AMERICAN ACADEMIC PUBLISHER

International journal of IoT (ISSN: 2692-5184)

Volume 05, Issue 01, 2025, pages 190-211
Published Date: -24-06-2025
Doi: -https://doi.org/10.55640/ijiot-05-01-10

Automated Firewall Policy Generation with Reinforcement Learning

Ashutosh Chandra Jha
Network Security Engineer, NewYork, USA

ABSTRACT

Network security would be incomplete without firewalls that control traffic flow through rule-based policies. The manual way to configure and manage firewall rules, however, is prone to various pitfalls; rules tend to become overly complex, human error occurs, and cyber threats continue to evolve. This work investigates the reinforcement learning (RL) - driven method for firewall policy generation, utilizing RL as an automated means for policy generation to increase adaptability and reduce administrative overhead. The proposed system utilizes RL agents that learn an optimal policy from real-time network traffic and dynamically update firewall rules to maximize security while minimizing false positives and latency. The key contributions of this work include a novel system architecture that integrates reinforcement learning (RL) with existing firewall frameworks, as well as methodologies for data collection, feature engineering, and reward function design. Additionally, the system is evaluated using simulated network environments and benchmark datasets. It is demonstrated that the RL-based system achieves better accuracy in threat detection compared to traditional static or heuristic approaches, as well as improved policy effectiveness and network performance. Computational cost, explainability, exploration risks, and model generalization are discussed, and future research directions in transfer learning, multi-agent coordination, and integration with broader security frameworks are addressed. This work moves the field closer to realizing real-time, intelligent, and adaptive firewalls that can handle today's cybersecurity challenges. It motivates further exploration of more secure, interpretable, and production-ready RL-driven security solutions.

Keywords: *Firewall automation, Reinforcement learning, Network security, Adaptive policies, Cyber threat detection*

1. INTRODUCTION

Firewalls are critically important network security devices as they act as gatekeepers that determine the flow of traffic into and out of computer networks. Therein lies the heart of what their operation is all about — firewall policies or sets of rules that determine which data packets should be allowed to pass and which to block. These policies enable networks to protect themselves by denying unauthorized entry, removing potentially harmful data, and permitting the safe and legitimate exchange among connected devices. This is a testament to the growing complexity of cyber threats, for which firewalls are becoming increasingly essential tools with ever stronger and more efficient firewall policies to ensure network security and integrity.

In the traditional case, firewall policies are written manually by those analyzing network requirements and security considerations to create a rule at a time. This approach works well when the network is much smaller or simpler, but rapidly breaks down as network size and complexity grow. Enterprise networks in the modern world often

require hundreds or even thousands of firewall rules to support the diverse range of devices, applications, and various types of traffic they accommodate. With this large and dynamic rule set, managing it manually creates numerous issues: for example, it is easy to develop either rule conflicts or overlaps or to miss existing rules. The errors may occur accidentally, creating security vulnerabilities or interfering with legitimate network operations, which can result in costly downtime or a breach. Ultimately, the manual process is both time-consuming and prone to errors, placing a significant burden on on-site administrators.

This creates these challenges, which are exacerbated by the rapidly evolving nature of cyber threats. Network traffic patterns are unpredictable due to the continuous development of new attack tactics, techniques, and procedures. These rapidly changing vectors make vectors render static firewall policies, which are configured once, quickly outdated. The problem is that purely human-written rules leave networks vulnerable to emerging threats, inadvertently blocking legitimate traffic with outdated or overly restrictive policies. This imposes a great need for automated and intelligent firewall management systems that can dynamically adapt to the real-time state of the network and the evolving threat landscape. There are several benefits to automating the generation of firewall policies. Continuously monitoring network traffic, analyzing traffic patterns, and adapting policies automatically in real-time helps reduce human error as well as the administrative overhead associated with updating manual rules. Such systems enable networks to respond quickly to new attacks or changing conditions, thereby enhancing overall network security and improving its operational efficiency. However, automated solutions that learn and perform well in complex network environments while maintaining trustworthiness and performance are an open technical problem.

This article explores the application of reinforcement learning (RL), a subfield of machine learning that enables an agent to learn decision-making through trial and error by interacting with its environment, as a promising technique for generating automated policies. The goal thus focuses on creating an adaptive system that can extract knowledge from real network traffic data, identify risky and safe communications, and operate by dynamically formulated rules that continuously evolve to offer optimal protection. In other words, this work aims to demonstrate how reinforcement learning can alleviate the manual burden of firewall management by building new rules based on real-time network behavior, enabling the detection of configurations and connection servers, and, in general, providing a more adaptive firewall framework. The system's success in blocking malicious traffic while permitting good traffic to pass unimpeded is evaluated, and the performance of the RL-based approach is compared to two static policies and other machine learning-based classifiers. This article presents a practical and scalable approach to designing intelligent and adaptive buildings that align with the goals of today's networked environments.

2. Background and Related Work

To fully understand why reinforcement learning (RL) is used for creating firewall policies, one must examine the systems and techniques employed and how machine learning and artificial intelligence have been applied to the broader security domain.

2.1 Traditional Methods of Firewall Management.

Computer networks have relied on firewalls as a basic layer of defense for ages. Essentially, they are composed of a set of rules that determine whether to allow rock or block specific types of traffic. Usually, these rules define which IP addresses, ports, and protocols traffic needs to satisfy to be accepted. Often, these rules are manually configured by system administrators using platforms such as iptables, Cisco ASA, or sense. The benefit of this manual configuration approach is precise control; however, it is problematic in large or frequently changing networks. Each time a new device, service, or security requirement is introduced, new rules or changes to existing

ones may be implemented. As networks become increasingly complex, they are more likely to encounter rule conflicts, redundancies, or security gaps. But these rules are also static by nature; they don't adjust dynamically to new patterns observed in traffic or changes in threat behaviors.

2.2 Policy Optimization & Conflict Resolution

Researchers and practitioners have explored methods for optimizing firewall policies to address some of these challenges (26). The goal of rule reordering algorithms is to increase processing speed by placing the top rules in a list that matches most frequently. Therefore, conflict detection systems search for overlapping policies where two or more rules can contradict each other. If not intended, the incorrect traffic can either be allowed or blocked. Some systems include heuristics that eliminate redundant entries or combine similar rules. These methods are helpful but still reactive, often requiring human oversight. However, they do not address the core limitation of static policies, namely their inability to adapt when attackers' tactics change or the network environment changes. Ultimately, these optimizations serve best as support tools and cannot replace the need for a more intelligent and automated solution.

2.3 AI for Cybersecurity

In recent years, artificial intelligence, particularly machine learning, has seen broad applications in cybersecurity (1). Intrusion Detection Systems (IDS), spam filtering, and malware classification are a few models that have been widely used with supervised learning techniques. These models learn patterns in labelled data that help them recognize potential threats. After deployment, they can automatically flag suspicious activity that may require human review or automated response (13). For anomaly detection, unsupervised learning methods are also used, whereby the system learns what regular traffic should look like and raises an alert if it detects something unusual. The value of these approaches lies in their ability to detect new attack types that do not yet have a signature. While AI models like these are promising, they are not deployed in the same manner as most models; they are used solely for detection rather than for taking action, such as writing firewall rules. As a result, they don't operate well in real-time environments.

As shown in Figure 1, AI technologies have made significant strides in cybersecurity across different categories—from detection to predictive analysis—but their integration into decision-making systems like firewalls remains limited.

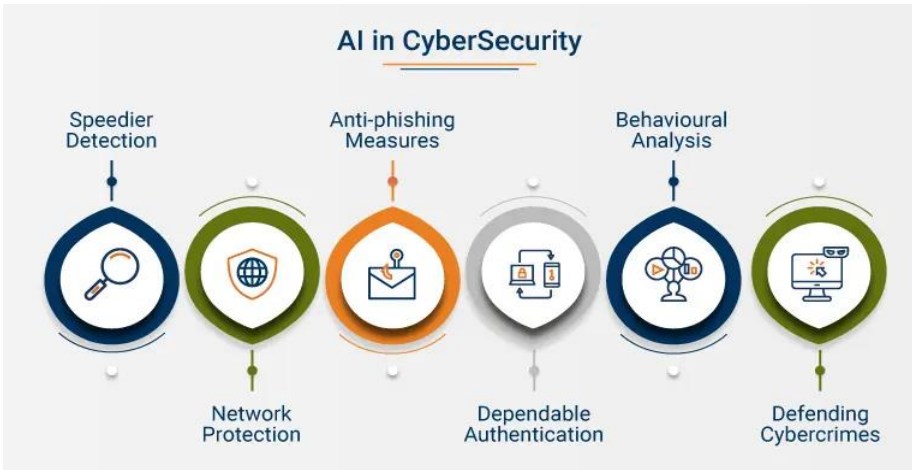

Figure 1: AI in CyberSecurity

2.4 Reinforcement learning in network security

A different approach to these problems is offered by reinforcement learning (22). Supervised models that don't rely

on labelled data learn through trial and error and receive feedback from the environment in the form of rewards or penalties. The result is a feedback loop for RL agents, allowing them to adapt over time and learn to behave more effectively in response to new scenarios. RL has been explored in the field of cybersecurity, particularly in access control, attack response planning, and various forms of anomaly detection. This can be implemented, for example, by deciding whether to activate specific defensive mechanisms or isolate compromised hosts based on their behavior. They demonstrate that RL can handle dynamic environments and discover powerful strategies without relying on pre-labelled data. In terms of research and practice, however, the usage of RL is still particularly limited to firewall policy generation (when one is required to generate, update, or remove filtering rules in real-time).

