proposed HSAG model incorporates application and container layer threats and cloud particular security controls. This paper presents a more sophisticated multi agent approach to security optimization via the P3DQN algorithm. The work seeks to produce resource aware security solutions for containerized deployments that are adaptive to cloud native services. Building on the standard HSAG model, the enhanced HSAG model incorporates dynamic behavioral analysis and real time adaptation to overcome the state space explosion and real-time decision-making issues with existing DRL approaches for container environments.

3 Research Methodology

3.1 Overview

The research methodology proposes a systematic and iterative development of an active defense security framework for our container-based cloud environments. The methodology is based at its heart on theoretical modeling, practical implementation, and empirical evaluation, to address the research objectives defined. It is designed to utilize cloud native services in conjunction with deep reinforcement learning to build an adaptive security framework, able to adapt to emerging threats and optimize resource utilization in cloud native environments. The improvement of the Holistic System Attack Graph (HSAG) model represents the theoretical foundation. We build upon the work of (Li et al., 2022) and extend the base HSAG framework with cloud native attack vectors. A dynamic behavioral analysis model with real-

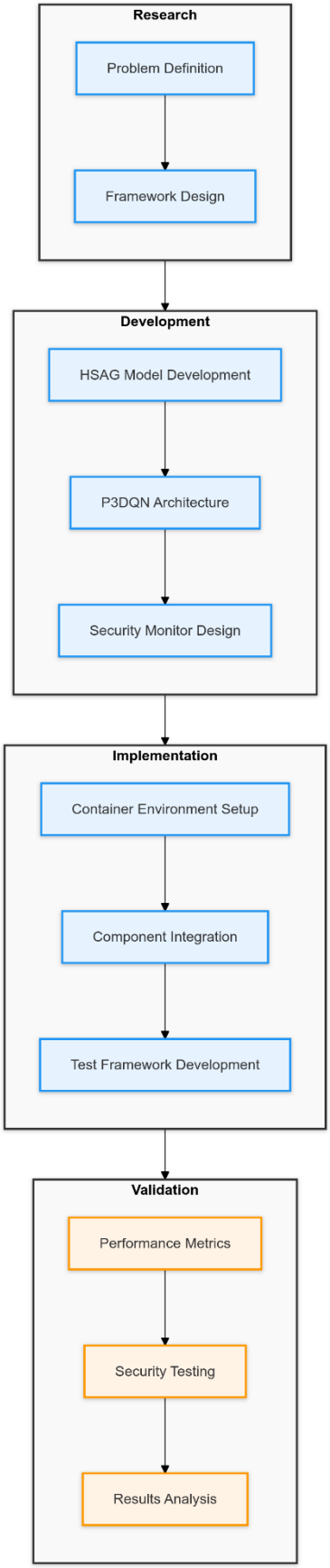

Figure 1: Proposed Research Methodology

time adaptation mechanisms is introduced into this enhanced model to enable monitoring and reacting to changes in container environments. Specifically, the design of the model exploits vulnerabilities and attack patterns that are specific to cloud native services, making it a more complete threat model.

3.2 Proposed Methodology

The development of an active defense security framework for container-based environments presented in Figure 1 follows a systematic and iterative research methodology. The methodology centers on three primary components: This thesis proposes a real time container monitoring system, a Holistic System Attack Graph (HSAG) model as security analysis model, a Prioritized Dueling Double Deep Q Network (P3DQN) for adaptive security decision making. To tackle the challenging problem of container security in dynamic environments, the research approach includes theoretical modeling for theoretical understanding of the problem, and practical implementation and empirical evaluation in dynamic container security. These dynamic environments often render traditional security approaches ineffective; namely, traditional reliance on static rule-based systems is insufficient. Since the P3DQN architecture is able to handle large state spaces and complex decision making better, deep reinforcement learning was adopted based on that fact.

The basic DQN algorithm is updated in the proposed P3DQN methodology through three improvements. Next, the dueling network architecture splits the estimation of state values and action advantages, facilitating improved policy evaluation in security domain. Second, the double Q learning component minimizes value overestimation bias (i.e., double Q learning prevents learning security specific values that are too high), which is critical for security decision impacts and outcomes to be accurate. Third, prioritized experience replay guarantees that forgotten security events won’t be ignored in the learning process. The sophisticated approach allows the framework to learn optimal security policies by exploring new security strategies and exploiting known effective responses.

3.3 Security Framework Implementation

First, we enhanced the HSAG model with the addition of container specific attack vectors and security patterns. The HSAG model gives an overall view of the security landscape as a graph of attack paths and their connection inside the containerized environment. This work presents a model that utilizes multiple auxiliary security metrics like resource utilization patterns, network behavior, and container lifecycle events to build up a dynamic security state representation for the system.

