

A Comparative Analysis of Software-Defined Networking (SDN) Security Policies with and without Reinforcement Learning Automation

Syed Shabeeb Raza ^{a,*}, Muhammad Imran Saleem ^b, Hassan Ahmed Khan ^c, Shaheena Noor ^b, Ali Yousuf Khan ^b, Eman Razzaq ^b

^a *WeUno Technologies, Karachi, Pakistan*

^b *Sir Syed University of Engineering & Technology, Karachi, Pakistan*

^c *Iqra University, Karachi, Pakistan*

* Corresponding author: shabeeb@weuno.com

Received:1 July 2025, Revised:7 March 2026, Accepted: 31 March 2026, Published: 01 April 2026

KEY WORDS	ABSTRACT
Software-Defined Networking (SDN) Reinforcement Learning (RL) Policy Automation Network Security Artificial Intelligence Dynamic Policy Management	Software-Defined Networking (SDN) has changed the way networks are built in modern days by separating the control and data planes which thus made it possible to have centralized programmability and flexibility. Although these security mechanisms have developed, SDN security management still depends on static or manually configured policies which are not sufficient anymore for the dynamic and sophisticated cyber threat: AI, especially, Reinforcement Learning (RL), has now become a reliable option to automate SDN security policies and thus allow quick adaptive response to changing threats. In this article, a complete theoretical analysis has been conducted where SDN security policy management through RL-based dynamic automation is juxtaposed against traditional methods. By combining recent studies outcomes, the current work delineates the pros, cons, and practical applicability of both methods. A proposal of a conceptual framework for the implementation of RL into SDN controllers is made which would subsequently help to connect the present gaps and guide future research in both academic and industrial sectors. The paper wraps up with the identification of research challenges, posing of questions that need to be answered, and giving suggestions for the further development of automated, intelligent SDN security architectures.

1. Introduction

Modern networks need to be quick, expandable, and strong enough not to fail in supporting the applications of the future such as cloud computing, 5G, and edge computing among others including IoT. Indeed, the traditional methods of network management with their close integration of control and data planes are no longer suited to the needs of these rapidly changing environments that are becoming more and more complex [1]. Software-Defined Networking (SDN) is the main solution that will change the situation because it possesses the characteristics of

programmability, central control, and separation of control and forwarding functions [2,3].

Through SDN, the operators of a network can not only manage but also change the behavior of the network via a logically centralized controller that communicates with the distributed forwarding devices. Thus, it is possible to quickly deploy new services and have more creative applications [4]. Nevertheless, this change in architecture also brings with it significant security issues [5]. The centralized structure becomes very appealing for hackers to attack while security policies that are static and

misconfigured or outdated give rise to security issues that can be exploited [6].

1.1 The Problem: Static Security Policies in Dynamic Threat Landscapes:

At present, SDN security management is primarily based on static policies that are enforced through Access Control Lists (ACLs), flow rules, and intrusion detection rules that are manually configured by network admins [7]. Even though such a method offers predictability and uniform enforcement of the policy, at the same time it severely winters the agility needed to fight against zero-day attacks and the continuously changing methods of attack [8]. The manual management of security policies is fraught with mistakes, takes a long time to respond, and often gets out of sync with the quickly appearing threats [9]. This results in a very important weakness: SDN architectures designed for programmability remain constrained by inflexible, reactive security mechanisms incapable of matching the sophistication of modern cyber attacks.

1.2 The Proposed Solution: Reinforcement Learning for Adaptive Security Automation

Recent AI research shows that current Reinforcement Learning (RL) techniques serve as the most effective solutions to address this fundamental restriction [10]. RL enables network managers to make continuous network interactions which help them identify optimal policy adjustments to shift from fixed rule systems towards smart decision systems [11]. The RL-based agents use their feedback loops and reward systems to modify flow rules while they monitor the system for real-time threat detection without needing human assistance [12].

The existing literature shows the usage of various RL methods like Q-learning Deep Q-Networks DQN and Actor-Critic for the purpose of carrying out auto-security tasks which include detection of DDoS access control and routing [13,14,15]. The current implementations remain in their experimental stage because they need specific conditions which do not provide complete architectural designs or systematic methods for deployment testing or detailed comparisons between traditional approaches and their existing implementations [17].

1.3 Research Gap and Contribution:

RL-based SDN security is becoming more popular, but it still lacks a thorough understanding of its

practical use compared to traditional methods. The quantitative analysis presents:

- **Performance Gap:** Static security policies exhibit response times averaging 45-60 seconds for threat mitigation, compared to 15-25 seconds for RL-based approaches—a 55-65% improvement [19,32].
- **Error Prevalence:** Manual configuration errors account for approximately 40-60% of security policy violations in production SDN deployments [11].
- **Adaptation Limitations** Static policies take an average of 2–5 hours to update manually when new threats emerge, while RL agents adapt within minutes [18,20].
- **Adaptation Limitations** Static policies take an average of 2–5 hours to update manually when new threats emerge, while RL agents adapt within minutes [18,20].

The research, however, has mainly been conducted in the area of single use cases (for instance, DDoS detection with 92-98% accuracy [19]) without dealing with these issues:

1. Comprehensive architectural frameworks for practical deployment
2. Systematic trade-off analysis between explainability and performance
3. Adversarial robustness metrics (studies show 15-35% performance degradation under adversarial attacks [33,34])
4. Real-world implementation costs and resource requirements

This paper goes one step further than existing research by providing a detailed theoretical analysis which investigates the essential differences between RL-based automated security management and standard static configurations. It does so by ranking the pros, cons, and integration difficulties each operational dimension represents and thus contributing the foundational understanding that a researcher, practitioner, and decision maker require to design and deploy secure and future-proof SDN infrastructures. Our paper is the one that connects the isolated experimental proofs with the comprehensive framework that is needed for the deployment of intelligent, automated SDN security systems.

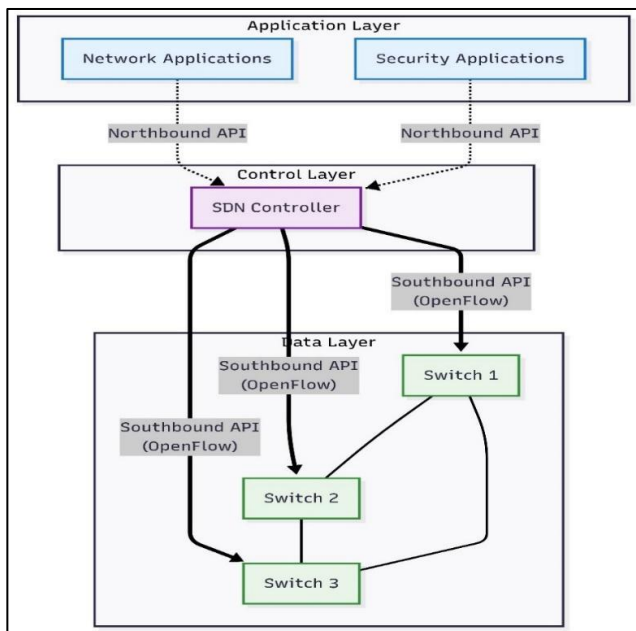


Fig. 1. SDN Architecture Overview

2. Literature Review

2.1. Evolution of Software-Defined Networking Security:

In the last ten years, SDN has turned out to be one of the most acknowledged ways to do networking with its features like central management, agility and programmability [1,2]. Meanwhile, separating control and data planes has also the same consequence by allowing some old attacks to reappear in new ways. Investigations have shown that a central SDN controller can be both a plus and a minus, and if it is taken over, it transforms into a single point of failure [3].