2.5 Gap in the research

While there has been remarkable progress in applying machine learning and reinforcement learning to security tasks, a clear space remains for solutions that automate dynamic and adaptive firewall rule generation. In most cases, current systems rely on manual updates, basic automation, or detection-based alerts that require follow-up action. It is necessary to develop a system that goes beyond detection, acting as a learning agent capable of understanding network traffic, selecting the best rule to apply, and adapting its strategy over time. Reinforcement Learning could meet this need; however, so far, there are no real-world studies or tools that apply RL to this task directly in an integrated fashion, making it deployable.

3. Problem Definition

This requires defining the problem in both technical and practical terms and demonstrating how it can be effectively applied to the context of reinforcement learning to generate firewall policies. It details what exactly firewall policy generation means, how one can represent rules suitable for machine learning models, and the objectives and constraints that must be considered. With this foundation, one can understand how to train an RL agent to make informed decisions in a network security context (8, 9).

3.1 Firewall Policy Generation Problem (FPGP)

The problem of generating the best set of rules that determines which traffic should be admitted through or blocked from a computer network, regarding a firewall policy generation problem (FPGP), is at the heart of it. In legacy systems, this process is completed manually by administrators who view traffic patterns, security alerts, and user needs and then create or update rules. In a reinforcement learning context, however, this becomes a decision-making problem instead. In this case, the decision maker is the RL agent. Instead, its environment is a network of traffic flowing through the system. The agent observes the current network state of the game and takes action to create a new rule, modify an old one, or do nothing. The agent receives a reward or penalty based on the outcome, such as improved security or the blocking of legitimate traffic. With each cycle, this feedback can help the agent learn a policy: a collection of actions that results in the best outcomes (21). Unlike a simple classification, the FPGP differs in its sequential nature and decisions based on the cumulative effect of previous rules. For example, there might be some rules that override or conflict with each other, which makes the generation task more complex.

3.2 Representation policies

For the RL agent to send a fine-tuned representation of a firewall, it can manipulate a firewall policy that consists of individual rules with a tuple-like structure. A rule can be described as each of those. They display (action, source IP, destination IP, protocol, port, direction, and priority) and evaluate those using simple trigonometric functions. For instance, a rule such as (block, 192.168.1.10, any, TCP, 80, inbound, high) may be defined to block incoming TCP

traffic (port 80) from the IP address 192.168.1.10. This structure must be converted into numerical or categorical values that the agent can process in a reinforcement learning system. The rule can be encoded or embedded by each component of the control, such as the IP address or protocol. So, the policy (in this case) can be viewed as a list of these rules; that is, the rule constitutes the action space available to the agent. During learning, the agent may add a rule, delete one, or reorder the list to increase its effectiveness. Each component of the firewall rule is encoded or embedded according to its data type, as shown in the table below

Table 1: ***Firewall Policy Rule Representation***

Component	Description	Data Type Example
Action	Allow, Block, Log, Rate-limit	Categorical (1 = allow)
Source IP	Originating IP address	Encoded String
Destination IP	Target IP address	Encoded String
Protocol	Type of protocol (e.g., TCP, UDP)	One-hot encoded
Port	Source or destination port	Integer
Direction	Traffic direction (Inbound/Outbound)	Binary (0 = in, 1 = out)
Priority	Rule application priority	Integer or ordinal rank

3.3 Objectives and Constraints.

The way forward is to consider any system that automatically generates firewall policies as one that must simultaneously satisfy several key objectives. Among these, the first is security, which involves effectively blocking unauthorized traffic and preventing malicious activity from occurring. In addition, the system must maintain network performance by allowing legitimate traffic to flow smoothly without inducing high latency or disrupting critical business services. Just as important is minimizing false positives and false negatives: it's distracting to users and limits their productivity if safe traffic is blocked or if dangerous traffic is allowed through. Therefore, the system must be carefully balanced by these competing factors.

The policy must also be dynamically responsive to changes in the environment, adapting to the network's growth or refining itself to address new threats as they emerge (3). Finally, a set of policies should be kept as concise and manageable as possible, as excessively long lists of rules can be challenging to audit, maintain in sync, and read. However, achieving these objectives is difficult, as improving one domain may at the same time make the other worse—for example, tightening the policy may reduce threats but lead to many false positives. As a result, the system will have to learn and balance these trade-offs to generate optimal and efficient firewall policies.

3.4. Challenges in Policy Generation

Firewall policy generation is a complex task in its own right. The main issue is that rules interact with each other in complex ways; firewalls process rules in sequence, and one rule can trump or cancel out another. Small changes can give rise to significant and unexpected consequences, which makes it hard to imagine the consequences of introducing a new rule without testing it carefully. The other challenge is real-time adaptation—networks run continually and need instant changes to the policy to address any upcoming threat. At the same time, reinforcement learning models are capable of learning and evaluating outcomes, which necessitate slowing down, as they need to result in necessary changes.

The scalability problem arises particularly in large networks, such as those with thousands of devices, where the

number of possible combinations of rules further grows exponentially. Without being inundated by the volume of data, the system must be able to make fast and efficient decisions. Moreover, a rule is often applied without immediate feedback on its effectiveness, as problems may not appear until later, for example, when rare traffic is blocked or when attackers find an easy way around protections. The learning and optimization process becomes more challenging since this feedback is necessarily delayed. When designing a reinforcement learning system to generate firewall policies, addressing these challenges is crucial. The following will focus on the basics of reinforcement learning and highlight how it can be effectively adapted to solve practical problems (28).

4. Reinforcement Learning Fundamentals

The goal is to apply reinforcement learning (RL) to the generation of firewall policy, and the first step is to understand how RL works. Compared to other machine learning techniques, RL differs in that it focuses on learning through interaction. This section describes how reinforcement learning works, its key components, and its applications in network security (18).

As shown in Figure 2, a policy-based RL system involves several components: the agent, the environment (in this case, the network traffic system), a policy module (which decides the agent's actions), and a reward mechanism (which evaluates the effectiveness of those actions).

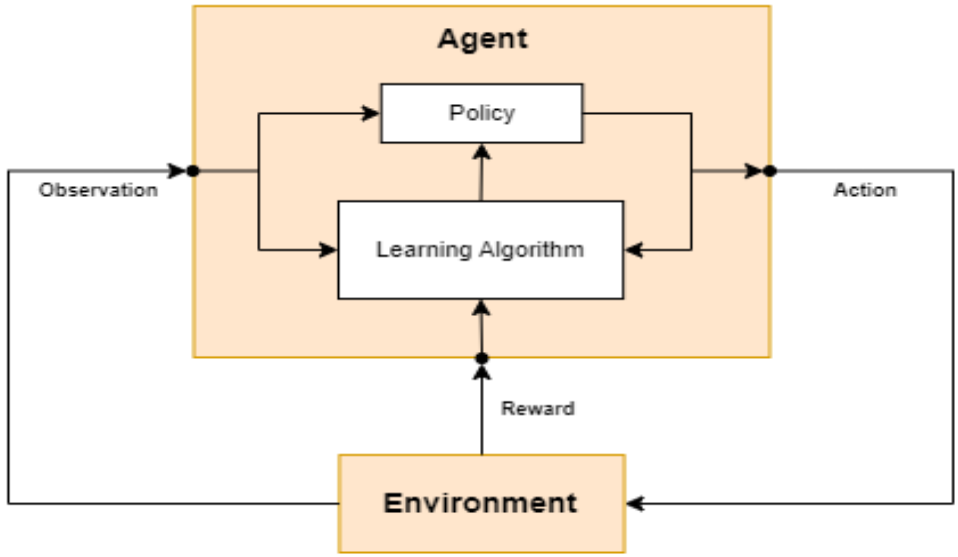

Figure 2: Diagram of a Policy-Based Reinforcement Learning System

4.1 Reinforcement Learning Basics

In machine learning, reinforcement learning is a type of learning where an agent learns while interacting with an environment. It is observed that the agent receives state information, and then feedback in the form of a reward or penalty is provided. The agent then updates its strategy based on this feedback with the hope of better actions in the future. Over time, the process continues, allowing the agent to learn how to maximize total rewards. Reinforcement learning works through trial and error and experience, unlike supervised learning, which depends on labelled data. The agent then explores different actions and learns which one is the most effective way for it to reach its goals. A key component of reinforcement learning (RL) involves four main elements: the environment (in which the agent operates), the state (representing the current situation of the environment), the action (the decision made by the agent), and the reward (which signals the effectiveness of the agent's decision). As illustrated

in Table 2, the RL agent acts as the policy generator, interacting with a network environment that provides real-time traffic data, while the state, action, and reward are defined to reflect network conditions and policy effectiveness.