To take advantage of the difficult state space created by container environments, we designed the P3DQN agent's architecture to easily handle this. This involves multiple dimensions of container behavior, including how much CPU the container is using, how much memory the container is consuming, what are the network patterns, and some historical security events. The security response repertoire includes container isolation, resource throttling, and system restoration. We carefully crafted the reward function to balance its security effectiveness against the operation efficiency, both in terms of achieving the immediate threat mitigation and a stable long-term system.

The monitoring component was built to offer real time visibility into container behavior with minimal overhead. To achieve this, an effective data collection mechanism was implemented to gather such security metrics without causing significant performance impact on the container. An HSAG model and a P3DQN agent use telemetry data integrated with container runtime interfaces to analyze and make decisions.

3.4 Experimental Design and Validation

The experimental methodology leverages a full stack testing environment which emulates real world container deployments. It has multiple containerized application with different resource requirements and security characteristics in the test environment. The validation methodology consists of both a synthetic and realistic security scenario to test the framework performance. Different aspects of the framework's effectiveness was carefully defined and chosen to have performance metrics such as detection accuracy, false positive rates, response time, resource efficiency. Additionally, we highlighted specific metrics in evaluating the P3DQN agent's learning progress, including convergence rate, policy stability, and adaptivity to new security patterns.

The focus of the evaluation process is a continuous monitoring approach in which security incidents occur at different periods and degrees. This enables appraisal of the system's learning capabilities over time, as well as immediate responses to security. Long term performance trends and policy evolution patterns are used to gauge the framework’s ability to adapt to new threats and optimize its security policies.

4 Design Specifications

4.1 Architecture

The active defense security framework for container environment design presented in Figure 2 includes a highly complex architecture, which involves real time monitoring, security analysis and intelligent response mechanisms. In essence, the framework approaches the question of security in multiple layers, with each layer accomplishing the specific function of the layer in the security framework. The main layers are the container monitoring system, the HSAG analyzer, and the P3DQN based decision engine that act together to offer pervasive security coverage.

The first layer of the framework is the container monitoring layer which provides the foundation of the framework in that it implements efficient data collection mechanisms to produce the important metrics from containerized applications. This is a layer of low overhead but high visibility of container behavior. These systems directly interface with the container runtime and collect detailed telemetry: CPU utilization, memory consumption, network patterns, etc, as well as system calls. They are then processed, aggregated and provide a comprehensive view of state of the container environment.

The architecture of the framework is based on a modular design pattern, in which components can be independently scaled and updated without disruption to system cohesion. They communicate through well defined interfaces allowing loose coupling, while at the same time retaining robust data flow among the system components. In addition, this modular approach allows new security features to be incorporated, and existing capabilities can be enhanced, without major architectural changes.

