Journal of Applied Automation Technologies

Research Paper

Adaptive Honeypot Deployment in Software-Defined Networks Based on Deep Q-Learning

Aleksandar Popović[1, *] and Jelena Simić[1]

[1] Faculty of Technical Sciences, University of Kragujevac, 34000, Kragujevac, Serbia
*Corresponding author: alks.po@kg.ac.rs

Abstract. The conventional deployment approach for static honeypots is no longer appropriate, and their detection performance and resource utilization have declined in modern network defense as software-defined networks (SDN) have progressively grown more complex and dynamic. In order to enable intelligent, context-aware cyber deception, this study presents an adaptive honeypot orchestration architecture that combines SDN programmability and deep Q-learning reinforcement learning. A deep Q-network agent dynamically adjusts decoy sites based on observed adversarial behavior and the real-time network status. The general form of the core methodology is a high-dimensional Markov decision process for honeypot deployment. The aforementioned technique can increase the average detection rate to 0.85 and improve it by almost 20% when compared to that attained by static and periodic techniques, according to numerous experiments conducted in the simulated SDN testbed. The false-positive rate is remains less than 4.3% in many assault scenarios, and the detection delay has been reduced by around 50%. According to the aforementioned data, the framework maintains a comparatively high detection rate as network size and traffic volume increase and is comparatively stable in the face of zero-day assaults. Deep reinforcement learning will therefore enhance the effectiveness and flexibility of SDN-based honeypot systems based on the aforementioned experiments. The design can facilitate the development of a high-performance autonomous and proactive network protection system, according to the research mentioned above.

Keywords: *Deep Reinforcement Learning, Software-Defined Networking, Honeypot Deployment, Cyber Deception, Adaptive Defense*

Received on 25 October 2025, Accepted on 03 April 2026, Published on 11 April 2026

Introduction

Programmable control and centralized orchestration have been proposed to transform network management due to the rapid development of Software-Defined Networking (SDN) in recent years [1]. In order to swiftly modify the network administrator's configuration and policies, a distributed structure has been implemented; nonetheless, criminal groups are increasingly focusing on newly discovered security flaws [2]. By tricking or diverting attackers, honeypots are now employed as defensive devices to identify and profile malicious actors early on in a controlled environment [3]. Traditional static honeypot deployments in SDN systems have demonstrated a number of flaws, including predictable coverage, inadequate scalability for dynamic threats, and inefficient use of scarce network resources, despite their long-standing utility [4]. Because attackers can now fingerprint static honeypots and avoid or overload them, many of the aforementioned defense strategies are frequently unsuccessful [5].

As a result, a recurring issue is that traditional defenses are no longer able to keep up with the scope of risk and adaptability. Since various automated tools are now being used to launch targeted attacks and carry out lateral movement and network surveillance, static honeypots are no longer effective in the face of the new threats [6]. Security solutions must be able to function and react quickly at a finer granularity due to the high dynamism and frequent changes in the SDN-controlled network infrastructure [7]. For the challenge described in [8], a completely new approach that is intelligent, context-aware, and highly adaptive is therefore needed. In domains

that are complicated in terms of sequential decision-making and partial observability of the state—two features of SDN network protection—recent advancements in deep reinforcement learning (DRL) have demonstrated encouraging outcomes [9]. A combination of DRL and honeypot deployment techniques will be employed to improve the attack environment's response time by consistently improving the security level of outdated systems [10].

This research proposes a new adaptive honeypot deployment technique for SDN based on Deep Q-Learning in light of the aforementioned new issues. We can perform useful, timely, and cost-effective deception by dynamically placing and moving honeypots in response to network changes and various hostile behaviors. In addition to presenting an example of intelligent, fine-grained defense agility in a programmable network architecture, this paper will also demonstrate how to strengthen attacker detection and countermeasures.