Table 2: ***Reinforcement Learning Mapping for Firewall Policy Generation***

RL Component	Firewall System Equivalent
Agent	RL-based policy generator
Environment	Network with real-time traffic flow
State	Current snapshot of the network and threat indicators
Action	Generate, modify, delete, or reorder firewall rules
Reward	Feedback based on rule effectiveness: security, performance, simplicity

4.2. Markov Decision Processes

Many reinforcement learning models are built from something called a Markov decision Process, or MDP. The decision-making problem is described mathematically, known as an MDP. It has a set of possible states, a set of actions the agent can take, a reward function that tells the agent what it gains for each action, and a transition function that describes how the environment transitions from one state to another. The property defines an MD P that allows the future to depend not only on the current state but also on the current action, regardless of how it arrived in that state (30). It makes things easier to learn. The state could include current traffic statistics, recent alerts, and the current rule set for generating firewall policies. The available actions are to add a new rule, modify an existing rule, or take no action. However, the reward would be a function of how well the new policy performed — e.g., how much bad traffic it blocked or how much good traffic it disrupted.

4.3 Exploration vs. Exploitation.

The balance between exploration and exploitation is one of the central ideas in reinforcement learning. Trying new actions to learn about the environment is called exploration. Exploitation is using what the agent already knows to make the best decision. The agent in firewall policy generation must navigate various rule sets to find the most effective ones. However, it must also avoid constantly changing the rules, which risks poor performance. Managing this balance is essential: if the agent explores too little, potential improvements may be missed; if it explores too much, unnecessary disruption results. Similarly, overexploiting too early may cause the agent to settle on a good policy but not the best (29).

4.4 Reward design and feedback loops

Reinforcement learning has a vital role in reward functions. This directly affects the behavior of the agent. Network security has a range of goals to achieve: stopping threats while avoiding false alarms, maintaining high performance, and adapting to changes. Designing a reward function is not an easy task. It must be able to provide helpful feedback even in cases where the results of a decision aren't immediately visible. For example, one rule may block an attack, but its benefit will not be apparent for hours later. A rule that appears quite sensible can surprise in peak usage, just as a rule can work fine for a while and then cause problems. Feedback loops are one way to improve on this. The RL model feeds back into the system that continuously monitors network performance and security logs. It means that the agent will not only learn based on immediate rewards but also consider some long-term effects.

4.5 The Study of RL in Firewall Policy Generation

Due to these reasons, reinforcement learning is well-suited for generating firewall policies (34). It first tackles the issue of decision-making over time, which is apropos of the dynamic nature of network security. Second, it can learn from interactions, which is essential when there are not many samples of the ideal policies to train on. Thirdly, it is adaptive. Since the network and threats continue to evolve, the RL agent can still learn and improve.

5. System Architecture

The automated firewall policy generation using a reinforcement learning system architecture incorporates several critical components that operate in real time. These components enable the collection and processing of network data, allowing the generation of intelligent policies and their enforcement within the firewall infrastructure. This section explains the architecture's structure, highlighting the role of each component and how the overall system can be integrated into network environments (6). As illustrated in Figure 3, the system supports parallel parsing to ensure scalability and efficiency. Incoming data is processed across multiple threads or nodes simultaneously, allowing the architecture to handle high-volume traffic without bottlenecks.

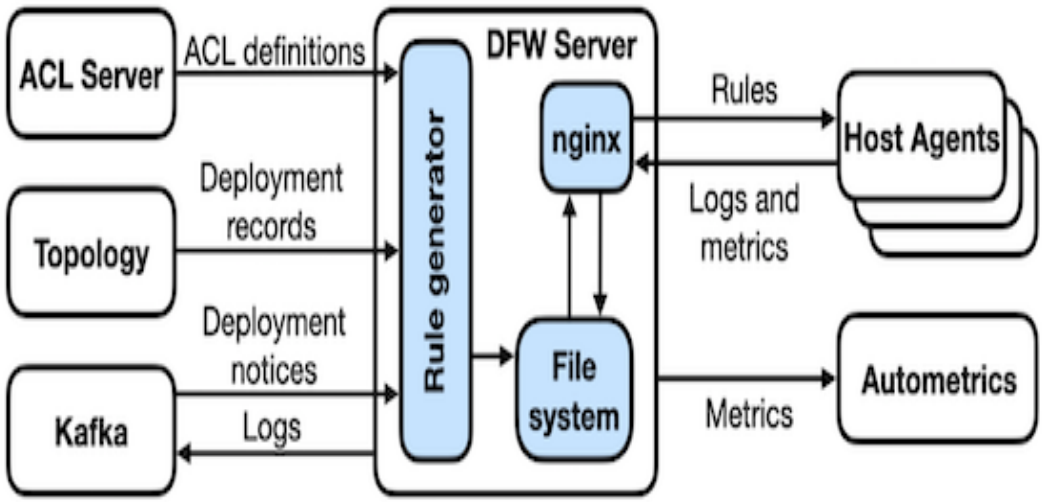

Figure 3: Distributed Firewall (DFW)

5.1 High-level Architecture Overview

The architecture is straightforward and robust. The flow of data from the network traffic monitor to the feature extractor, the reinforcement learning–based policy generator, and the policy enforcement engine. It forms a closed loop whereby the system monitors network behavior, makes reasonable and intelligent decisions, and applies existing firewall rules. This architecture is designed to operate near real-time, detecting changes in the network, analyzing them, and responding without delay (27). The system can remain ahead of advancing threats without manual updates by automating this process. The entities are flexible and well-suited for different types of infrastructure, ranging from small-scale to large-scale enterprise systems. All the modules communicate with each other so that they can make decisions as easily as possible and enforce actions without affecting each other or causing chaos.

5.2 Key components

Four main components constitute the system to automate the firewall process, and each contributes to a particular step in the firewall automation process:

Traffic Monitor

This captures raw network traffic data, including packet headers, flow information, and metadata such as timestamps and source and destination IP addresses. It is a real-time feed that the rest of the system sits on. To tap into the network, this module may be able to do so using port mirroring, packet sniffing, or integration within existing monitors.

Feature Extractor

Next, once the data is captured, it must be processed in a form that a reinforcement learning agent can understand. It parses the traffic data and identifies the protocols used, port numbers, connection durations, and other relevant information, as well as suspicious patterns. These are the state features that the RL model will use to make its decision.

This is the RL Policy Generator.

The reinforcement learning agent is at the heart of the system. Given the extracted features and the state of the network, it selects the optimal action. It could involve creating a new rule, changing an existing one, or perhaps there is no need for any change at all. Over time, the agent receives feedback on which decisions improve network security and performance and learn from this.

Policy Enforcement Engine

An FMC is responsible for linking records between applications, such as a Web application and a backend database server, and enforcing traffic direction through a firewall or filtering out security threats. The last part interprets the decisions made with the RL model into actual firewall rules (20). The rules of these lies are applied to the firewall in real time to control the flow of network traffic. Safety and caution are built into this engine, which, whenever possible, simulates the effects of rules before deployment. It also features a rollback mechanism that allows for the quick undoing of changes.

5.3 Data Flow Process.

Packets flow through the network, and that is where the entire process starts. The Traffic Monitor continuously captures these packets, taking no performance toll on our networks. These packets are passed into the Feature Extractor, where features are extracted from the data, generating clean, labelled data points. This includes information such as destination port, protocol type, packet size, frequency of access by source port, and whether the source is internal or external. The RL Policy Generator takes this structured data as input. At each decision period, the agent examines the current network state and history and then selects an action to optimize the firewall policy. For instance, it could detect a surge of failed connection attempts originating from a specific IP and subsequently block that IP.