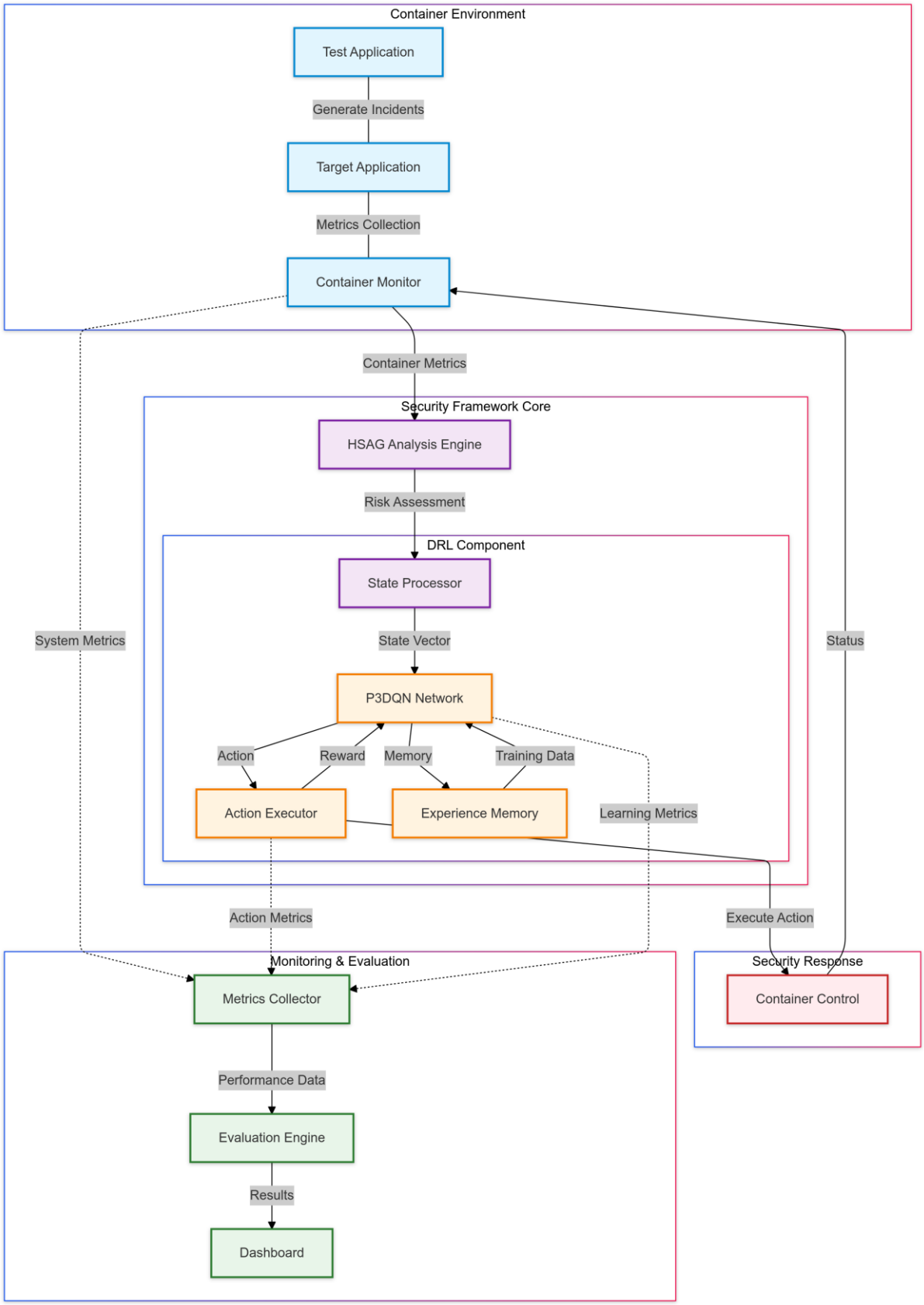

Figure 2: The Proposed Active Security Mesh Framework Architecture

4.2 Component Design

The HSAG analyzer component is intended to deal with processed metrics and create a holistic description of security state. This piece accomplishes the task of implementing complex pattern recognition algorithms to recognize unusual container behaviour that can be potential security threat. It consists of an analyzer that keeps a dynamically formed graph structure representing relationships amongst security events and there possible impacts on the system. The real time view of the current security landscape is given in this graph, which is updates continuously with new observations and security events.

Several novel design features endow the P3DQN agent with improved decision-making capabilities. The architecture of the agent's neural network contains dueling streams that each respectively estimate action advantages and state value. By providing separation, the agent is better able to assess the significance of different security states as well as prospects of effect resulting from possible actions. The prioritized experience replay mechanism guarantees significant security events to get the necessary attention in learning process and, thus, allow the agent to perform better in critical security situations.

4.3 Interface Specifications

A full set of interfaces are implemented in the framework to help various components communicate and talk to each other. They define standardized methods for collecting container metrics, and for reporting security events, through the monitoring interface. This guarantees that we collect consistent data across containers running at different container runtimes and environments. Analysis interface offers techniques for the processing of the security events and the derivation of the risk assessments whereas the response interface characterises the execution of the security actions and their resultant effect on the state of information security.

The visualization interface of the system provides real time insight into the security state of the container environment. Efficient data aggregation and visualization techniques are implemented into this interface to assist security administrators in the monitoring of system behavior and the tracking of security incidents. The security state is presented by the component dashboard in multiple views: real time metrics, historical trends, and incident reports.

4.4 Security Tests, Responses and Validation

The design of mechanisms within the framework is targeted in order to offer graduated security responses depending on the nature and severity of presented threats. There are mechanisms that implement different security actions, for example, container isolation, resource throttling and system restoration. Specifically, each response mechanism is purposely designed to provide minimal impact to legitimate container operation, while nevertheless being able to effectively counter security threats.

The feedback of the response system measures the effectiveness of the security actions adopted and provides the data for the P3DQN agent to learn how to choose the actions. These mechanisms measure response time, resource impact and threat mitigation effectiveness. It allows the system to constantly adjust security response and learn from these threats in real time.

The framework contains complete testing specifications that prescribe ways to prove system performance and security effectiveness. These specifications establish procedures for test scenario generation, system response measurement and security outcome evaluation.

Automated test sequences are tested employing the testing framework—varying security incidents are simulated, and the detection and response capacity of the system is measured.

The validation specifications dictate which metrics are to be considered to measure the system performance in diverse measurable aspects, i.e. detection accuracy, response time, and resource utilization. They also describe ways to measure the learning progress of the P3DQN agent and the effectiveness of its security decisions.