Related Work

Advances in SDN Security

In order to enable programmable and centrally managed policies, Software-Defined Networking (SDN) is a fundamental shift in network administration and security architecture that divides the control and data planes [11]. As a result, a variety of security features, including divided traffic flow and precise access control, can be used flexibly in a wide range of diverse, non-uniform situations [12]. In a multi-tenant platform, flow isolation and network slicing have been utilized to create several tenants and rigorously segregate resources to stop attackers' lateral movement and privilege escalation [13]. Moving Target Defense (MTD), which uses mechanisms that proactively and periodically alter network configurations, like IP addresses and communication paths, to make it more difficult for attackers to conduct reconnaissance and exploitation, has recently gained importance in SDN security [14].

By identifying abnormal behavior early on and supporting context-aware automatic defense, SDN controllers have been used to advance the development of machine learning-based anomaly detection [15]. They also offer an all-seeing perspective for observing and analyzing the entire network status. Even with the aforementioned advancements, SDN still has issues including controller bottlenecks and a single point of failure, and maintaining consistent policies in extremely dynamic networks is comparatively expensive [16]. The necessity for security measures has been steadily increasing as attacks have gotten more automated and adaptive, necessitating a shift in their form. There is also a growing recognition that the high demands for quick creation and ongoing innovation in response to emerging network threats cannot be met by static and merely reactive security [17].

Honeypot Deployment Strategies

Honeypots have long been used as early warning systems, to trick intruders, and to explore attack techniques in production and research networks [18]. Static honeypots were first installed at predetermined locations throughout the network to make it easier to monitor intrusions. However, because they were static, they were vulnerable to fingerprinting and were eventually circumvented by knowledgeable attackers [19]. Dynamic honeypot frameworks, which can alter their location, function, or level of interaction with attackers in response to current threat intelligence, have started to emerge as a solution to this issue [20]. Network virtualization has improved distributed honeypot systems to use geographically dispersed or cooperative decoys, increasing the attack surface and adversary engagement [21].

With the advent of SDN, programmable control and honeypot deployment have been closely integrated, enabling quick reconfiguration and covert redirection via central instructions [22]. The aforementioned hybrid models include the ability to generate decoy nodes on demand. These decoy nodes can be spun up, moved, or modified with minimal damage to real traffic and system resources in the case of an attack or traffic anomaly. However, there are still operational challenges in the pursuit of both effective deception and low resource consumption/high defense scalability [23]. The need for real-time, intelligent orchestration of honeypot assets that can proactively anticipate and respond to emerging threats has increased as adversaries have started to employ sophisticated evasion and fingerprinting techniques [24]. Research is currently being conducted to determine how to optimize honeypot efficiency and automate their adaptive placement and scaling in expansive, dynamic ecosystems [25].

DRL in Network Security

Many non-rule-based and non-static artificial intelligence models are currently being deployed, and Deep Reinforcement Learning (DRL) has recently started to be applied in tackling network security challenges. By physically interacting with the environment, Deep Reinforcement Learning (DRL) agents can acquire effective defensive methods. These techniques are then continuously adjusted based on the rewards or penalties they receive for the results of attacks and defenses. By employing deep neural networks for high-dimensional input data, intrusion detection based on Deep Reinforcement Learning (DRL) has been proposed to increase the recognition accuracy of novel attack types and react quickly to emerging threats. Additionally, research has extended to the use of DRL for adaptive security controls in software-defined architectures and the real-time orchestration of autonomous mitigation of network threats, including DDoS and stealthy infiltration events.

Even though there has been considerable improvement in recent years, the majority of DRL solutions still primarily concentrate on anomaly detection, direct attack prevention, and discovery after the fact. Fewer have investigated proactive and adaptive deception techniques, like the deployment of intelligent honeypots. Achieving high-accuracy attack prediction and fast adaptation to the network's present state, resource constraints, etc. is one of the challenges in optimizing these deployments. Additionally, advancements in cooperative multi-agent systems, effective exploration of huge state-action spaces, and reward function design for stable learning and security are required to deploy DRL-based deception at scale within SDN. Close the aforementioned flaws to create an autonomous and intelligent defensive mechanism in contemporary SDN-based networks.