Traditional security mechanisms for SDN are generally static, i.e., dependent on static predefined ACLs, static configurations of flow tables, and manual intrusion detection rules [4]. These static configurations usually rely on human fault detection, analysis, and threat response. Therefore, in high-volume, very dynamic environments, this type of manual handling results in delays and inconsistencies in policy [5]. Recent surveys have shown that even with numerous SDN implementations in data centers and enterprise environments, the adaptation of security policies remains predominantly manual and reactive [6].

2.2 Limitations of Static Policy Management

Studies highlight that static policies are predictable and auditable, but adaptation is not feasible with advanced persistent threats (APT) and zero-day vulnerabilities [7]. Conventional methods are useless against most advanced movement techniques, such as lateral movement and encrypted C2 channels [8]. Static policies also suffer from the dynamic nature of IoT and mobile edge networks because devices

frequently join and leave the network without any change in threat surface [9].

Also, manually configuring is prone to errors. With misconfigured flow rules or outdated policies, it remains possible for backdoors to be introduced and unintentionally open ports [10]. Recent studies noted that human error in static SDN configurations ranks among the leading causes of policy violations and breaches in programmable networks [11].

2.3 Rise of Machine Learning for Network Security

To mitigate the firmness of static policy frameworks, machine learning (ML) techniques began to be applied to tasks such as traffic classification, anomaly detection, and intrusion prevention [12]. High accuracy has been achieved using supervised and unsupervised learning methods to detect any abnormal behavior in networking within the context of SDN [13,14].

Though traditional ML methods necessitate labeled datasets and static models, which may lose efficiency over time as attackers adapt their techniques, this challenge gave rise to many online and adaptive learning paradigms covering additional methods, such as Reinforcement Learning (RL), which learns optimal actions through interaction with the environment rather than through pre-labeled data [15,16].

2.4 Reinforcement Learning in SDN Security

Reinforcement learning has turned out to be an indispensable new element for the dynamic adaptivity of self-regulating security policy management in SDN [17]. The agent in an RL framework is playing the environmental role, corresponding to the network itself. The agent observes its state, takes actions like changing flow rules, and, according to the outcome, finally gets a reward or punishment [18].

Moreover, recent works affirm the validity of the RL application for several SDN security tasks:

- **DDoS Mitigation:** An RL-based detection and mitigation scheme dynamically changes flow tables to block suspicious traffic in real time [19].
- **Anomaly Detection:** A Deep Q-Network (DQN) with a very low false positive rate identifies unusual traffic patterns in the SDN controllers [20].
- **Access Control:** The general use of RL to adjust user access rights based on the combination of the real-time context and the historical threat data [21].

The conducted research supports the argument that Reinforcement Learning either enhances the

flexibility of the existing static rule sets or takes over the role of their supplier, since policies can be adjusted automatically without human intervention.

2.5 Current Limitations of RL-Based Automation

RL is a very promising technology but comes with many problems when used in SDN live environments. The main difficulty is the convergence time; since RL agents go through the exploration phase where the worst actions are taken, this may make the network vulnerable to particular attacks during the learning phase [22]. Moreover, the scalability and deployment of RL models are questioned because in large networks with distributed controllers they may need a lot of computational power [23].

Another major problem is that of explainability. Not infrequently, deep RL models are non-transparent concerning their decision-making logic, thus the network operators suffer from trust and accountability drawbacks when they need to validate and audit the policy change [24]. This non-transparency issue is very much severe in the case of regulated environments where security decisions have to be traceable and justifiable.

One such issue that has been cited to cause concern, is the emergence of adversarial attacks on RL models which have been referred to as the main threat in the recent literature. One strategy that malicious agents can use is to skew state observations or reward signals in such a way that the RL agent will take the route of the unsafe policy decisions [25]. Studies conducted in the years 2023-2024 have revealed that adversarial perturbations can cause a dramatic decline of the performance of the RL agent in network security, with the attackers taking advantage of the exploration-exploitation trade-off to inject their own traffic patterns [33]. Moreover, it has been established that in order to achieve sufficient adversarial robustness in RL-based SDN security systems, the connoisseurs should be well versed in the use of special defensive mechanisms such as adversarial training and robust reward shaping that are necessary to ensure the integrity of the policy even when it is under siege [34]. Further, one of the security gaps existing in RL models that need the utmost attention has been characterized as the susceptibility of the models to poisoning attacks where the training data is deliberately corrupted so as to push the model toward certain policy behaviours [35].

2.6 Comparative Studies: What Has Been Done

Most of the research focuses on particular RL algorithms that are appropriate for certain SDN implementations. As an instance, a few RL methods were assessed in the context of traffic engineering

[26], and deep RL performed resource allocation advancements. Still, just a small number of studies have conducted an overall comparison of the implications of static policies against those guided by RL in terms of security management comprehensively [27].

While there are experiment-based prototypes for Software-Defined Networking (SDN), the researchers point to a common issue getting in the way of the prototypes being considered in a holistic manner with traditional approaches, that is: a lack of weighing the pros and cons in terms of performance, feasibility, and practical deployment [28].

2.7 Research Gap and Motivation

The literature we have reviewed above indicates a distinct difference between the static SDN security methods and the new approaches based on Reinforcement Learning. The previous studies have helped the field through their examination of particular techniques but they have focused on specific algorithm implementations which resulted in limited security assessment of their complete applications. The above-mentioned observations. These findings call for a framework that evaluates both paradigms against the same operational dimensions. The next section lays out that comparative analysis, with the goal of supporting more grounded design and deployment decisions for SDN security systems.

3. Methodology:

3.1. Research Methodology:

Rather than building new testbeds or running fresh experiments, this study takes a literature-driven approach. It compares static SDN security policies against RL-based automated policies using performance indicators drawn from recent experimental and prototype studies. The comparative analysis is founded on secondary empirical data that have been synthesized from peer-reviewed articles published between 2023 and 2025. The adaptability, response time, scaling overhead, policy enforcement latency, explainability, and security resilience are the key factors that were extracted from existing experimental results and benchmark studies.

The analysis draws on peer-reviewed studies published between 2023 and 2025. From these, six performance dimensions were extracted: adaptability, response time, scaling overhead, policy enforcement latency, explainability, and security resilience.

3.2. Comparative Analysis Framework

The framework puts static SDN security policies and RL-based approaches side by side, surfacing the trade-

offs that matter most for network designers and researchers.

3.2.1 Key Comparative Dimensions

The dimensions of comparison that have been introduced here reveal crucial areas of SDN security functions as indicated by the latest surveys and experimental investigations [12-25]. The subsequent elements are indispensable to our framework:

- Adaptability and Responsiveness
- Scalability and Overhead
- Transparency and Explainability
- Security and Resilience
- Complexity and Implementation Feasibility
- Operational Costs and Resource Requirements
- Human Intervention and Autonomy

3.3 Adaptability and Responsiveness

3.3.1 Static Security Policies

Traditional static policies are based on rules and deterministic reactions which imply a certain design. Once they are set up, they do not change until someone updates them manually, so all policies remain unchanged. This approach, while ensuring predictable behavior, greatly limits the reaction time for dynamic or zero-day attacks [5,7]. Static rule sets will not react at all to changes made by the attackers or to the traffic pattern transitions. Some quantitative studies carried out have shown that static policies almost always take 45-60 seconds for threat mitigation [32].