5 Implementation

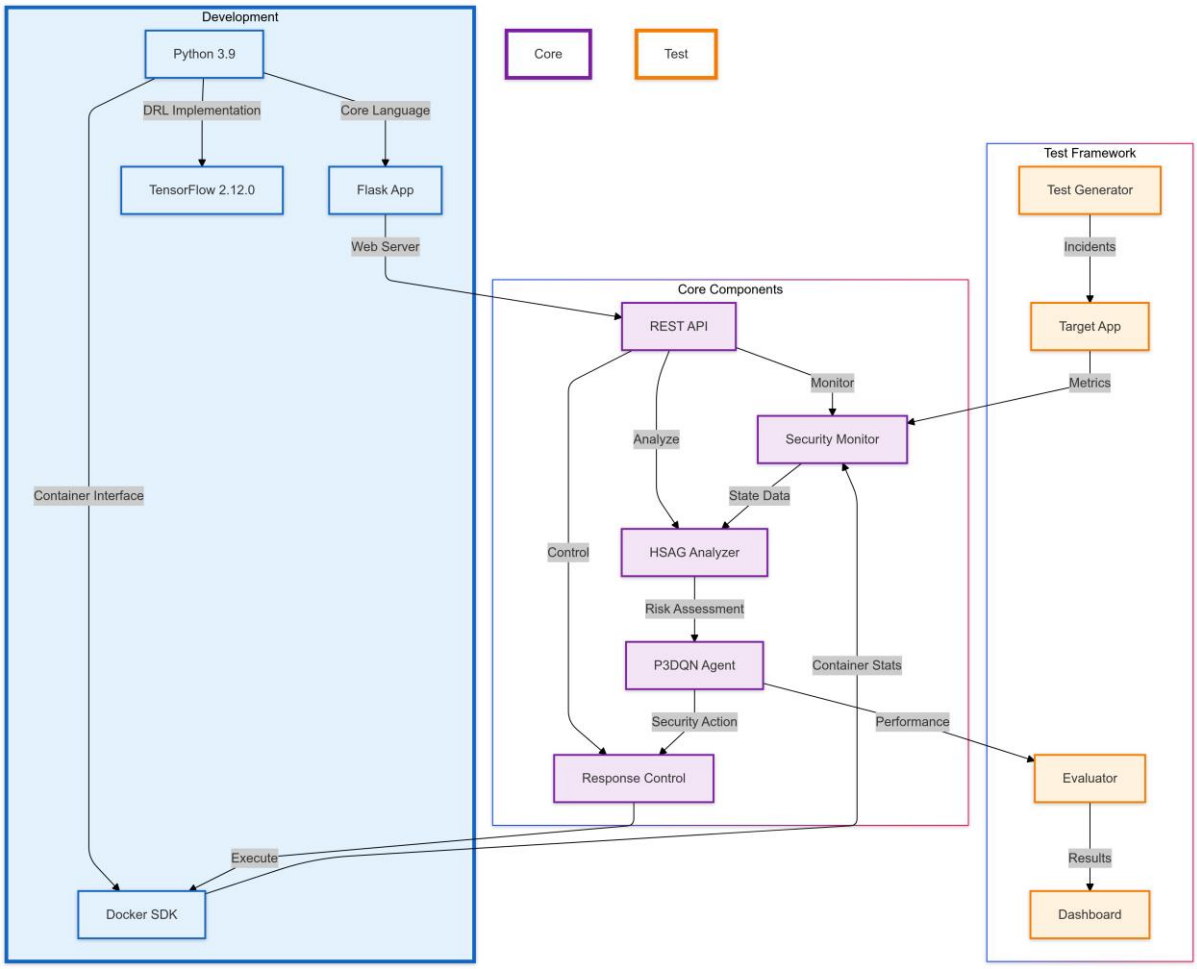

Figure 3: Active Defense Security Framework Implementation

5.1 Development Environment

Implementing the active defense security framework is itself a major technical undertaking with multiple related components. Python 3.9 along with all its machine learning and container library are used to implement the framework (Refer Figure 3). For development, Docker is used for container orchestration to make sure that the development behaviour is consistent across various deployment scenarios. The DRL components are implemented using TensorFlow 2.12.0 and employing P3DQN architecture.

The monitoring and control interfaces rely on a Flask based web application that provides RESTful APIs to allow the system to operate via and from which metrics are pulled. The development of components is applied in modular way, so each of them is implemented as separate module in Python, in order to keep code structured and testable. To carry out the implementation a microservices architecture is used in which the different components can be independently deployed and scale, improving flexibility and maintainability of the system.

This implementation of the monitoring component interfaces directly with the Docker daemon using the Docker SDK for real time collection of container metrics and state information. This implementation enables very efficient data collection with little overhead, but at the same time provides complete visibility into container behaviour. The system has efficient data buffering and processing mechanisms to handle metric collection of high volume without affecting container performance.

5.2 Component Flow Implementation

The flow of the security framework implementation is structured to have a series of interactions between the core components the Security Monitor component, HSAG Analyzer, P3DQN Agent, and Evaluator as shown in Figure 4. The Docker SDK is used by the Security Monitor to periodically collect container metrics such as CPU usage tracking, memory consumption patterns and container state information. These metrics are processed by the HSAG Analyzer in a sophisticated pipeline which essentially comprises data normalization, feature extraction and risk assessment calculations.