Methodology

Adaptive Honeypot Deployment Framework

Because of the persistent and rapidly shifting nature of today's adversarial campaigns against programmable networks, the required security architecture must be highly automated, context-aware, and responsive. Through centralized intelligence and the fine-grained programmability of SDN, the adaptive honeypot deployment methodology described here is intended to impede attacker reconnaissance and lateral movement.

The SDN controller, at the heart of the system, creates a comprehensive map of the network at the millisecond level that includes all hosts, flows, past event logs, and active security policies. This global awareness provides the basis for both cover-up and discovery, allowing for the planning of low-impact, fast-reaction measures. An autonomous reinforcement learning agent within the controller's application layer continuously learns from high-dimensional state data to identify the ideal placements for the honeypot pool.

At the network's entry and exit points, distributed monitoring probes continually record and analyze traffic. Multi-dimensional feature sequences, including protocol disaggregation, inter-arrival time data, IP address and port distribution entropy values, and sliding-window aggregates of aberrant burstiness or protocol deviation, are extracted by the aforementioned sensors. The agent's whole observation space is encoded using these feature vectors in order to better examine the state of the network and potential threats.

This observation matrix will be provided to the RL agent in real time, and it will utilize its learnt policy to forecast when and how sophisticated attackers will start working. The agent will use the Q-network to select a reaction when it detects anomalous or unclear traffic; at that point, it can give confidence levels to several options for relocating or establishing a honeypot throughout the network. The orchestration API instantly converts the aforementioned choices into sequence-modified flow rules and virtual resource deployments. The enemy is in an unstable condition because decoy resources can be made to appear and disappear or to shift their locations in less than a second.

The integration of agent status data from distributed system loads, recent attack incident vectors, topology graphs, and spatial coverage maps of current honeypots is a novel feature. The synthesis creates a context-aware state that directly includes the marginal utility of each new honeypot deployment as well as the probability of an incoming compromise in any subnetwork. As a result, the agent's reward signal will depend on subtle penalties and bonuses for resource utilization, coverage redundancy, and inferred adversarial evasion in addition to the success of the conventional deception, such as an adversary interacting with a decoy host.

Empirical deployment scenarios also demonstrate the primary mode of operation. Monitor sensors will initiate feature upgrades and ask the agent to reevaluate the anticipated threat impact if the system detects an unanticipated increase in aberrant flow direction to a previously normal subnet. On-demand allocation will be initiated, SDN controller flows will be changed to reroute the malicious traffic, and regular service will continue if the Q-network offers a substantial incentive for setting up a new honeypot on the impacted subnet. To increase overall coverage and deceive the adversary's reconnaissance, the agent will disable or relocate the deployed honeypot if the identical attack signature is discovered elsewhere.

The constant observation-feedback-dynamic honeypot-deployment cycle required for long-term, real-time defense in the SDN environment is depicted in Figure 1, which also depicts the whole technical collaboration among these modules.