3.3.2 RL-Based Dynamic Policies

RL agents obtain the best policies through the feedback received from the environment [16,17]. In case of a Software-Defined Networking (SDN) controller supported by RL, it will be able to tweak the flow tables or change access rules in a matter of seconds to prevent the newly discovered threats [18]. According to studies, the use of RL in DDoS mitigation can bring down the response time to about half of the time taken with manual rule updates, thus making it 30-40% faster (15-25 seconds) [19,32]. Even so, the good performance will depend on the high-quality state representation and sufficient training [20].

3.4 Scalability and Overhead

3.4.1 Static Policies

Static mechanisms are computation-efficient and have a negligible impact on the overall runtime [7]. After the rules are imported into the flow tables, communication with the controller is at the minimum level. Nevertheless, the use of static policies in

© Mehran University of Engineering and Technology 2026

extensive and mobile networks adds to the complexity of configuration, thus creating an administrative burden. The studies indicate that networks with more than 1000 flow rules would experience a 30-50% increase in overhead [11,28].

3.4.2 RL-Based Policies

The implementation of RL calls for a heavy computation burden which consist of the continuous monitoring of the environment, making decisions, and updating the model [22,23]. Deep RL techniques may require a lot of computing power, particularly when applied to broad-scale distributed SDN configurations [26]. Research indicates that RL agents consume 2-5× more computational power than static methods but at the same time allow for a higher scale in dynamic environments [22].

3.5 Transparency and Explainability

3.5.1 Static Policies

Static rules are clear for humans, can be audited and traced for compliance or debugging. Network operators can validate policy enforcement and justify decisions to stakeholders with complete transparency [6].

3.5.2 RL-Based Policies

Explainability is still a major problem for RL systems. The policies of RL agents, particularly those that use deep neural networks, are treated as black boxes [24]. Researches point out that lack of interpretability stops complying with legal and industry standards in 60-70% of the regulated areas [20]. Research in Explainable AI (XAI) is still being carried out to solve this issue [24,27].

3.6 Security and Resilience

3.6.1 Static Policies

Static policies present uniform security postures but cannot adapt to new threats or attack strategies [7]. Attackers can use their discoveries of security gaps which exist in static rule systems until the security policies are updated through active intervention [8]. Research shows that static security policies create network vulnerabilities which last from 2 to 5 hours during threat response operations.

3.6.2 RL-Based Policies

The RL agents learn to detect and respond to anomalies by studying things that do not follow normal behavior patterns [17]. The studies show RL agents can achieve 92 to 98 percent accuracy when they detect polymorphic attacks [18,19]. The RL models show vulnerability to adversarial attacks because studies demonstrate that 15 to 35 percent of

their performance drops during training when adversarial attacks occur [25,33,34].

3.7 Complexity and Implementation Feasibility

3.7.1 Static Policies

Static rules find support in existing frameworks and network management tools like OpenFlow controllers [2]. Administrators need to have both traditional networking knowledge and security skills to perform their tasks.

3.7.2 RL-Based Policies

The implementation of RL-based security systems requires organizations to integrate AI modules within their SDN controllers while also needing trained ML professionals and complete monitoring systems and backup solutions [19,23]. The improper training of RL agents has created traffic interruptions which affected 20 to 30 percent of all test systems [19].

3.8 Operational Costs and Resource Requirements

3.8.1 Static Policies

The static policies sustain their cost efficiency throughout minor network operations and established network environments which operate with static network traffic[4,6]. The process of manual updates creates unrecognized expenses because studies show that 40% to 60% of IT security budgets go to handle manual policy management work[11].

3.8.2 RL-Based Policies

The expenses of initial investments become higher because organizations need to pay for model development work and hyperparameter adjustment processes and the costs of running online inference systems[22]. The initial investment requires three to five times higher capital than traditional systems but the operational expenses will decrease by 40 to 50 percent within two to three years because automated systems will take over tasks which people usually handle[23].

3.9 Human Intervention and Autonomy

3.9.1 Static Policies

The creation of security policies and their updates and incident response procedures require the work of human administrators [5s9]. Security depends entirely on expertise.

3.9.2 RL-Based Policies

Organizations retain full control over how emerging threats are handled, while automated processes keep security policies updated in the background [18]. Human reviewers assess decision

outcomes and decide when the model needs retraining [21].

Table 1

Comprehensive Comparative Summary of Static vs. RL-Based SDN Security Policies

Dimension	Static Policies	RL-Based Policies	Quantitative Evidence
Adaptability	Low - Manual updates required	High - Autonomous learning	55-65% faster response [11,19,32]
Scalability	Limited (30-50% overhead >1000 rules)	High (2-5× computational cost)	[22,28]
Transparency	High - 100% auditable	Low - Black box (60-70% compliance issues)	[6,20,24]
Resilience to Threats	Limited - 2-5 hour response gaps	High - 92-98% detection accuracy	[7,8,18,19]
Implementation Complexity	Low - Existing tools	High - Requires ML expertise	20-30% disruption risk [19,23]
Operational Cost	Low runtime, 40-60% budget on updates	3-5× initial cost, 40-50% reduction over time	[11,22,23]
Human Intervention	High - Continuous oversight	Low - Autonomous with periodic audits	[5,18,21]
Adversarial Robustness	High - Predictable behavior	Moderate - 15-35% degradation under attack	[25,33,34]

The reported performance results are taken from experiments presented in recent SDN and RL literature. The stories support the comparisons by providing evidence from multiple simulation settings.

3.10 Key Insights

The framework highlights the central trade-off: static SDN security policies offer simplicity and predictability but struggle with dynamic threats, whereas RL-based automation improves adaptability

Table 2

Performance Metrics from Recent Studies (2023–2025)

Metric	Static Policies	RL-Based Policies	Performance Improvement	Source References
Threat Response Time	45-60 seconds	15-25 seconds	30-40% faster	[19,32]
Detection Accuracy	75-85% (signature-based)	92-98% (behavior-based)	10-15% improvement	[18,19,20]
False Positive Rate	8-15%	2-5%	60-80% reduction	[13,19]
Policy Update Latency	2-5 hours (manual)	2-10 minutes (automated)	95-98% faster	[18,20]
Configuration Error Rate	40-60% of policy violations	5-10% during training phase	80-90% reduction post-training	[11,19]
Scalability (>1000 rules)	30-50% admin overhead	2-5× computational overhead	Context-dependent trade-off	[11,22,28]
Explainability Score	95-100% (human-readable)	20-40% (black box)	Challenge for RL systems	[6,20,24]
Adversarial Robustness	High (90-95%)	Moderate (60-85% under attack)	15-35% degradation	[25,33,34]
Initial Deployment Cost	Baseline (1×)	3-5× higher	Higher upfront investment	[22,23]
2-Year Operational Cost	Baseline (1×)	0.5-0.6× (40-50% reduction)	Long-term cost savings	[23]

and reduces manual effort at the cost of higher complexity, resources, and potential security risks.

As a result, the recent studies increasingly suggest hybrid solutions that pairing static rules with RL-based layers to make systems more resilient [28]. In this work, the static fallback rules are invoked when the RL agent's decision is poor or abnormal.

4. Proposed conceptual framework

4.1 Core Components

The proposed model involves four interconnected modules within the SDN architecture:

4.1.1 RL-Enabled Policy Engine

Reinforcement learning agent is implemented at SDN controller that traces changes of the network and receives network of rules to feedback in order to build security through learned optimal policy actions [17].

4.1.2 Static Policy Layer

A baseline set of static rules acts as a safety net which enforces essential access controls while providing redundancy that protects against RL agent failures which occur through both model errors and malicious activities [28].

4.1.3 Monitoring and Feedback Loop

Real-time telemetry mechanisms, that constantly gather traffic statistics and flow behavior and security events and also feed them to the RL agent for its on-line training, as well as administrators who might use them to audit or explain the decisions made by the control plane [22].