When an action is chosen, the Policy Enforcement Engine transforms it into a rule, for example, “blocking incoming TCP connections from IP X on port Y.” This rule is inserted into the firewall rule set. Finally, the firewall is updated while the system remains responsive to existing sessions and adaptive against malicious attackers. This entire process is repeated continuously to keep the system current with network behavior and adapt to new threats in real time. After integrating existing real systems, challenges such as data duplication and significant differences among systems are resolved. The system is designed for practical adoption, compatible with widely used firewall platforms. It can communicate with Linux-based firewalls such as iptables, open-source firewalls like pfSense, and commercial firewalls like Cisco ASA. Brown abstracts the system calls, APIs, or configuration interfaces provided by these platforms into this integration. The Policy Enforcement Engine shares updates with the firewall’s rule

management system when deployed in a live environment. It ensures rules are syntactically valid and follow the business rules that the platform currently operates under. This compatibility also ensures organizations do not need to replace their existing infrastructure to benefit from RL-based policy automation. Additionally, safety mechanisms are integrated into the rules to prevent disruptions. For instance, the system can apply new rules in a sandbox environment before deploying them to the live firewall. This intelligent automation is introduced without degrading system stability or user experience (25, 5).

6. Methodology

The section elaborates on the way the reinforcement learning system is developed and trained to generate an automated firewall policy. It details how data is collected and processed, how the model is structured, how learning is guided, and how to evaluate system performance. The objective is to develop a system that responds to threats in an innovative manner while maintaining a secure and robust network environment.

6.1 Data collection

High-quality network traffic data is needed to train and evaluate the reinforcement learning model. Publicly available and custom-simulated datasets are used in this project (24). The public intrusion detection datasets (e.g., CICIDS2017, UNSW-NB15) are the primary sources, consisting of traffic logged in controlled environments along with a mix of regular and malicious traffic. These datasets are widely used in the research community and contain labelled traffic examples, including DoS attacks, brute force attempts, port scanning, and other typical threats. Other datasets were also developed based on real traffic, and traffic was also simulated within a controlled testbed environment. Such a setup emulated a diverse set of user activities, server operations and network attacks. Because the data is simulated, the model can be tested under various specific conditions, including rare (or novel) attacks that may not be present in public datasets. Training the model on a combination of real-world and simulated data helps ensure that it is trained on realistic and diverse examples, which significantly improves its generalization capabilities.

The datasets used for training and evaluation are summarized in Table 3, detailing the source type, traffic types included, labeling status, and the intended purpose of each dataset

Table 3: ***Datasets Used for Training and Evaluation***

Dataset Name	Source Type	Traffic Types Included	Labeling	Purpose
CICIDS2017	Public	DoS, DDoS, Port Scans, Infiltration, Benign	Yes	Training & Validation
UNSW-NB15	Public	Shellcode, Backdoors, Worms, Fuzzers, Benign	Yes	Benchmarking & Testing
Simulated Testbed Data	Custom-Simulated	File transfers, web browsing, brute force, zero-day	Yes	Scenario-specific model evaluation
Real Traffic Logs	Real-World Capture	User activity, traffic bursts, protocol shifts	Partial	Realism injection during training

6.2 Feature Engineering

Next is the step of extracting meaningful features about each network flow or packet using the collected data? The reinforcement learning model takes as input these features, whose role is to represent the essential components of the network's behavior accurately. A typical feature of data related to network traffic includes source and

destination IP addresses, port numbers, protocol types, packet sizes, and section durations. These indicators help the model distinguish between standard patterns and suspicious traffic patterns. Various preprocessing are applied to prepare the data for learning. Protocol types, such as categorical features, are encoded into numeric values using one-hot or label encoding methods. For instance, features such as packet size or connection duration are continuous and normalized to the same scale to prevent large values from skewing the model's results. It alleviates the problem of noise by reducing features to those that consistently contribute to the decision-making process and consequently improves the model's accuracy.

6.3 Design of state-action space

Reinforcement learning agents require the "state" of the environment to be well-defined and a list of valid "actions" to take. In such a system, the state comprises the current status of the network, real-time traffic counts, threat indicators and the present set of firewall rules. The RL agent can operate within this context if it understands the state, which provides information about the network's current state, including its ongoing activity, given the agent's possible actions in a specific time step. Possible actions include allowing or blocking traffic, logging it for further inspection, rate-limiting suspicious traffic, and modifying and reordering firewall rules. In doing so, these actions provide the agent not only with the ability to respond to impending threats but also with the means to perform optimally and effectively over time.

6.4 Reward Function Design

Feedback is central to guiding the agent's learning on whether the agent's actions are desirable. The reward in this system comes at the expense of security, performance, and policy simplicity. Thus, the agent receives positive rewards for blocking malicious traffic or permitting legitimate traffic. It is penalized if it blocks safe connections (false positives) or allows the attacks through (false negatives); otherwise. The agent is also penalized for producing too many rules or overly specific rules, which can prevent the development of bloated and overly complex policies. Further penalties are inflicted when the rules cause high latency and when the rules are too frequently changed. The reward structure of this problem motivates the agent to learn a policy that can perform well on both threat mitigation and maintaining network performance simultaneously.

6.5 Model Selection and Training

Automated firewall policy generation is complex, and therefore, selecting the appropriate reinforcement learning algorithm is essential. Some notable algorithms include Deep Q-Network (DQN), Proximal Policy Optimization (PPO), and Asynchronous Advantage Actor-Critic (A3C). DQN, however, is a value-based method that works well in problems with an ample action space. As a method that ranges from minor to medium-sized, stable, and efficient policy-based methods, it balances exploration and exploitation. A3C is a parallel learning approach that learns by running multiple agents in parallel. Since PPO is stable, easy to tune, and has demonstrated strong performance in comparable control problems, it was chosen for this project **(**7). Many episodes, which involve agents interacting with environments and updating their policies, utilize the rewards they receive to inform their subsequent actions. Using an exploration strategy, such as epsilon-greedy, is beneficial to balance the discovery of new actions with leveraging known effective policies. The agent will continue to be trained until its performance reaches a relatively strong level of non-degeneracy and degeneracy.

6.7 Policy Change Mechanism.

What is important after training is to regularly relearn the policy to maintain its effectiveness in dynamic network

environments over time. Of the two main learning strategies, one involves on-policy learning, which occurs while actively using the current policy, and the other is off-policy learning, which consists of learning from past experiences. It employs a hybrid approach that combines recent data with historical knowledge, enabling responsiveness to new threats while retaining previously collected insights. The training occurs periodically or when the network environment undergoes a significant change. Before deployment, the updated policies are validated in a test environment to ensure stability without causing unplanned disruptions.

6.7 Metrics Evaluation

Multiple metrics are used to evaluate system performance. These metrics measure the accuracy of the agent—that is, how often it correctly chooses to allow or block traffic. The quality of false positive and false negative rates indicates the effectiveness of threat detection. Additionally, firewall policies should not be changed in ways that significantly affect legitimate users in the system. Finally, any increase in latency due to firewall rules is monitored to ensure it does not significantly impact traffic flow. These metrics are applied throughout both the training and deployment phases to track the learning progress and the system's real-world performance. A well-performing system should maintain high accuracy and low false detection rates while imposing negligible impact on network latency (19).

7. Experimental Setup

This paper describes the environment and tools used to assess the automated firewall policy generation system, as well as the techniques employed for comparing the system's performance relative to baselines.

7.1. Simulation Environment

Several simulation environments were employed to achieve sufficient flexibility and realism in the system under test (10). They were using Mininet, NS-3, and a setup of the cloud lab in a virtualized environment. With Mininet, one could create realistic software-defined network topologies with controllable hosts and switches, allowing for detailed packet-level analysis. The discrete event network simulator NS-3 was used to model complex network protocols and traffic flows, which could be used for stress-testing the RL policy under various conditions. A near-realistic environment was created by housing virtual machines in the virtualized cloud lab, simulating typical enterprise network components for deploying and testing firewall policies.

As shown in Figure 4, the simulation environment also incorporated the SLR protocol for coordinating simulation events and logging outcomes consistently across the different tools and layers of abstraction.