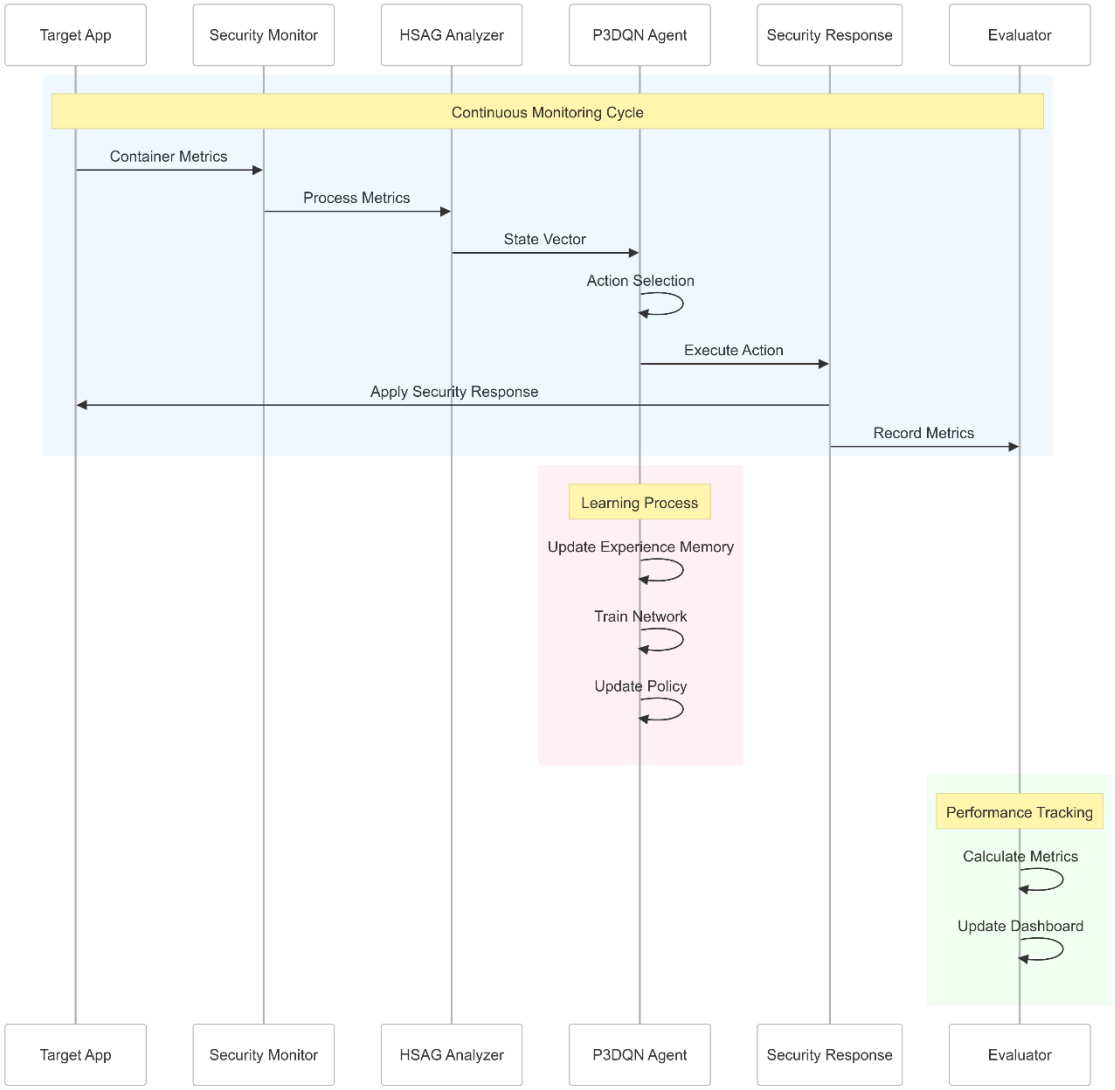

Figure 4: Core Components Implementation Workflow

The P3DQN Agent is the decision-making core of the framework, which introduces the state vector produced by the HSAG Analyzer at its neural network architecture realized with TensorFlow. Selected actions taken by the Security Response component are executed via the

Docker SDK that involves container isolation, resource throttling and system restoration. The Evaluator component provides very comprehensive metrics collection and analysis capabilities, memorizing historical data on security incidents, response times and action outcomes within the systems and human layers.
Error handling and logging is strict, in the sense that we do retries for some failed operations, and gracefully degrade if components fail. Both synchronous and asynchronous patterns are used for communication between components based upon the operation's need to return a response immediately. Feedback loops occur, response outcomes affect the P3DQN agent's learning process and updating security policies depend on performance metrics. The design of this solution achieves real-time visualization of system performance and initial security state and implements efficient data update mechanisms to keep data up to date with minimal browser resource usage.