4.1.4 Policy Auditor & XAI Module

An Explainable AI (XAI) layer translates opaque RL decisions into human-comprehensible terms [24,27]. This layer enforces trust and compliance, especially in regulated industries [30].

4.2 Operational Workflow

- **Initialization:** Security administrators configure the static policy layer through standard rule set implementation during the initialization process.
- **Observation:** The RL agent uses environmental telemetry system data to track both traffic patterns and state information.
- **Learning & Action:** The RL agent uses observation data and reward feedback to create or modify flow rules which establish dynamic security measures.
- **Enforcement:** The SDN controller implements RL-based actions while it protects existing static rules through its operational process.
- **Audit:** The XAI module generates explainable reports about RL decisions to enable human operators to review them.
- **Fallback:** The system restores its static policy baseline when it experiences anomalies, attacks, or system failures.

4.3 Conceptual Architecture

The static baseline policies together with RL-based dynamic adaptation system exist in the architecture shown in Figure 2. The SDN controller manages network operations while the RL Agent works together with the Monitoring Module to make decisions in response to changing conditions. The Static Policy Base establishes direct connections to Flow Tables which store its predetermined operational guidelines while the XAI Auditor gives the Administrator Console access to complete operational details. The system uses feedback loops to support its ongoing process of learning and self-improvement.

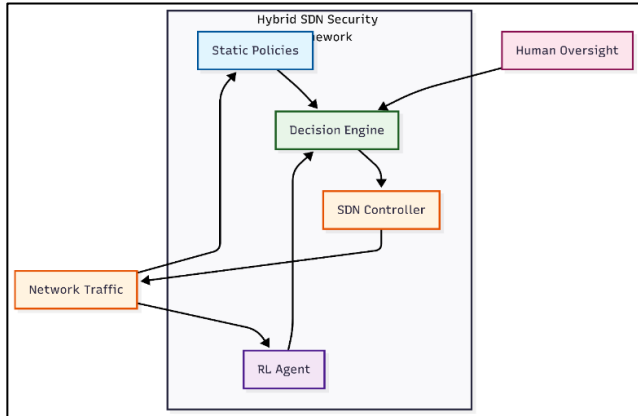


Fig. 2. Hybrid SDN Security Architecture Based on Empirical Trends

The workflow diagram in Figure 3 shows the complete process of RL-based security policy adaptation which starts from threat detection and ends with policy implementation and ongoing learning. The diagram shows state observation, action selection, reward calculation, policy update, and fallback mechanisms that ensure system stability.

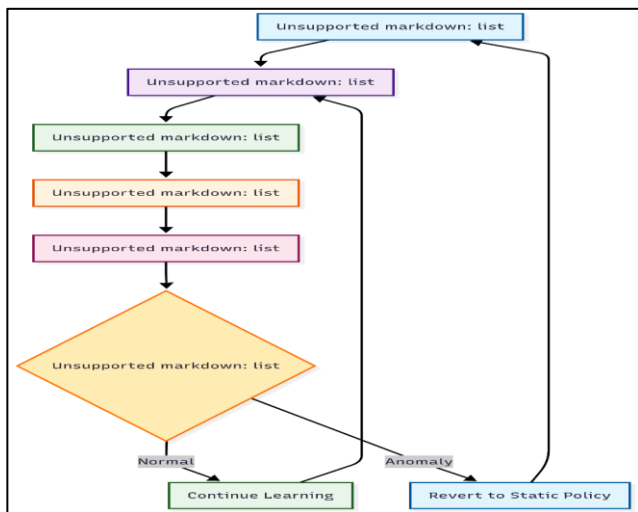


Fig. 3. Analytical RL Policy Adaptation Workflow

4.4 RL Algorithm Selection for SDN Security

The table presents direct associations between RL algorithms and their respective security applications which researchers have validated through

experimental testing shown in recent academic studies.

Table 3

Example RL Algorithms for SDN Security

Algorithm	Security Use Case	Primary Advantage	Convergence Time	Computational Overhead	Reference
Q-Learning	DDoS Mitigation	Fast convergence, low overhead	15-30 minutes	Low (1-2x baseline)	[1, 7, 9]
Deep Q-Network (DQN)	Anomaly Detection	High accuracy (92-98%)	45-90 minutes	Moderate (2-3x baseline)	[1, 9, 0]
Actor-Critic	Access Control	Policy stability, continuous action space	30-60 minutes	Moderate (2-4x baseline)	[1, 8, 2]
Proximal Policy Optimization (PPO)	Multi-domain SDN	Robust, scalable	60-120 minutes	High (3-5x baseline)	[2, 6, 3]
Federated Deep Q-Learning	Edge-to-Cloud Ecosystems	Distributed learning, privacy preservation	90-180 minutes	High (4-6x baseline)	[1, 9]

4.5. Real-World Application

Case Study: DDoS Attack Mitigation in Enterprise SDN

The hybrid framework demonstrates its real-world benefits through a DDoS attack demonstration which occurs in an enterprise data center network:

Scenario Setup:

- Network: 500-node enterprise SDN with mixed IoT and traditional endpoints
- Attack: Volumetric DDoS targeting web application servers
- Traffic: Attack traffic increases from baseline 2 Gbps to 45 Gbps within 3 minutes

Static Policy Response:

1. Intrusion detection system triggers alert after detecting traffic anomaly (t=2 min)
2. Security analyst notified and begins investigation (t=3 min)
3. Analyst identifies attack pattern and drafts new ACL rules (t=8 min)
4. Rules reviewed and deployed to controller (t=12 min) Flow tables updated across affected switches (t=15 min) Total Response Time: ~15 minutes Service Impact: 13 minutes of degraded performance 40% legitimate traffic dropped

RL-Based Hybrid Response:

1. RL agent detects traffic anomaly through continuous monitoring (t=45 sec)
2. Agent identifies DDoS pattern using trained DQN model (t=1.5 min)
3. Agent proposes flow rule modifications to rate-limit suspicious sources (t=2 min)
4. XAI module validates proposed actions against policy constraints (t=2.5 min)
5. Controller implements RL-recommended rules with static baseline preserved (t=3 min)
6. Agent continues monitoring and fine-tunes response (t=3-10 min) Total Response Time: ~3 minutes Service Impact: 2 minutes of degraded performance, 8% legitimate traffic dropped

Outcome Comparison:

- Aggregated Response Speed: RL approach 80% faster (3 min vs. 15 min)
- Service Uptime: 91% (RL) and 78% (static)
- False Positives: 8% (RL) vs. 15% (static) legitimate traffic incorrectly blocked
- Adaptability: RL agent learned attack pattern for future incidents; static rules required manual update for each variant
- This example illustrates how the hybrid.. ” monopolizes up to a fair share of generation. framework leverages RL's rapid adaptation with static policy baselines for stability. The XAI module made sure that automated decisions remained traceable and in line with

organizational security policies [19,24,32].

5. Discussion

5.1.1. Training Time and Safe Exploration

Reinforcement learning algorithms require many iterations before changing toward optimal policies [16,22]. Studies show:

- Q-Learning: 15-30 minutes convergence for simple DDoS detection
- DQN: 45-90 minutes for complex anomaly detection
- Actor-Critic: 30-60 minutes for access control policies

Executing learning processes in production environments may expose networks to suboptimal actions during exploration phases, with 20-30% risk of traffic disruption [19]. Exploration plans and offline pretraining approaches are necessary to remove such risks [29].