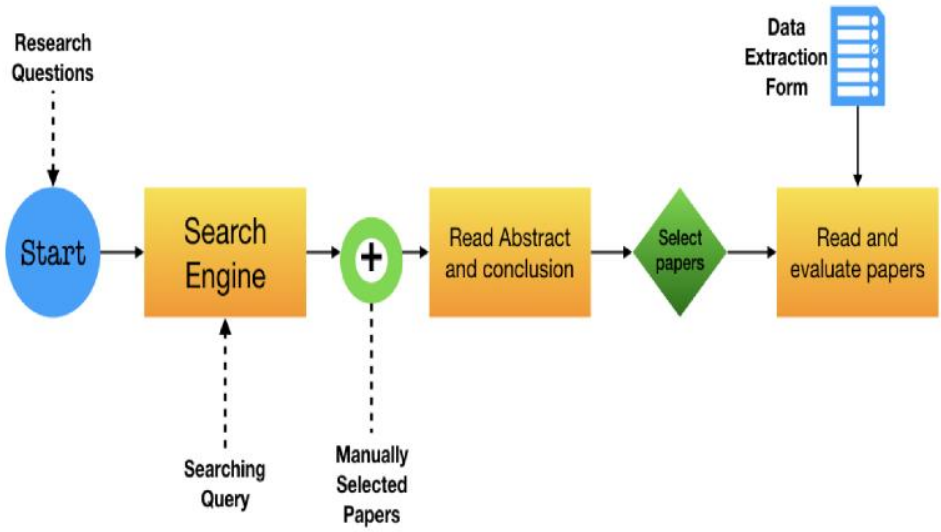

Figure 4: SLR Protocol Used.

7.2 Traffic Generation Tools.

An assortment of tools were used to simulate network traffic. Hping3 was used to send custom TCP, UDP, or ICMP packets, allowing customization to craft attack patterns such as SYN floods or port scans. The public datasets replayed captured traffic traces, allowing for the simulation of realistic network loads using TCPreplay. However, an automated framework utilizing the Metasploit system was able to launch known exploits and penetration testing attacks against the system, thereby challenging its ability to detect such attacks. Custom scripts were also created to build specific traffic scenarios and combinations of benign and malicious flows, covering all cases in testing.

7.3 Baseline Systems

Several baseline systems were evaluated to see how the reinforcement learning-based firewall policy generation fares. These are static firewall policies that are manually configured by network administrators, representing traditional, and fixed-rule approaches. Furthermore, he included heuristic rule-based systems, which seek to detect common attack patterns using predefined threshold or signature-matching rules. Mixed traffic scenarios were used for testing, combining benign and malicious flows (4). Network environments were simulated in these scenarios to contain different ratios of attack traffic. The evaluation involved monitoring to ensure that the system correctly allowed legitimate traffic through while blocking or mitigating malicious activity. Detection accuracy, false positive and negative rates, policy enforcement latency, and rule update frequency were recorded to obtain a comprehensive performance profile for these metrics (23).

8. RESULTS AND ANALYSIS

The reinforcement learning model is trained, and the results are presented. The effectiveness of the generated firewall policies is evaluated, and their performance is compared to that of baseline systems.

8.1 Results from Training

During training, the RL agent demonstrated steady progress, consistently improving its cumulative rewards across multiple episodes, indicating the learning of effective firewall policies. The learning curves converged within a reasonable number of cycles, which helps demonstrate the stability of the chosen PPO algorithm. In early episodes, there was obvious exploration and fluctuating rewards. Still, as training proceeded, the decisions made by the agent

became more sophisticated, wherein security measures were chosen to their fullest potential while unnecessary traffic blocking was avoided. Smooth reward trends in the late training stages are representative of the agent learning to satisfy multiple objectives, such as simultaneously blocking attacks without compromising legitimate connections.

The bar graph below illustrates the RL agent's training progress. It shows a steady increase in cumulative rewards across episodes, reflecting the agent's improved decision-making over time.

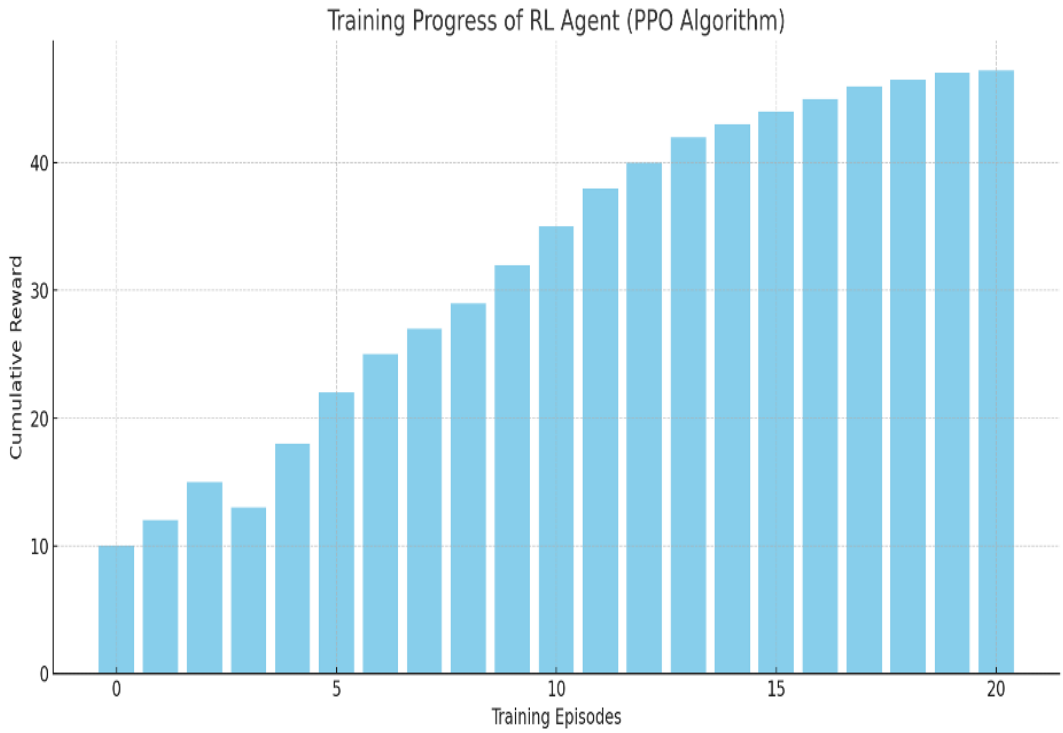

Figure 5: RL agent's training progress

8.2 Effectiveness of Policy

An analysis was conducted to demonstrate that a significant increase in threat detection and mitigation was achieved when comparing the network behavior before and after the application of appRL-generated behavior between firewall rules. These adaptive policies, created by the agent, were proven to be more flexible than static rules because they could adjust themselves to emerging threats on the fly. For instance, when tested in simulated attack scenarios, the RL-based policy prevented unknown and/or poorly policed malicious flows. The system also maintained low false favorable rates, meaning that fake traffic seldom disrupted legitimate traffic, which is critical for both user experience and operational continuity (16).

8.3 Case Studies

The system is further illustrated through specific case studies that highlight its practical impact (14). During Distributed Denial of Service (DDoS) attack simulations, the RL policy quickly adapted to detect and respond to abnormal traffic spikes. It enacted rules to throttle or block offending sources, thereby minimizing network congestion. For cases of SYN flood attacks, the system learned to filter suspicious connection requests while maintaining customary TCP handshakes. It also effectively blocked probing attempts at port randomization by lying and blocking probing behavior, leaving the standard service ports unaffected. For data exfiltration attempts, the

adaptive policy identified unusual outbound traffic patterns, terminating data exfiltration attempts and preventing sensitive information leaks, which demonstrated the effectiveness of the threat coverage in place.
As illustrated in the Figure below, these use cases are supported by a cloud-based attack simulation platform that demonstrates the RL agent's robust, context-aware responses in real time.