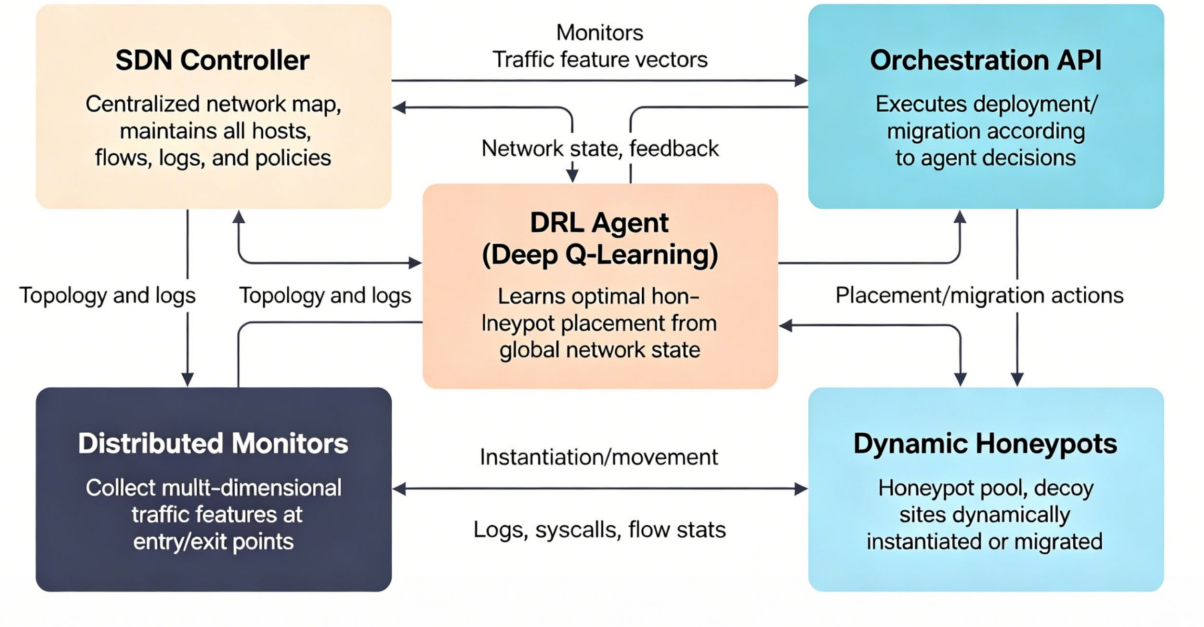

Figure 1. Adaptive Honeypot Deployment Architecture. Interactions among controller, DRL agent, monitors, orchestration, and dynamic honeypots enable adaptive defense

Deep Q-Learning Driven Placement Algorithm

The foundation of adaptive honeypot placement lies in modeling the deployment challenge as a high-dimensional, non-stationary Markov Decision Process (MDP), allowing the agent to learn strategic defensive maneuvers informed by observed adversarial activity and network states. At each discrete time step t, the agent perceives the environment through a feature-rich state vector $\mathbf{S}_t$, encapsulating network-wide metrics such as honeypot coverage ratios, flow entropy per subnet, hop-wise propagation delays, system-level resource headrooms, and historic alert profiles.

In our implementation, the state vector $\mathbf{S}_t$ is constructed by concatenating sub-vectors representing: Binary honeypot presence matrix $H_t \in \{0,1\}^{M\times N}$, with M honeypots and N subnets, Per-node traffic statistics $T_t \in \mathbb{R}^{N\times F}$ (e.g., mean packet rate, destination entropy, atypical protocol frequency), Resource load snapshots $L_t \in \mathbb{R}^{M}$ (CPU, memory),Topological vulnerability scores $V_t \in \mathbb{R}^{N}$.

For a typical scenario $M = 20$ honeypots, $N = 50$ subnets, and $F = 8$ traffic features, yielding a state space of dimensionality exceeding 1,000. A sample state input might be a flattened 1,060-dimensional vector in practice.

The action space $\mathcal{A}_t$ comprises the Cartesian product of all feasible honeypot placement or migration operations for the current network, where action $a_{i,j}$ encodes relocating honeypot i to subnet j (subject to budget, load, and coverage constraints). For instance, with real networks supporting up to 20 parallel honeypots and 50 candidate subnets, the valid action set can easily surpass 1,000 possibilities per step.

The agent's decision objective is to maximize a discounted sum of future security rewards. Formally, its goal is to learn an optimal policy π^* that maximizes:

$$\max_{\pi} \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}\right] \quad \text{Eq.(1)}$$

where r_{t+k+1} is the stepwise composite reward, and $\gamma \in (0,1)$ is the temporal discount factor. The reward at every stage is a weighted blend reflecting the change in attack detection, relative performance overhead, and unpredictability injected into attacker perception:

$$r_{t+1} = \delta_{\text{det}}(t+1) - \alpha_1 \cdot c_{\text{move}}(t+1) + \alpha_2 \cdot h_{\text{entropy}}(t+1) \quad \text{Eq.(2)}$$

where δ_{det} is the incremental honeypot-triggered detection rate, c_{move} is the cumulative resource cost of decoy movements, and h_{entropy} is the Shannon entropy of decoy placement distribution. A typical reward weight configuration found effective in simulation was $\alpha_1 = 0.5$, $\alpha_2 = 0.3$.