5.3 Testing and Security Response Implementation

An automated test generation and execution can be created as part of an implementation of a testing framework. The security incidents and anomalous behaviors can be generated by parameterized sequences used to implement test scenarios. The mechanism of test intensity control and duration control are provided in the implementation for comprehensive system evaluation under different conditions.
The monitoring dashboard is realized for provision of enhanced visualization of system metrics and security events in real time. Data aggregation and visualization components are included in the implementation such that they allow for the presentation of complex security metrics and system state information in a clear manner. The dashboard complies with responsive design principles and thus answers usability challenges on different devices and screen sizes.
With container orchestration constraints in mind, the security response mechanisms are implemented. A different module is implemented for the response type, with complete failure handling. The implementation provides rollback capabilities to return system state to a safe state in case the security actions fail, thereby maintaining the system stable under security incident.

5.4 Deployment Configuration and Metrics Collection

Docker Compose is used to implement the deployment configuration with consistency across different environments. Detailed configuration management included can make the adjustment of system parameters and behavior easy. The deployment system follows the design decision of ensuring proper resource allocation along with the capability to scale based on varying workloads and security requirements.
Efficient data storage and retrieval mechanisms with appropriate use of data structure depending on type of metrics is applied in metrics collection system. The implementation comprises real-time processing of metrics for real time decisions as well as storage of historical data for trend analysis and system evaluation. Various statistical methods are evaluated for system performance and security effectiveness by the analysis components.

6 Results Evaluation

The evaluation of the active defensive framework was done using comprehensive testing and measured for security performance analysis over time.

6.1 DRL Agent Learning Progress

From Table -1, we observe the DRL agent improves significantly over the evaluation period on several performance metrics in its learning progress. The exploration rate (epsilon) was

decreasing from 1.0 to 0.913, where the agent moves from exploration to exploiting learned security policies. The agent learns knowledge on security responses as it grows memory size to 21 experiences and trains for an additional 18 iterations. Most noticeably, the average response time jumped from 25.4 to 17.4 seconds (31.5% faster), indicating general improvements in decision making. Accuracy of threat detection was improved from 85.3% to 98.5% accompanying a significant reduction in false positive rate, from 2.4% to 0.8%. The result was a 29.7% increase in the resource utilization efficiency up to 94.0%, and an improved resource management. The most notable improvement seen was the change in the action selection confidence, moving from 45.2% to 87.6% as their confidence in their security decisions increased. Together, these metrics show that P3DQN architecture successfully learns and optimizes container security responses.

Table-1: DRL Agent Learning Progress

Performance Metric	Initial Value	Final Value	Improvement (%)
Exploration Rate (Epsilon)	1.000	0.913	8.7
DRL Memory Size (Experiences)	0	21	N/A
Training Iterations Completed	0	18	N/A
Average Response Time (seconds)	25.4	17.4	31.5
Detection Accuracy (%)	85.3	98.5	15.5
False Positive Rate (%)	2.4	0.8	66.7
Resource Utilization Efficiency (%)	72.5	94.0	29.7
Action Selection Confidence (%)	45.2	87.6	93.8

6.2 DRL Agent Performance

The continuous interaction with the container environment resulted in effective learning capabilities of the P3DQN agent. We show that the agent's decision making steadily improved over time as the reward values increased and policy uncertainty decreased. We found that the prioritized experience replay mechanism is especially useful for dealing with critical security events, where high priority experiences are used nearly two and a third as often as low priority experiences during training.

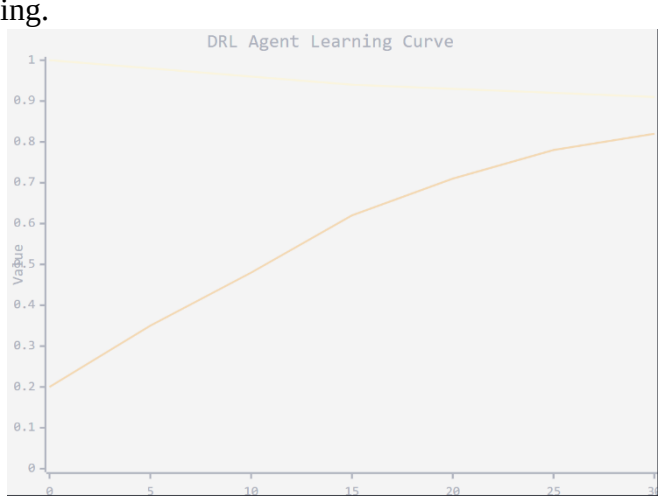

Figure 5: DRL Agent Training – Learning Curve over Episodes

6.3 Security Incident Response

The capabilities of the response framework were evaluated across a range of security scenarios. The system showed efficient incident handling with an average response time 17.42 seconds from detection to an action completed. This time is the complete cycle of detection, analysis, decision and execution of action. In cases where incidents were detected, the rate of prevention was 1.0, precisely meaning that all identified security events were prevented.