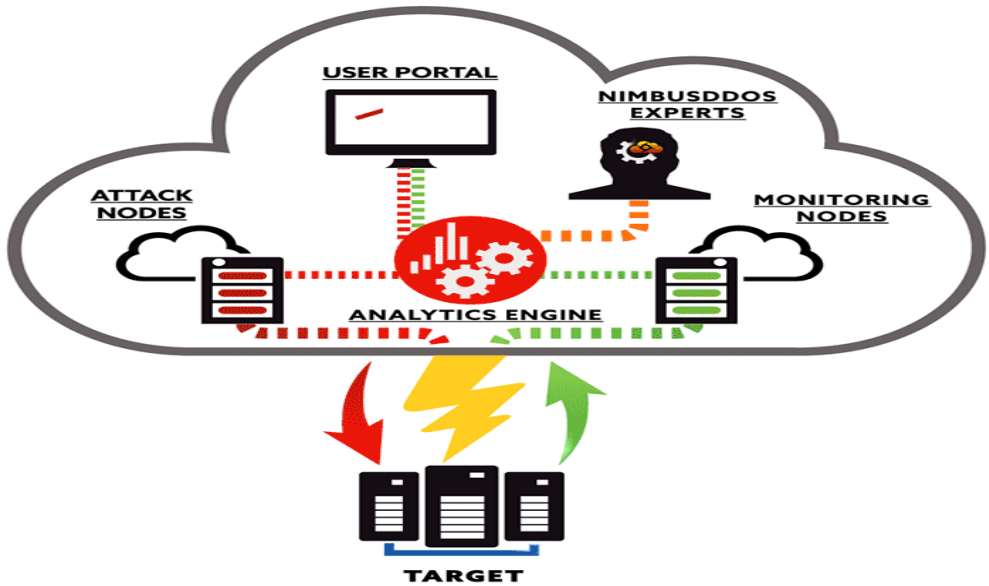

Figure 6: cloud-based-ddos-attack-platform

8.4 Comparative Performance

RL-based firewall generators outperformed other baseline systems, such as static and heuristic rule-based approaches, in various dimensions when compared with them. RL policies adapted to changing traffic conditions by remaining fixed, whereas static policies changed while providing higher detection rates and better quality of generated rules. In comparison to traditional machine learning classifiers, the RL system exhibited better throughput and a minor latency impact, resulting in higher throughput due to its ability to optimize rule sets and avoid unnecessary overhead. At the same time, the false favorable rates were notably reduced, indicating that the discrimination between benign and malicious traffic became more precise. The effectiveness with which reinforcement learning manages these complex, evolving network security challenges confirms these results.
As shown in Table 4, reinforcement learning approaches demonstrate superior adaptability, detection rates, and rule optimization compared to static rule-based systems and traditional machine learning classifiers

Table 4: ***Comparative Performance of RL-Based Firewall Systems vs Traditional Approaches***

Performance Metric	**Static Rule-Based Systems**	**Traditional ML Classifiers**	**Reinforcement Learning (RL) Approach**
Adaptability	Poor – does not respond to new threats	Moderate – retraining required	Highly adaptable to real-time traffic dynamics
Detection Rate	Moderate	High	Very High
False Positive Rate	High	Moderate	Low
False Negative Rate	High	Moderate	Low
Rule Optimization	Manual and rigid	Requires tuning	Dynamic and optimized via learning

Performance Metric	Static Rule-Based Systems	Traditional ML Classifiers	Reinforcement Learning (RL) Approach
Latency Impact	Low	Moderate	Minor
Throughput	Moderate	Moderate	High – due to leaner and smarter rule sets
Maintenance Overhead	High – frequent manual updates	Medium – needs retraining and tuning	Low – self-adjusting rules
Suitability for Evolving Threats	Low	Moderate	High

9. Challenges and Limitations

9.1 Computational Requirements

While the results are promising, a major challenge is that training complex reinforcement learning models can be computationally expensive (11). Simulating these problems in realistic and dynamic network environments requires these models to run for extended periods and utilize demanding computing resources. This may limit the number of times retraining will be allowed and, in turn, delay the system's ability to react quickly to emerging threats. Moreover, the model must infer speedily while deployed in the live scene. Latency resulting from this may hurt network performance, providing visibility to the system when traffic is changing rapidly or exposing the system to vulnerabilities.

As shown in the Figure below, the fundamental architecture supporting this system must balance real-time policy generation with high-throughput data processing. It outlines how the model interfaces with data ingestion, feature extraction, policy evaluation, and action enforcement components within a resource-optimized loop.

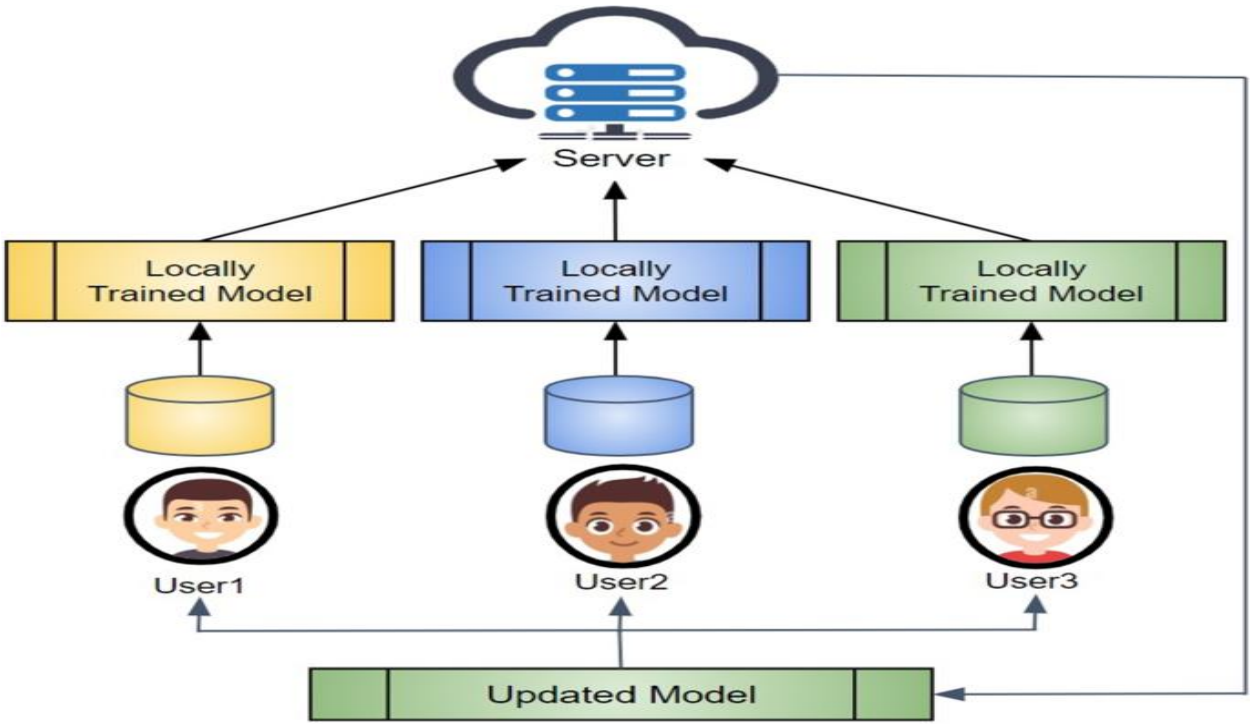

Figure 7: Basic architecture for FL

9.2 Explainability Gap

Another critical limitation when utilizing RL-generated policies is the interpretability of those policies (12). The agent creates firewall rules that often involve complex, non-intuitive combinations, making them difficult for human administrators to understand or audit. In security configurations that require high validation, compliance, and explainability, this lack of transparency can undermine trust and acceptance of automated policy systems. Reinforcement learning also involves an exploration phase, which carries inherent risks. During this phase, the agent experiments with various actions to find optimal policies but may accidentally block legitimate traffic or allow malicious flows. If not carefully controlled, this trial-and-error process can cause disruptions or security gaps. In live networks, these mistakes can be severe; therefore, it is imperative to implement safe exploration and fallback strategies to minimize potential damage during training (31, 32).

10. Future Work

10.1 Transfer Learning

Transfer learning techniques can be applied to reinforcement learning-based firewall policy generation, representing a promising direction for future research. By reusing policies learned in one network environment, the system can adapt more quickly to similar or evolving network conditions without requiring extensive retraining from scratch. This approach would enhance both efficiency and responsiveness, particularly in dynamic, large-scale deployments where network traffic and threat landscapes frequently change (33).

10.2 Multi-Agent Systems

Another important area of exploration is leveraging multi-agent reinforcement learning to allow coordinated policy generation across distributed firewalls or enterprise networks. With such systems, multiple agents' shared information is optimized, providing a more holistic and adaptive defense mechanism for complex architectures.

As illustrated in the Figure below, multi-agent reinforcement learning in action involves agents interacting not only with their local environments but also with each other, exchanging state or reward signals to align on global policy objectives.