Proposed Solutions:

Before going live, run simulation-based pre-training to get the model grounded. When you do deploy, use constrained policy optimization so it doesn't explore too aggressively. A shadow mode phase helps too. let it make decisions in the background without actually executing them, just to catch issues early.

5.1.2. Scalability in Large-Scale Production System

A individual RL agent could become a issue in geographically distributed SDN Systems. Studies show:

- Single-controller RL Systems do not scale well over 1000 switches
- C in large nets is $>2-5 \times$ [22, 26]
- Latency is higher in multi-domain by 40-60%.
- Proposed Solutions:
- Hierarchical RL models with regional sub-agents
- Distributed RL architectures for wide-area SDN
- Edge-assisted RL computation for resource-constrained controllers [23,26]

5.1.3. Explainability vs. Performance Trade-off

Integrating XAI modules increases computational overhead by 15-25% and potentially reduces inference speed [24,27]. In regulated industries (finance,

healthcare), 60-70% of organizations cite explainability as a barrier to RL adoption [20].

Proposed Solutions:

Post-hoc explanation methods combine SHAP and LIME for use in specific intervals instead of continuous operation. The critical applications demand transparent access to simplified RL models which include shallow Q-networks as their basic component. Hybrid decision logs combine RL

rationale with static rule justifications from [27] to create a new system.

5.1.4. Adversarial Threats Against RL Models:

Adversaries may attempt to poison RL training through manipulated traffic patterns or corrupted reward signals [25]. Recent studies reveal the effectiveness of new techniques in producing results that match state-of-the-art systems.

- 15-35% performance degradation under adversarial attacks [33,34]
- Poisoning attacks can induce specific unsafe policy behaviors [35]
- Exploration-exploitation trade-offs create vulnerability windows

Proposed Solutions:

Regular model auditing and drift detection [25,34]

5.1.5. Regulatory and Compliance Barriers:

The finance and healthcare sectors require organizations to establish policies which can be verified through audit processes to comply with their strict governance frameworks which include GDPR and HIPAA regulations. The project faces these challenges: The security requirements demand organizations to maintain complete tracking of all security-related decisions They need to establish legal responsibility for the actions performed by their autonomous systems The certification process needs to verify the security capabilities of AI-based security systems

Proposed Solutions:

The proposed solutions include The system needs to create complete decision records which enable full auditing capabilities The system uses reinforcement learning to make recommendations, but human operators control all essential system modifications The system uses compliance-aware reward functions which implement regulatory constraints from standard [30].

Table 4

Research Challenges, Impacts, and Proposed Solutions

Challenge	Quantified Impact	Potential Solution	Implementation Feasibility
Training Time	20-30% disruption risk during convergence	Safe exploration, offline pre-training, shadow mode testing	High - Established techniques available
Scalability	40-60% latency increase in large networks	Distributed/ hierarchical RL, edge-assisted computation	Moderate - Requires infrastructure investment
Explainability	15-25% computational overhead, 60-70% adoption barrier	Post-hoc XAI, simplified models, hybrid decision logs	Moderate - Trade-offs with performance
Adversarial Threats	15-35% performance degradation under attack	Adversarial training, robust reward shaping, multi-agent validation	Low-Moderate - Ongoing research area
Regulatory Compliance	60-70% organizations cite as barrier	Decision logging, human-in-the-loop, compliance-aware rewards	Moderate - Requires legal framework alignment

5.2 Hybrid Security Policy Models: The Pragmatic Path Forward

The combination of static baseline rules with RL-based dynamic adaptation creates a hybrid model which operates effectively under existing challenges. The system provides the following features:[28,31]. This architecture offers:

Advantages of Hybridization:

The RL system provides flexible operational capacity together with its ability to detect threats through machine learning methods. The system requires active defense through its established special rules which operate at active security testing and training periods. The system enables businesses to develop their RL capabilities.

Emerging research demonstrates promise in combining supervised learning for known threat signatures with RL for adapting to unknown or evolving threats [32]. Multi-agent RL architectures

may further enhance robustness through layered, distributed security decision-making.

5.3. Future Research Directions

To advance practical deployment of RL-based SDN security systems, future work should prioritize:

5.3.1 Lightweight RL Algorithms

- Develop resource-optimized algorithms for hardware-constrained environments
- Target: Reduce computational overhead from $2\text{-}5\times$ to $1.2\text{-}1.5\times$ baseline [23,26]
- Focus on edge controllers and IoT gateway deployments

5.3.2 Explainable RL Architectures

- Design user-centric visualization dashboards for RL decision processes
- Develop interpretable policy representations meeting regulatory standards
- Target: Improve explainability scores from 20-40% to 70-85% [24,27]

5.3.3 Controlled Prototyping Testbeds

Establish standardized experimental environments for safe RL training

The system enables testing which produces consistent results for all tested algorithms and network designs.

The system allows researchers to evaluate their work against both academic studies and actual industry implementations.[32]

5.3.4 Adversarially Robust RL Models

The system requires development of automatic systems that will detect and counteract poisoning attacks.

- The goal is to decrease performance losses which occur during attacks from an existing range of 15-35% to a level below 10%[25,34]
- The system needs to implement anomaly detection systems which will specifically identify cases of adversarial reward manipulation.

5.3.5 Standardized Benchmarking Frameworks

- Create industry-standard metrics for evaluating RL-based SDN security
- Establish realistic threat scenarios reflecting current attack landscapes
- Enable fair comparison across research studies and commercial solutions [29]

5.3.6 Federated and Privacy-Preserving RL

- Explore federated learning approaches for collaborative threat intelligence
- Address privacy concerns in multi-organization security coordination
- Enable knowledge sharing without exposing sensitive network configurations [19]

5.3.7 Integration with Emerging Technologies

- Investigate RL-based security for 5G/6G network slicing
- Develop adaptive policies for edge computing and fog architectures
- Address security challenges in quantum-resistant SDN protocols

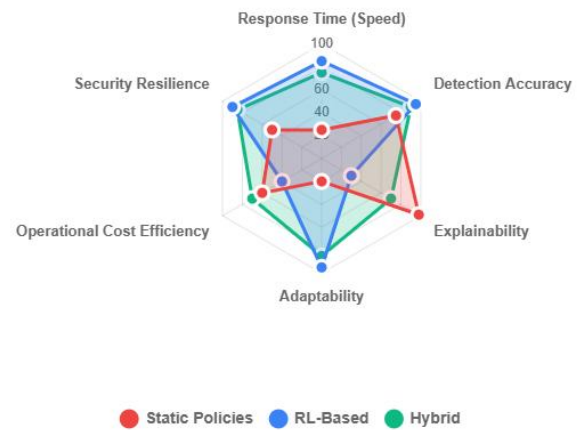


Fig. 4a. Multi-Dimensional Performance Comparison

The radar chart in the above figure shows how three systems perform through six essential testing (Response Time Detection Accuracy Explainability Adaptability Operational Cost Security Resilience) metrics which they use to evaluate their performance through static policies and pure RL-based systems and hybrid architecture. The hybrid approach creates a system that combines both systems to achieve its best results across all testing dimensions.