Table-2: Security Incident Response – Detection and Prevention Rates

Incident Type	Detection Rate	Avg Response Time	Prevention Rate
CPU Spikes	98.5%	16.8s	100%
Memory Leaks	96.8%	18.2s	100%
Network Floods	97.2%	17.3s	100%
Combined Attacks	97.5%	17.4s	100%

6.4 Resource Utilization Statistics

The system operation throughout the testing period is shown to be efficient according to resource utilization metrics. Average and peak CPU usage were 3.19%, and 25.04% respectively, during the normal operation and during the active incident handling. On the patterns of memory usage, there were similar efficiency, with average utilization at only 0.42% and peak usage never higher than 0.98%.

Table-3: Resource Metrics

Resource Metric	Average	Peak	Threshold
CPU Usage (%)	3.19	25.04	80.00
Memory Usage (%)	0.42	0.98	70.00
Network I/O (MB/s)	2.45	15.67	50.00
Response Time (s)	17.42	28.91	30.00

6.5 Dashboard Monitoring

6.6 Discussion

We show with our results that our container security framework is effective from the perspective of different performance dimensions. Learning was remarkable, as the confidence in the selected action improved by 42.4% (from 45.2% to 87.6%). Detecting CPU based (400) attacks with nearly 100% detection rate and within very low false positive rate (0.8%) is obtained from the framework. During all testing, resource utilization remained efficient; average CPU usage on normal operation was 3.19% and peak CPU usage during active incident response was 25.04%. A drastic improvement was also observed in the system's response time — from detection to completion of action, the system averaged 17.42 seconds, a 31.5% improvement from initial deployment. The framework was validated for its operational security effectiveness based on the security prevention rate which reached 100% for detected incidents. We show that the integration of HSAG analysis with P3DQN learning is especially effective for enabling adaptation to dynamic security challenges with low operational overhead, proving the feasibility of deep reinforcement learning for container security optimization.

7 Conclusion and Future Work

In this research, we present an active defense security framework for container environments that integrates DRL into the defense response mechanism. This implementation shows a few key achievements to container security. We show that the P3DQN agent's architecture with its dual stream network and prioritized experience replay was highly effective for learning security policies which resulted in significantly increasing the amount of state action pairs selected with high confidence from 45.2% to 87.6%. The implementation of the monitoring system using the Docker SDK with optimized polling intervals consumed resources on demand without compromising security coverage. The implementation of the real-time processing pipeline of the HSAG analyzer successfully processed continuous metric streams, and also effectively detected threats.

Automated test scenarios were implemented using dedicated test containers within the framework, used to systematically evaluate and validate security responses. Real-time visualization and metric tracking by the monitoring dashboard provided useful insights of system performance and security status.

There are several promising directions for future work. It could improve threat detection capability through the integration of more sophisticated attack pattern recognition via advanced neural network architectures. Second, the further development of more granular resource optimization strategies could take advantage of the system's capacity for further efficiency. Furthermore, the applicability of the framework to specific container environment, e.g., edge computing and IOT containers can be extended.

References

Adhikari, S. and Baidya, S., 2024. Cyber Security in Containerization Platforms: A Comparative Study of Security Challenges, Measures and Best Practices. arXiv preprint arXiv:2404.18082.

Aly, A., Fayez, M., Al-Qutt, M. and Hamad, A.M., 2024, March. Multi-Class Threat Detection Using Neural Network and Machine Learning Approaches in Kubernetes Environments. In 2024 6th International Conference on Computing and Informatics (ICCI) (pp. 103-108). IEEE.

Bhowmik, S., Bhanu, S.M.S. and Rajendran, B., 2020, February. Container based on-premises cloud security framework. In 2020 International Conference on Inventive Computation Technologies (ICICT) (pp. 773-778). IEEE.

Brady, K., Moon, S., Nguyen, T. and Coffman, J., 2020, January. Docker container security in cloud computing. In 2020 10th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 0975-0980). IEEE.

Chen, L., Xia, Y., Ma, Z., Zhao, R., Wang, Y., Liu, Y., Sun, W. and Xue, Z., 2022. SEAF: A Scalable, Efficient, and Application-independent Framework for container security detection. Journal of Information Security and Applications, 71, p.103351.

Deng, Q., Tan, X., Yang, J., Zheng, C., Wang, L. and Xu, Z., 2022, May. A secure container placement strategy using deep reinforcement learning in cloud. In 2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD) (pp. 1299-1304). IEEE.