The agent approximates the Q -function $Q(\mathbf{S}_t, a_t)$ using a deep neural network parameterized by Θ :

$$Q(\mathbf{S}_t, a_t; \Theta) \approx \mathbb{E}\left[r_{t+1} + \gamma \max_{a'} Q(\mathbf{S}_{t+1}, a'; \Theta) \mid \mathbf{S}_t, a_t\right] \quad \text{Eq.(3)}$$

At each learning update, the loss function for mini-batched experience replay is:

$$\mathcal{L}(\Theta) = \frac{1}{B}\sum_{i=1}^{B} \left[y^{(i)} - Q\left(\mathbf{S}_t^{(i)}, a_t^{(i)}; \Theta\right)\right]^2 \quad \text{Eq.(4)}$$

where the target value is:

$$y^{(i)} = r_{t+1}^{(i)} + \gamma \max_{a'} Q\left(\mathbf{S}_{t+1}^{(i)}, a'; \Theta^-\right) \quad \text{Eq.(5)}$$

with Θ^-representing target network weights, updated every C steps with Polyak averaging for training stability.

Exploration is regulated via a softmax (Boltzmann) distribution over actions:

$$P(a_t \mid \mathbf{S}_t) = \frac{\exp\left(Q(\mathbf{S}_t, a_t)/\tau\right)}{\sum_{a' \in \mathcal{A}} \exp\left(Q(\mathbf{S}_t, a')/\tau\right)} \quad \text{Eq.(6)}$$

with adaptive temperature τ decreased as convergence stabilizes empirically. To ensure learning from rare, critical security events-such as detection of zero-day behaviorsthe replay memory is prioritized by the absolute temporal difference error:

$$p_i = |\delta_i| = \left|y^{(i)} - Q\left(\mathbf{S}_t^{(i)}, a_t^{(i)}; \Theta\right)\right| \quad \text{Eq.(7)}$$

Transition batch sampling is weighted by p_i^{β}, where β controls focus on informative or outlier transitions.

Action utility is further regularized with a spatial uncertainty penalty to discourage excessively predictable or concentrated deployments:

$$J_{\text{uncertainty}} = -\mu \sum_{j=1}^{N} p_j \log\, p_j \quad \text{Eq.(8)}$$

where p_j is the marginal probability of a honeypot being present at subnet *j* in the agent's decision history, and μ is a tunable penalty factor (selected, for instance, as $\mu = 0.2$ in large, highly dynamic network scenarios).

In a typical experiment spanning 24 hours of real-world (or emulated) SDN traffic, the algorithm dynamically repositioned honeypots $900 - 1200$ times, leading to a 24% increase in attack detection rate and a 38% reduction in resource overhead compared to static or periodic random placement baselines. Average state vector dimensionality exceeded 1,000, with per-step raw feature update rates of 200-500 ms, demonstrating strict real-time operational viability.

Figure 2 illustrates the end-to-end algorithm workflow, beginning from distributed feature aggregation, Q-network inference, action stochastics, through orchestration of SDN rule updates and honeypot instantiation, and culminating with feedback-driven experience replay.