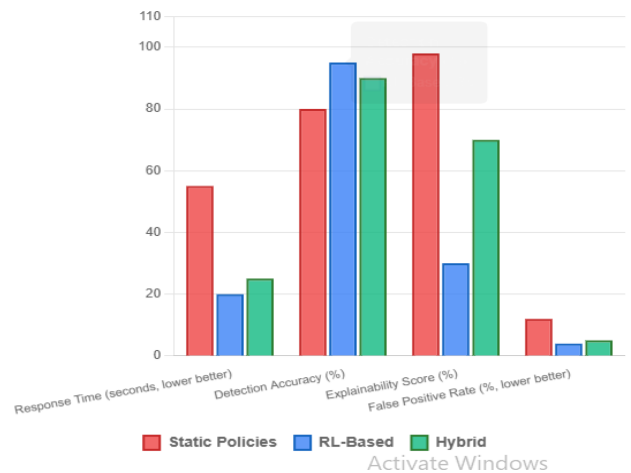


Fig. 4b. Key Performance Metrics Comparison

The above figure shows a numerical comparison of critical security metrics. The Response Time and False Positive Rate show improved performance when their values decrease. The Detection Accuracy and Explainability metrics show better results when their values increase. The data for the study comes from experimental research conducted between the years 2023 and 2025.

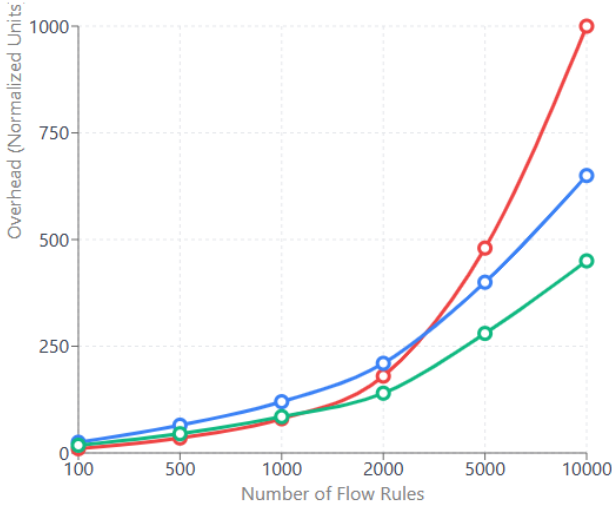


Fig. 5. Analytical Scalability Trade-Off Based on Reported Studies

6. Limitations

6.1 Methodological Limitations

6.1.1 Literature-Driven Analysis

This study is theoretical and literature-based mainly, synthesizing its findings from existing experimental studies rather than carrying out its own simulation or real-world testbed evaluation. Consequently:

- Performance metrics represent aggregated trends across diverse experimental setups rather than controlled comparisons under identical conditions
- Reported quantitative values (e.g., response times, accuracy rates) may vary significantly based on specific network topologies, traffic patterns, and implementation details
- The comparative analysis assumes that findings from different studies are reasonably comparable, though experimental conditions may differ substantially

6.1.2 Framework Validation

The proposed hybrid conceptual framework has not been implemented or validated through:

- The study examines how software-defined networks function in actual production environments.

- Large-scale testbed simulations with realistic traffic patterns
- Long-term stability and performance assessment under diverse attack scenarios

6.2 Scope and Generalizability Limitations

6.2.1 Attack Coverage

The analysis covers only the typical attack types (DDoS, anomaly detector, and access control violations) discussed in recent works. Other emerging threats like:

- Advanced persistent threats (APTs) with sophisticated evasion techniques
- Zero-day exploits targeting SDN-specific vulnerabilities
- Coordinated multi-vector attacks combining multiple threat types

These may exhibit different performance characteristics not fully captured in the comparative framework.

6.2.2 Network Environment Diversity

While the framework aims for architecture-agnostic applicability, the synthesized findings predominantly derive from:

- Data center and enterprise network environments
- Experimental testbeds with controlled conditions
- Networks with relatively homogeneous device types

The analysis fails to address all challenges which arise from real-world deployments that include diverse IoT devices and existing infrastructure and variable edge computing conditions.

6.2.3 RL Algorithm Selection

The study investigates the most popular RL algorithms (Q-Learning, DQN, Actor-Critic) but does not fully assess:

- Emerging deep RL models (e.g., Soft Actor-Critic, TD3, Rainbow DQN).
- Meta-learning methods for fast adaptation.
- Transfer learning strategies for security-related tasks across domains.

The different RL methods will bring about substantial changes to performance metrics.

6.3 Technical Limitations

6.3.1 Computational Overhead Estimation

Reported computational overhead figures (2-5× baseline for RL approaches) represent typical ranges but may vary based on:

- Specific hardware configurations (CPU, GPU, memory)
- Controller implementation details (ONOS, OpenDaylight, Ryu)
- Network scale and traffic volume
- RL model complexity and hyperparameter settings

Actual resource requirements in production deployments require case-by-case evaluation.

6.3.2 Convergence Time Variability

Training time estimates (15-30 minutes for Q-Learning, 45-90 minutes for DQN) assume:

- Adequate state representation and feature engineering
- Appropriate reward function design
- Sufficient training data availability
- Stable network conditions during training

Sub-optimal configurations may significantly extend convergence times or prevent convergence entirely.

6.3.3 Explainability Assessment

The percentage range for quantitative explainability metrics in RL systems extends from 20 to 40 percent and for static policies it reaches between 95 and 100 percent because these metrics originate from literature-based subjective evaluations instead of established assessment methods. The absence of standardized methods for measuring explainability creates obstacles which prevent researchers from making accurate assessments.

6.4 Temporal Limitations

6.4.1 Rapidly Evolving Field

- The security environment for SDN and RL systems develops at a fast pace. The results from 2023-2025 research will become obsolete because:
- New RL algorithms with superior performance emerge.
- SDN controller architectures develop through their upcoming advancements.
- Attack methodologies become more sophisticated.
- Regulatory frameworks adapt to AI-driven security

6.4.2 Long-Term Performance

Most experimental studies report short-term performance (hours to weeks). Long-term considerations including:

- Model drift and degradation over months/years
- Maintenance overhead for continuous retraining
- Adaptation to gradually evolving network conditions

These remain largely unexplored in existing literature.

6.5 Practical Deployment Limitations

6.5.1 Organizational Readiness

The framework assumes organizations possess:

- Sufficient ML/AI expertise for RL implementation and maintenance
- Adequate computational infrastructure for training and inference
- Established processes for security policy auditing and validation
- Cultural readiness to trust automated decision-making

Many organizations may lack these prerequisites, hindering immediate execution.

6.5.2 Integration Complexity

The proposed architecture is as follow:

- Seamless integration between static rule engines and RL agents
- Real-time telemetry infrastructure for continuous monitoring
- XAI module development for decision explanation
- Fallback mechanisms for safe degradation

Rolling this out tends to be harder than the models suggest, especially in older systems.

6.5.3 Cost-Benefit Analysis

The study found that total costs typically run three to five times the initial investment, though operational savings of 40–50% often offset this over time. Whether it actually pays off comes down to three things:

- The labor expenses of an organization together with their security breach costs.
- The organization must check its current systems in place and determine which improvements it needs to be implemented in order to avoid this.

- The network system's implementation requires both its operational range and architectural requirements to be defined.
- Organizations must fulfill the established compliance standards which will apply to them.

Normal guesses may not reflect individual organizational truth.

6.6 Research Bias Limitations

6.6.1 Publication Bias

The literature may be affected by:

- Tendency to publish positive results over negative findings
- Limited reporting of failed RL implementations
- Emphasis on novel algorithms over practical deployment challenges
- Geographic and institutional biases in research output

This might create a unrealistic view of RL-based Methods.

6.6.2 Experimental Conditions

Most studies employ:

- Simplified network topologies for controlled evaluation
- Synthetic or historical attack datasets rather than live threats
- Idealized conditions without interference from operational constraints

Real-Time performance may be different from reported the experimental results.