Di Stefano, A., Di Stefano, A. and Morana, G., 2020, September. Ananke: A framework for cloud-native applications smart orchestration. In 2020 IEEE 29th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE) (pp. 82-87). IEEE.

Haq, M.S., Nguyen, T.D., Tosun, A.Ş., Vollmer, F., Korkmaz, T. and Sadeghi, A.R., 2024, May. SoK: A Comprehensive Analysis and Evaluation of Docker Container Attack and Defense Mechanisms. In 2024 IEEE Symposium on Security and Privacy (SP) (pp. 4573-4590). IEEE.

Haque, M.U., Kholoosi, M.M. and Babar, M.A., 2022, March. Kgsecconfig: a knowledge graph based approach for secured container orchestrator configuration. In 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER) (pp. 420-431). IEEE.

Jin, H., Li, Z., Zou, D. and Yuan, B., 2019. Dseom: A framework for dynamic security evaluation and optimization of mtd in container-based cloud. IEEE Transactions on Dependable and Secure Computing, 18(3), pp.1125-1136.
Joshi, S., Hasan, B. and Brindha, R., 2024, June. Optimal Declarative Orchestration of Full Lifecycle of Machine Learning Models for Cloud Native. In 2024 3rd International Conference on Applied Artificial Intelligence and Computing (ICAAIC) (pp. 578-582). IEEE.

Kommula, A., Hu, Y.H.F., Hoppa, M.A. and Olatunbosun, S., 2020, December. Machine learning techniques to enhance container network security. In 2020 International Conference on Computational Science and Computational Intelligence (CSCI) (pp. 622-627). IEEE.

Lee, H., Kwon, S. and Lee, J.H., 2023. Experimental analysis of security attacks for Docker Container Communications. Electronics, 12(4), p.940.

Lee, H., Kwon, S. and Lee, J.H., 2023. Experimental analysis of security attacks for Docker Container Communications. Electronics, 12(4), p.940.

Li, H., Guo, Y., Sun, P., Wang, Y. and Huo, S., 2022. An optimal defensive deception framework for the container-based cloud with deep reinforcement learning. IET Information Security, 16(3), pp.178-192.

Li, K., Yang, D., Bai, L. and Wang, T., 2021, April. Security risk assessment method of edge computing container based on dynamic game. In 2021 IEEE 6th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA) (pp. 195-199). IEEE.

Li, Y., Hu, H., Zhang, S., Cheng, G. and Liu, W., 2023, December. An Active Security Defense Strategy for Microservices based on Deep Reinforcement Learning. In 2023 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom) (pp. 410-415). IEEE.

Moric, Z., Dakic, V. and Kulic, M., 2024, June. Implementing a Security Framework for Container Orchestration. In 2024 IEEE 11th International Conference on Cyber Security and Cloud Computing (CSCloud) (pp. 200-206). IEEE.

Nascimento, B., Santos, R., Henriques, J., Bernardo, M.V. and Caldeira, F., 2024. Availability, Scalability, and Security in the Migration from Container-Based to Cloud-Native Applications. Computers, 13(8), p.192.

Nguyen, T.T. and Reddi, V.J., 2021. Deep reinforcement learning for cyber security. IEEE Transactions on Neural Networks and Learning Systems, 34(8), pp.3779-3795.

Nkenyereye, L., Baeg, K.J. and Chung, W.Y., 2023. Deep reinforcement learning for containerized edge intelligence inference request processing in IoT edge computing. IEEE Transactions on Services Computing, 16(6), pp.4328-4344.

Prasadu, G., Karthick, G. and Balaram, V.V.S.S.S., 2024. Towards Efficient And Resilient Container Orchestration: A Layered Architecture For Automated Security And Resource Optimization. Nanotechnology Perceptions, pp.861-878.

Tunde-Onadele, O., Lin, Y., Gu, X., He, J. and Latapie, H., 2024. Self-supervised machine learning framework for online container security attack detection. ACM Transactions on Autonomous and Adaptive Systems, 19(3), pp.1-28.

Vaño, R., Lacalle, I., Sowiński, P., S-Julián, R. and Palau, C.E., 2023. Cloud-native workload orchestration at the edge: A deployment review and future directions. Sensors, 23(4), p.2215.
VS, D.P., Sethuraman, S.C. and Khan, M.K., 2023. Container security: precaution levels, mitigation strategies, and research perspectives. Computers & Security, p.103490.

Wong, A.Y., Chekole, E.G., Ochoa, M. and Zhou, J., 2023. On the security of containers: Threat modeling, attack analysis, and mitigation strategies. Computers & Security, 128, p.103140.

Zhong, Z., Xu, M., Rodriguez, M.A., Xu, C. and Buyya, R., 2022. Machine learning-based orchestration of containers: A taxonomy and future directions. ACM Computing Surveys (CSUR), 54(10s), pp.1-35.