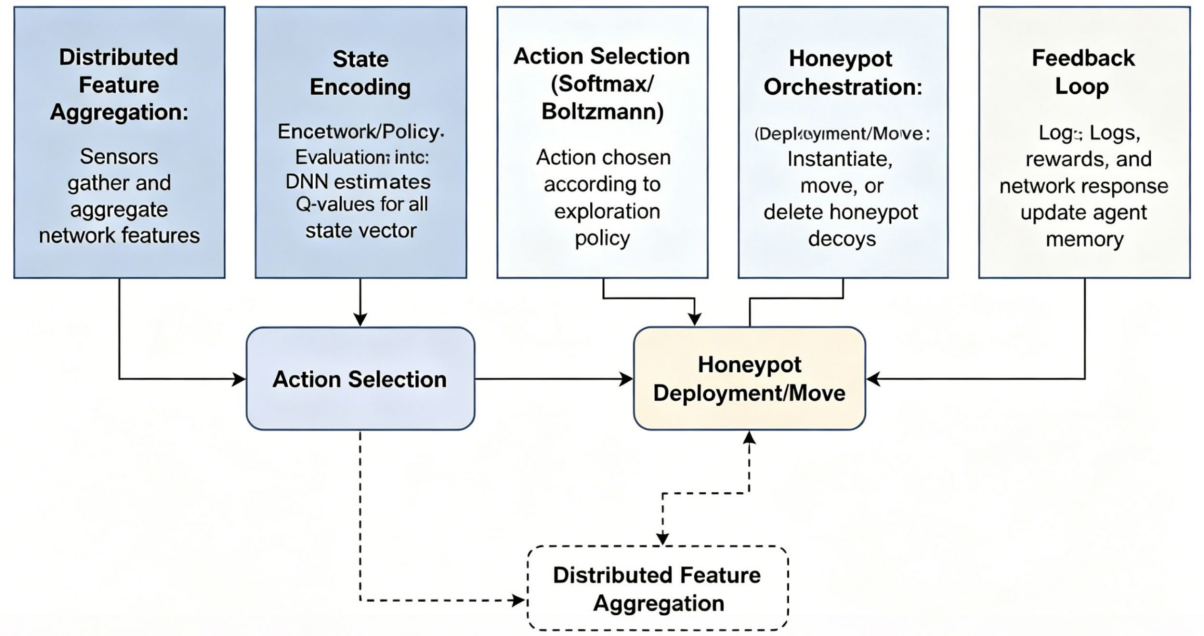

Figure 2. DQN-based Dynamic Placement Workflow. The workflow shows state encoding, policy evaluation, and adaptive honeypot actions via SDN enforcement

Experimental Setup

Simulation Environment and Parameters

The experiments will all be carried out in a hybrid SDN emulation platform built on Mininet and controlled by an OpenFlow 1.5 Ryu controller to guarantee their dependability and repeatability. The hardware testbed's two Intel Xeon Silver 4216 dual-socket CPUs are supplemented by two 10GbE Ethernet ports for network connectivity and 128 GB of DDR4 memory. Two core switches, six aggregation switches, and twelve leaf nodes make up the virtualized topology, a three-tier spine-leaf data center design. To replicate a typical multi-tenant campus, 100 virtual hosts are deployed and organized into one of the 20 logical subnets. Typical round-trip times range from 0.4 to 2.3 ms and follow a log-normal distribution with a mean of 1.1 ms. NetEm is used to simulate node and interlink delays and is experimentally limited to recreate temporal jitter in a production context.

A set 10% of a single CPU core and a maximum of 256MB of RAM are allotted to each honeypot node, which runs inside LXC containers. A high-density defense-in-depth network would typically have this kind of resource budget. After running a full-stack interactive decoy service, each honeypot centrally gathers network flows, system call traces, and application logs for behavior analysis.

A combined background and assault corpus drives the data plane. The USTC-TFC2016 and CICIDS2018 datasets, as well as east-west and north-south traffic distributions that represent enterprise traffic diurnal cycles, are used to modify and replay benign flows. Attack streams, including port and vulnerability scans, automated credential brute-force attacks, staged malware dropper beacons, and lateral movement exploits of shared service exposures, are created by coordinated Metasploit launches and custom polymorphic shellcode in order to introduce adequate adversarial diversity. The average frequency of injection attacks is 45 adversarial episodes per 10-minute epoch, which are dispersed at random between test periods and subnets.