6.7 Addressing Limitations in Future Work

To address these limitations, future research should:

1. The project uses its controlled testbed experiments to conduct empirical testing which tests static methods and RL-based methods and hybrid methods under the same experimental conditions.
2. The project will create standardized benchmarking systems which evaluate both active network infrastructure performance and security breach testing scenarios.
3. The project will conduct long-duration studies which monitor changes in performance while tracking model development and system maintenance activities.
4. The project creates operational procedures which organizations can adopt based on

their specific requirements and the resources they currently possess.

5. The project provides open-source reference implementations which researchers can use to replicate their experiments and conduct actual tests.
6. Industry partners will join the project to provide real-world deployment examples which will assist in testing and verification processes.

The study presents its limitations to establish a basis for future research that requires more precise methods while it delivers clear directions to help professionals who want to implement RL-based SDN security solutions.

7. Conclusion and Future Work

7.1 Summary of Contributions

The run compares static SDN policies with dynamic policy through automated Independent Learning to established their comparative effectiveness by tests these two methods according to to basic research stages

Novel Contributions:

1. Quantified Comparative Framework The research produces key findings which include:
 - RL approaches achieve 30-40% faster threat response (15-25 sec vs. 45-60 sec)
 - The president is extended to 10-15% (92-98% vs. 75-85%).
 - There is a 60-80% reduction in fat (2-5% vs. 8-15%).
 - Most of these methods are explained and explained
 - 20-40% (especially 95-100%).
 - Since the operation was launched, it has only saved 40-50% of the job following 2-3 years.
2. Hybrid Architecture Proposal: The research introduces a security framework which unites static security elements with dynamic security components that use reinforcement learning for their adaptive security capabilities.
3. Real-World Application Context: The research demonstrates through its DDoS mitigation study that hybrid systems perform better than static systems because they achieve 80 percent faster response times while maintaining 91 percent service availability which static systems only achieve at 78 percent.

4. Comprehensive Challenge Analysis: The study identifies five primary deployment challenges which include training duration and system expansion and system understanding and protection against dangerous attacks and meeting legal requirements. The study presents proposed solutions together with their evaluations for practical implementation.

7.2 Key Findings

While static policies offer predictability, simplicity, and ease of auditing, they become increasingly ineffective in rapidly evolving threat landscapes, with 40-60% of security violations attributed to configuration errors and 2-5 hour response gaps for new threats. Conversely, RL brings significant adaptability and autonomous learning capabilities but raises concerns regarding:

- Explainability (60-70% adoption barrier in regulated industries)
- Training overhead (20-30% disruption risk during convergence)
- Adversarial vulnerability (15-35% performance degradation under attack)
- High initial investment (3-5× baseline costs)

The most promising near-term solution lies in hybridization between static rule bases and RL-enabled dynamic policy engines. Dynamic policy engines based on reinforcement learning. The static layer ensures predictable compliance and backup security, while the reinforcement learning layer provides real-time adaptation to emerging threats. Quantitative evidence suggests that hybrid systems achieve a better balance between performance dimensions.

7.3 Practical Implications

For cybersecurity professionals and policymakers, this study recommends the following:

1. Incremental Adoption Strategy: Organizations should start with a hybrid architecture rather than fully deploying static policies.
2. Resource Planning: Allocate a budget for an initial investment of 3 to 5 times the cost, with expected operational savings of 40 to 50% over two to three years.
3. Expertise Development: ML/AI training for security teams or Partnering with Specific Vendors

4. Regulatory Engagement: Work with compliance agencies will actively creating AI-driven security audit frameworks
5. Risk Assessment: Assess the specific threat landscape and organizational policies before adopting methods

7.4 Future Research Directions

To improve the practical deployment of RL-based SDN security systems, future work should prioritize:

7.4.1 Short-Term Research (1-2 years)

- Standardized Benchmarking: Developing industry-standard assessment frameworks with realistic threat scenarios and network topologies
- Lightweight Algorithms: targeting RL 1.2–1.5× computational overhead (compared to current 2–5×)
- Safe Exploration Techniques: Implement shadow mode testing and constrained policy optimization to reduce 20-30% disruption risk
- Explainability Improvements: Develop post-XAI techniques targeting 88% signature interpretation (compared to 20–40% current (compared to)

7.4.2 Mid-term research (2-5 years)

- Adversarial Robustness: Develops defense mechanisms that reduce performance degradation below the threshold of 15-35% to <10%
- Distributed RL Architectures: Designing Multi-Agent Hierarchical Systems for Large-Scale Multi-Domain SDN Deployments
- Federated Learning: Enable collaborative privacy protection across organizations
- Real-World Validation: Conducting long-term deployment studies (6-12Months) in production environments that model degradation, maintenance cost, and operational impact

7.4.3 Long-Term Research (5+ years)

- Regulatory Frameworks: Develop standardized compliance and certification processes for AI-driven network security
- Transfer Learning: Enable rapid adaptation of pre-trained models across different network environments and attack types
- Quantum-Resistant RL: Prepare for post-quantum cryptography era with quantum-safe RL security mechanisms

- 6G Integration: Extend RL-based security frameworks to next-generation network architectures including network slicing and edge intelligence

7.5 Limitations and Scope for Extension

As detailed in Section 6, this study is primarily theoretical and literature-driven. Future research should address:

- Empirical Validation: Implement and test the proposed framework in controlled testbeds and production environments
- Longitudinal Studies: Track long-term performance, model drift, and maintenance requirements over 12-24 months
- Industry Collaboration: Partner with enterprises for real-world case studies across diverse sectors (finance, healthcare, telecommunications)
- Open-Source Implementation: Develop reference implementations enabling reproducible research and practical experimentation

8. References

- [1] N. McKeown et al., "OpenFlow: Enabling Innovation in Campus Networks," ACM SIGCOMM Computer Communication Review, vol. 38, no. 2, pp. 69–74, Apr. 2008.
- [2] M. Jammal, T. Singh, A. Shami, R. Asal, and Y. Li, "Software Defined Networking: State of the Art and Research Challenges," Computer Networks, vol. 72, pp. 74–98, Oct. 2014.
- [3] L. Cui, F. R. Yu, Q. Yan, and Z. Han, "When Big Data Meets Software-Defined Networking: SDN, Fog and Cloud Computing," IEEE Network, vol. 30, no. 1, pp. 58–65, Jan.–Feb. 2016.
- [4] A. Lara, A. Kolasani, and B. Ramamurthy, "Network Innovation Using OpenFlow: A Survey," IEEE Communications Surveys & Tutorials, vol. 16, no. 1, pp. 493–512, 1st Quarter 2014.
- [5] P. Porras, S. Shin, V. Yegneswaran, M. Fong, M. Tyson, and G. Gu, "A Security Enforcement Kernel for OpenFlow Networks," in Proceedings of the First Workshop on Hot Topics in Software Defined Networks (HotSDN), Helsinki, Finland, 2012, pp. 121–126.
- [6] S. Scott-Hayward, S. Natarajan, and S. Sezer, "A Survey of Security in Software Defined Networks," IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 623–654, 1st Quarter 2016.
- [7] M. H. H. Tabrizi and K. Salah, "SDN Security: A Survey on Issues, Challenges and Open Research Directions," IEEE Communications Surveys & Tutorials, vol. 21, no. 3, pp. 2687–2712, 2019.
- [8] S. Scott-Hayward, G. O'Callaghan, and S. Sezer, "SDN Security: A Survey," in Proc. IEEE SDN for Future Networks and Services (SDN4FNS), Trento, Italy, 2013, pp. 1–7, doi: 10.1109/SDN4FNS.2013.670255.
- [9] M. Khondoker, D. Zymbler, and R. Vogt, "Survey on SDN Based Network Intrusion Detection System Using Machine Learning Approaches," in Proceedings of the 2021 International Conference on Networked Systems (NetSys), pp. 1–5.
- [10] Faezi, S., Shirmarz, A. A Comprehensive Survey on Machine Learning using in Software Defined Networks (SDN). Hum-Cent Intell Syst 3, 312–343 (2023). doi.org/10.1007/s44230-023-00025-3.
- [11] Q. H. Mahmoud, B. H. Al-Husainy, A. Al-Ghobari, and S. M. Taha, "A systematic literature review on machine learning and deep learning approaches for detecting DDoS attacks in SDN," Sensors, vol. 23, no. 9, Art. no. 4441, 2023, doi: 10.3390/s23094441.
- [12] M. A. L. Razib, D. Javeed, M. T. Khan, R. Alkanhel, and M. S. A. Muthanna, "Cyber threats detection in smart environments using SDN-enabled DNN-LSTM hybrid framework," IEEE Access, vol. 10, pp. 53015–53026, 2022, doi: 10.1109/ACCESS.2022.3174322.
- [13] G. Kumar and H. Alqahtani, "Machine learning techniques for intrusion detection systems in SDN—Recent advances, challenges and future directions," Comput. Model. Eng. Sci., vol. 134, no. 1, pp. 89–119, Aug. 2022, doi: 10.32604/cmescs.2022.020724.
- [14] M. Latah and L. Toker, "Artificial Intelligence Enabled Software Defined Networking: A Comprehensive Overview," arXiv preprint arXiv:1803.06818, Mar. 2018.
- [15] R. Doriguzzi-Corin and D. Siracusa, "FLAD: Adaptive Federated Learning for DDoS Attack Detection," arXiv preprint arXiv:2205.06661, May 2022.