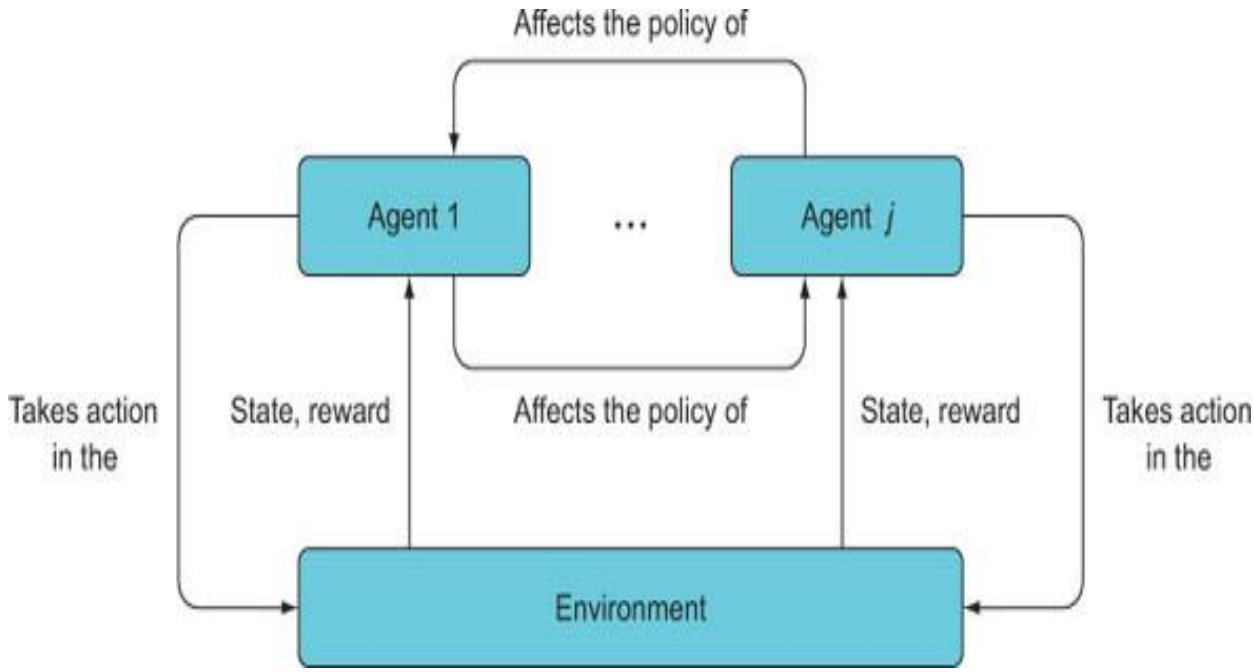

Figure 8: deep-reinforcement-learning-in-action

10.3 Explainable RL.

Despite considerable progress in reinforcement learning, improving the explainability of learned policies remains a significant challenge (35). Future work could focus on developing methods to make learned rules more interpretable and transparent, allowing administrators to audit, trust, and improve automated decisions. A few techniques from explainable AI (XAI) can be applied to render those policy changes human-understandable. Future implementations would incorporate the RL-based firewall system with a Security analytics platform for decision-making. By utilizing more contextual data and real-time threat intelligence, the RL agent can make informed policy adjustments based on the overall network security posture, thereby enhancing detection accuracy and response times.

10.4 Hybrid learning approaches

The ability to embed reinforcement learning in a supervised manner provides an avenue to increase model performance. Training the system on labelled attack datasets during supervised phases, optimizing the policy concurrently, and employing a hybrid learning approach may produce firewall policies that are more robust and effective than those generated by using either reinforcement learning or supervised learning alone (17).

12. Recommendations

Practical recommendations are proposed to harness reinforcement learning effectively (RL) for firewall policy automation. For this reason, organizations are advised to initially adopt a hybrid deployment approach that combines manual oversight with automated policy generation. During this transition, the safety of operation and trust in the RL system are maintained, enabling human experts to monitor and intervene if deemed necessary. It is a phased rollout that minimizes disruption and allows the technology to spread more smoothly (2). It is equally critical to enforce robust policy validation protocols. RL-generated policies must be extensively offline-tested on extensive, realistic, and real-world data sets before deploying them to live environments. This step eliminates the risk of a false positive, preventing legitimate traffic from being blocked. The robustness and responsiveness of the learned policies under various types of threats can be carefully benchmarked and also simulated in attack scenarios. Another essential feature of an effective RL-based firewall is continuous monitoring. Feedback loops that allow an RL agent to learn from new traffic behavior should be coupled with real-time performance tracking. The adaptability of the system ensures that it withstands the offensive change and that it adapts changes to a dynamic environment, achieving optimal results with and achieving modular yet synergized with other security tools, such as Security Information and Event Management (SIEM) systems, Intrusion Detection/Prevention Systems (IDS/IPS), and threat intelligence feeds, it enables increased contextual insight into the incident it is intended to manage, improving its decision-making. This synergy allows for the RL agent to adjust its policy in a more informed and situationally relevant manner. It is organizational readiness that makes the difference in implementation. RL model behavior needs to be trained by the security team, and any interpretations or corrective actions must be taken as required. The human-in-the-loop governance model ensures trust and accountability that can be consistent with existing IT security protocols, paving the way for sustainable, intelligent firewall automation (15).

12. CONCLUSION

The results demonstrate that reinforcement learning can effectively automate the generation of firewall policies while potentially alleviating the increasing complexity of network security management in such environments. It learns the optimal policies to adapt to ever-changing network threats dynamically, balancing conflicting security requirements with the need for efficient network performance. Experimentation was performed in the system, and through it, a marked improvement in key performance indicators was observed, including increased detection

accuracy and reduced false positives, which is critical to limiting the amount of unnecessary disruption to legitimate network traffic. Notably, latency was not sacrificed to achieve these gains; therefore, reinforcement learning holds promise for practical application in real-world firewall automation scenarios, where low latency is critical to maintaining smooth network operation.

This research core comprises several key contributions that expand both the theoretical knowledge base and the practical application of RL in cybersecurity. In addition, the study outlines a new system architecture that enables and integrates existing firewall management frameworks with reinforcement learning algorithms, making this combination practically feasible. This allows the continuous refinement of such policies in response to real-time observations of the network, thereby bridging the gap between automated decision-making and enforceable security measures. Second, a comprehensive methodology was adopted, encompassing all building blocks of the workflow, from collecting and preprocessing various network data to performing intelligent engineering of the network features and designing a reward function to achieve the goal of securing the environment in the real world. A rigorous methodological design ensures that the model not only learns effective policies but also generalizes well to various network conditions and threat profiles. Thirdly, in a thorough evaluation against a wide array of threat scenarios, including denial-of-service attacks, port scans, and attempts at data exfiltration, the system is demonstrated to be effective and robust in addressing complex cybersecurity problems. Collectively, these contributions extend the state of the art of AI-driven network security towards putting reinforcement learning into production in environments more typically governed by static or manually administered rules.

These findings have profound implications for network security. This research is a step toward developing real-time, adaptive, and intelligent firewall systems whose responses are also autonomous in the face of the ever-changing landscape of cyber threats. Through the ability to adapt policies based on experience, firewalls can become more resilient against sophisticated attacks while being less and less dependent on human intervention. This takes the weight off the shoulders of security administrators, allowing them to focus on strategic defense, such as fingerprinting, rather than frequent rule upgrades. In addition, adaptive systems provide more proactive defense approaches; that is to say, they forecast and prevent damage from occurring, reflecting cybersecurity needs in an accelerated digital environment.

Though promising, this study also notes that there is certainly more research to be done. However, it is essential to note that there remains a significant opportunity to enhance explainability for reinforcement learning–based firewall policies. With learned policies, the complexity and opacity cause issues for human comprehension and trust, which are prerequisites for adoption into operational environments with extensive compliance and auditing requirements. Moreover, these models need to be made robust and generalizable so that they can behave consistently with varying network architectures and traffic patterns. Addressing all of these limitations will require interdisciplinary efforts that combine reinforcement learning with other artificial intelligence techniques and cybersecurity frameworks. Additionally, future work should be conducted to explore mechanisms for seamless integration with other existing security infrastructures (e.g., Security Information and Event Management (SIEM) or Intrusion Detection System (IDS)) to provide a broader context for informed decision-making. Advancing these areas will help pave the way for production-ready, secure, and explainable firewall systems powered by reinforcement learning, which are essential for addressing modern network security challenges.