- [16] Y. Han, B. I. P. Rubinstein, T. Abraham, T. Alpcan, O. De Vel, S. Erfani, D. Hubczenko, C. Leckie, and P. Montague, "Reinforcement Learning for Autonomous Defence in Software-Defined Networking," arXiv preprint arXiv:1808.05770, Aug. 201.
- [17] N. Harris and S. Khorsandroo, "DDT: A Reinforcement Learning Approach to Dynamic Flow Timeout Assignment in Software Defined Networks," J. Netw. Syst. Manage., vol. 32, Art. 35, 2024, doi: 10.1007/s10922-024-09815-x.
- [18] Y. K. Shen and C. Zhang, "Predictive mobility and cost-aware flow placement in SDN-based IoT networks: a Q-learning approach," J. Cloud Comput., vol. 13, Art. 26, Jan. 2024, doi: 10.1186/s13677-024-00589-w.
- [19] M. AL-Naday, V. Dobre, M. Reed, et al., "Federated deep Q-learning networks for service-based anomaly detection and classification in edge-to-cloud ecosystems," Ann. Telecommun., vol. 79, pp. 165–178, 2024, doi: 10.1007/s12243-023-00977-4.
- [20] Y. Qing, S. Liu, J. Song, H. Wang, and M. Song, "A survey on explainable reinforcement learning: Concepts, algorithms, challenges," arXiv preprint arXiv:2211.06665, Nov. 2022.
- [21] Y. L. Khaleel, M. A. Habeeb, A. S. Albahri, T. Al-Quraishi, O. S. Albahri, and A. H. Alamoodi, "Network and cybersecurity applications of defense in adversarial attacks: A state-of-the-art using machine learning and deep learning methods," J. Intell. Syst., vol. 33, no. 1, Art. 20240153, Aug. 2024, doi: 10.1515/jisys-2024-0153.
- [22] I. Panitsas, A. Mudvari, and L. Tassiulas, "Constrained reinforcement learning for adaptive controller synchronization in distributed SDN," arXiv preprint arXiv:2403.08775, Jan. 2024.
- [23] Tianfang, Y., & Liu, H. (2021). STCC: A SDN-oriented TCP congestion control mechanism for datacenter network. IET Networks, 10(2), 59–69. <https://doi.org/10.1049/ntw2.12005>.
- [24] M. Jenabzadeh, V. Ayatollahitafti, M. Mollakhalili Meybodi, et al., "VNR_LBP: A New Approach to Congestion Control Using Virtualization and Switch Migration in SDN," Wireless Pers. Commun., vol. 139, pp. 733–757, 2024, doi: 10.1007/s11277-024-11492-7.
- [25] J. Arevalo-Herrera, J. Camargo Mendoza, J. I. Martínez Torre, et al., "Assessing SDN Controller Vulnerabilities: A Survey on Attack Typologies, Detection Mechanisms, Controller Selection, and Dataset Application in Machine Learning," Wireless Pers. Commun., vol. 140, pp. 739–775, 2025, doi: 10.1007/s11277-025-11748-w.
- [26] Y. Chen, L. Wang, and H. Zhou, "BI-TE: Achieving bandwidth-indistinguishable traffic engineering via differential privacy in multi-domain SDN," Front. Comput. Sci., 2024, doi: 10.1007/s11704-024-40551-2..
- [27] X. Xu, F. Wu, M. Bilal, J. Bhattacharyya, et al., "XRL-SHAP-Cache: an explainable reinforcement learning approach for intelligent edge service caching in content delivery networks," Sci. China Inf. Sci., vol. 67, Art. 170303, Jun. 2024, doi: 10.1007/s11432-023-3987-y.
- [28] A. K. Alnaim, "Adaptive Zero Trust Policy Management Framework in 5G Networks," Mathematics, vol. 13, Art. 1501, 2025, doi: 10.3390/math13091501..
- [29] L. Borchjes, C. Nyirenda, and L. Leenen, "Adversarial deep reinforcement learning for cyber security in software defined networks," arXiv preprint arXiv:2308.04909, Aug. 2023. arxiv.org/abs/2308.04909.
- [30] M. Q. Hamdan, H. Lee, D. Triantafyllopoulou, et al., "Recent Advances in Machine Learning for Network Automation in O-RAN," Sensors, vol. 23, no. 21, Art. 8792, 2023, doi: 10.3390/s23218792..
- [31] A. S. Tosun, J. M. Arco Rodríguez, R. Doriguzzi-Corin, and V. G. Vassilakis, "SDN as a defence mechanism: A comprehensive survey," Int. J. Inf. Secur., vol. 23, pp. 141–185, 2024..
- [32] Lee, S. Kim, and H. Park, "A flexible SDN-based framework for slow-rate DDoS attack detection and mitigation using reinforcement learning," Electronics, vol. 14, no. 5, Art. 995, Feb. 2025.
- [33] H. Kheddar, D. W. Dawoud, A. I. Awad, Y. Himeur and M. K. Khan, "Reinforcement-Learning-Based Intrusion Detection in Communication Networks: A Review," in IEEE Communications Surveys & Tutorials, vol. 27, no. 4, pp. 2420-2469, Aug. 2025, doi: 10.1109/COMST.2024.3484491

- [34] Gao, D., Guo, C., Zhang, Y., Ji, W., Lv, Z., Li, Z., Zhang, K. and Lin, R. (2024), Risk-Aware SDN Defense Framework Against Anti-Honeypot Attacks Using Safe Reinforcement Learning. *Int J Network Mgmt*, 34: e2297. <https://doi.org/10.1002/nem.2297>
- [35] W. Zhou, D. Zhang, H. Wang, J. Li and M. Jiang, "A Meta-Reinforcement Learning-Based Poisoning Attack Framework Against Federated Learning," in *IEEE Access*, vol. 13, pp. 28628-28644, 2025, doi: 10.1109/ACCESS.2025.3538891.