REFERENCES

1. Abbas, N. N., Ahmed, T., Shah, S. H. U., Omar, M., & Park, H. W. (2019). Investigating the applications of artificial intelligence in cyber security. *Scientometrics*, *121*, 1189-1211. https://link.springer.com/article/10.1007/s11192-019-03222-9

2. Bangash, Y. A., Zeng, L. F., & Feng, D. (2017). MimiBS: Mimicking base-station to provide location privacy protection in wireless sensor networks. *Journal of Computer Science and Technology*, *32*, 991-1007. https://link.springer.com/article/10.1007/s11390-017-1777-0
3. Brass, I., & Sowell, J. H. (2021). Adaptive governance for the Internet of Things: Coping with emerging security risks. *Regulation & Governance*, *15*(4), 1092-1110. https://doi.org/10.1111/rego.12343
4. Campazas-Vega, A., Crespo-Martínez, I. S., Guerrero-Higueras, Á. M., Álvarez-Aparicio, C., Matellán, V., & Fernández-Llamas, C. (2023). Malicious traffic detection on sampled network flow data with novelty-detection-based models. Scientific Reports, 13(1), 15446. https://www.nature.com/articles/s41598-023-42618-9
5. Chavan, A. (2023). Managing scalability and cost in microservices architecture: Balancing infinite scalability with financial constraints. Journal of Artificial Intelligence & Cloud Computing, 2, E264. http://doi.org/10.47363/JAICC/2023(2)E264
6. Chavan, A., & Romanov, Y. (2023). Managing scalability and cost in microservices architecture: Balancing infinite scalability with financial constraints. *Journal of Artificial Intelligence & Cloud Computing, 5*, E102. https://doi.org/10.47363/JMHC/2023(5)E102
7. S. K. Gunda, "Machine Learning Approaches for Software Fault Diagnosis: Evaluating Decision Tree and KNN Models," 2024 Global Conference on Communications and Information Technologies (GCCIT), BANGALORE, India, 2024, pp. 1-5, https://ieeexplore.ieee.org/document/10861953
8. Dhanagari, M. R. (2024). MongoDB and data consistency: Bridging the gap between performance and reliability. *Journal of Computer Science and Technology Studies, 6*(2), 183-198. https://doi.org/10.32996/jcsts.2024.6.2.21
9. Dhanagari, M. R. (2024). Scaling with MongoDB: Solutions for handling big data in real-time. *Journal of Computer Science and Technology Studies, 6*(5), 246-264. https://doi.org/10.32996/jcsts.2024.6.5.20
10. Dulac-Arnold, G., Levine, N., Mankowitz, D. J., Li, J., Paduraru, C., Gowal, S., & Hester, T. (2021). Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, *110*(9), 2419-2468. https://link.springer.com/article/10.1007/s10994-021-05961-4
11. Faruque, M. O., Strasser, T., Lauss, G., Jalili-Marandi, V., Forsyth, P., Dufour, C., ... & Paolone, M. (2015). Real-time simulation technologies for power systems design, testing, and analysis. *IEEE Power and Energy Technology Systems Journal*, *2*(2), 63-73. https://doi.org/10.1109/JPETS.2015.2427370
12. Gilbert, T. K., Dean, S., Zick, T., & Lambert, N. (2022). Choices, risks, and reward reports: Charting public policy for reinforcement learning systems. arXiv preprint arXiv:2202.05716.
13. Goel, G., & Bhramhabhatt, R. (2024). Dual sourcing strategies. *International Journal of Science and Research Archive*, 13(2), 2155. https://doi.org/10.30574/ijsra.2024.13.2.2155
14. Greenhalgh, T., Jackson, C., Shaw, S., & Janamian, T. (2016). Achieving research impact through co-creation in community-based health services: literature review and case study. *The Milbank Quarterly*, *94*(2), 392-429. https://doi.org/10.1111/1468-0009.12197
15. Karunamurthy, A., Kiruthivasan, R., & Gauthamkrishna, S. (2023). Human-in-the-Loop Intelligence: Advancing AI-Centric Cybersecurity for the Future. *Quing: International Journal of Multidisciplinary Scientific Research and Development*, *2*(3), 20-43. https://doi.org/10.54368/qijmsrd.2.3.0011
16. Karwa, K. (2023). AI-powered career coaching: Evaluating feedback tools for design students. Indian Journal of Economics & Business. https://www.ashwinanokha.com/ijeb-v22-4-2023.php

17. Karwa, K. (2024). Navigating the job market: Tailored career advice for design students. *International Journal of Emerging Business*, *23*(2). https://www.ashwinanokha.com/ijeb-v23-2-2024.php
18. S. K. Gunda, "Analyzing Machine Learning Techniques for Software Defect Prediction: A Comprehensive Performance Comparison," 2024 Asian Conference on Intelligent Technologies (ACOIT), KOLAR, India, 2024, pp. 1-5. https://ieeexplore.ieee.org/document/10939610
19. Kumar, A. (2019). The convergence of predictive analytics in driving business intelligence and enhancing DevOps efficiency. International Journal of Computational Engineering and Management, 6(6), 118-142. Retrieved from https://ijcem.in/wp-content/uploads/THE-CONVERGENCE-OF-PREDICTIVE-ANALYTICS-IN-DRIVING-BUSINESS-INTELLIGENCE-AND-ENHANCING-DEVOPS-EFFICIENCY.pdf
20. Liu, Q., Hagenmeyer, V., & Keller, H. B. (2021). A review of rule learning-based intrusion detection systems and their prospects in smart grids. *Ieee Access*, *9*, 57542-57564. https://doi.org/10.1109/ACCESS.2021.3071263
21. MacGlashan, J., Ho, M. K., Loftin, R., Peng, B., Wang, G., Roberts, D. L., ... & Littman, M. L. (2017, July). Interactive learning from policy-dependent human feedback. In *International conference on machine learning* (pp. 2285-2294). PMLR. https://proceedings.mlr.press/v70/macglashan17a
22. Nguyen, T. T., Nguyen, N. D., & Nahavandi, S. (2020). Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications. *IEEE transactions on cybernetics*, *50*(9), 3826-3839. https://doi.org/10.1109/TCYB.2020.2977374
23. Nyati, S. (2018). Transforming telematics in fleet management: Innovations in asset tracking, efficiency, and communication. International Journal of Science and Research (IJSR), 7(10), 1804-1810. Retrieved from https://www.ijsr.net/getabstract.php?paperid=SR24203184230
24. Peltomäki, J. (2023). *LiDAR Place Recognition with Image Retrieval* (Doctoral dissertation, University of Tampere, Finland). https://trepo.tuni.fi/bitstream/handle/10024/146010/978-952-03-2788-0.pdf?sequen
25. Raju, R. K. (2017). Dynamic memory inference network for natural language inference. International Journal of Science and Research (IJSR), 6(2). https://www.ijsr.net/archive/v6i2/SR24926091431.pdf
26. Ramesh, D., Harshith, R. P. V., Mahendrakar, S. G., Srinivasan, K., & Kumar, N. P. (2024). Exploring Contemporary Perspectives on the Implementation of Firewall Policies: A Comprehensive Review of Literature. *Indiana Journal of Multidisciplinary Research*, *4*(3), 218-222. https://doi.org/10.5281/zenodo.12679953
27. Rathore, M. M. U., Paul, A., Ahmad, A., Chen, B. W., Huang, B., & Ji, W. (2015). Real-time big data analytical architecture for remote sensing application. *IEEE journal of selected topics in applied earth observations and remote sensing*, *8*(10), 4610-4621. https://doi.org/10.1109/JSTARS.2015.2424683
28. S. K. Gunda, "A Deep Dive into Software Fault Prediction: Evaluating CNN and RNN Models," 2024 International Conference on Electronic Systems and Intelligent Computing (ICESIC), Chennai, India, 2024, pp. 224-228, https://ieeexplore.ieee.org/document/10846549
29. Sardana, J. (2022). The role of notification scheduling in improving patient outcomes. *International Journal of Science and Research Archive*. Retrieved from https://ijsra.net/content/role-notification-scheduling-improving-patient
30. Scherer, W. T., Adams, S., & Beling, P. A. (2018). On the practical art of state definitions for Markov decision process construction. *IEEE Access*, *6*, 21115-21128. https://doi.org/10.1109/ACCESS.2018.2819940
31. Singh, V. (2022). Visual question answering using transformer architectures: Applying transformer models to improve performance in VQA tasks. Journal of Artificial Intelligence and Cognitive Computing, 1(E228). https://doi.org/10.47363/JAICC/2022(1)E228

32. Singh, V., Doshi, V., Dave, M., Desai, A., Agrawal, S., Shah, J., & Kanani, P. (2020). Answering Questions in Natural Language About Images Using Deep Learning. In *Futuristic Trends in Networks and Computing Technologies: Second International Conference, FTNCT 2019, Chandigarh, India, November 22–23, 2019, Revised Selected Papers 2* (pp. 358-370). Springer Singapore. https://link.springer.com/chapter/10.1007/978-981-15-4451-4_28

33. Sukhadiya, J., Pandya, H., & Singh, V. (2018). Comparison of Image Captioning Methods. *INTERNATIONAL JOURNAL OF ENGINEERING DEVELOPMENT AND RESEARCH*, *6*(4), 43-48. https://rjwave.org/ijedr/papers/IJEDR1804011.pdf

34. Vihervaara, J., & Monnonen, M. (2024). ARTIFICIAL INTELLIGENCE IN MODERN FIREWALLS. https://trepo.tuni.fi/bitstream/handle/10024/161576/AhmadWajeh.pdf?sequence=2

35. Vouros, G. A. (2022). Explainable deep reinforcement learning: state of the art and challenges. *ACM Computing Surveys*, *55*(5), 1-39. https://dl.acm.org/doi/abs/10.1145/3527448