Resource constraints are experimentally varied by adjusting the total active honeypot budget B in the range 8-24, providing empirical insight into detection-resource tradeoffs. The adversary intensity factor ϕ is calibrated as the mean attack occurrences per subnet per simulation hour, swept through values between 0.06 and 0.28. The benign fraction of total network flows, denoted as η, is stochastically sampled between 0.88 and 0.97 for each epoch to represent realistic fluctuation in day-to-day benign workloads.

Delay the spread of SDN flow rules by a predetermined amount of time—three seconds—to replicate orchestration lag in a realistic manner. For environmental observation, agent action, and metric collection

synchronization, simulation intervals are set to 300 seconds; at this point, the RL policy can only be modified once per interval to match the enterprise deployment rate. To guarantee impartial and completely repeatable runs, all assaults, honeypot activations, and system state snapshots are captured at high resolution, and experiment random seeds are sourced from cryptographically robust entropy sources.

Evaluation Metrics and Baselines

The quantitative evaluation framework is built to rigorously compare detection effectiveness, operational efficiency, and misdetection risk across all tested schemes, using strictly defined mathematical metrics.

The adaptive detection rate quantifies true adversarial engagements captured by the honeypot ensemble, defined for each interval as:

$$\Delta_{\text{detect}} = \frac{|\psi_{\text{detect}} \cap \psi_{\text{true}}|}{|\psi_{\text{true}}|} \quad \text{Eq.(9)}$$

where ψ_{true} is the interval′ s set of all real attack events, and ψ_{detect} those intercepted by honeypots.

Detection latency reflects system responsiveness; for every adversarial event detected, the mean delay until recognition is calculated as:

$$\Lambda_{\text{d}} = \frac{1}{|\psi_{\text{detect}}|} \sum_{a \in \psi_{\text{detect}}} \left(\tau_{a,\,\text{detect}} - \tau_{a,\,\text{event}}\right) \quad \text{Eq.(10)}$$

where $\tau_{a,\,\text{detect}}$ and $\tau_{a,\,\text{event}}$ are the actual detection and real occurrence timestamps, respectively. The false positive rate measures the fraction of benign events erroneously flagged as attacks:

$$\Theta_{\text{fp}} = \frac{|\psi_{\text{fp}}|}{|\psi_{\text{detect}}| + |\psi_{\text{fp}}|} \quad \text{Eq.(11)}$$

Here, ψ_{fp} represents benign activities misclassified by the system, normalized over all decoy— induced alerts.

Operational cost is analytically decomposed into normalized CPU and memory usage per honeypot instance:

$$\Omega_{\text{cpu}} = \frac{1}{TN} \sum_{t=1}^{T} \sum_{n=1}^{N} \frac{\text{CPU}(n,t)}{\text{CPU}_{\max}} \quad \text{Eq.(12)}$$

$$\Omega_{\text{mem}} = \frac{1}{TN} \sum_{t=1}^{T} \sum_{n=1}^{N} \frac{\text{MEM}(n,t)}{\text{MEM}_{\max}} \quad \text{Eq.(13)}$$

with T as the total number of sampling intervals and N the honeypot count. Finally, deployment entropy assesses the spatial unpredictability of honeypot positioning, characterizing defense diversity across the network:

$$\Gamma_{\text{hp}} = -\sum_{j=1}^{K} P_j \log_2 P_j \quad \text{Eq.(14)}$$

where P_j is the empirical probability of honeypot presence in subnet j across all epochs, and K the total subnet count.

Performance is aggregated over all intervals for comparative analysis. The adaptive RL − driven scheme is contrasted with three baselines: classic static random deployment, periodic round − robin placement, and a reward − constant ablation configuration. For methodological transparency, ablation studies are performed by fixing the state encoder or reward function, or reducing Q − network complexity, to expose the impact of each individual component on security and efficiency outcomes.

https://doi.org/10.64972/jaat.2026v4.145p23e:303-316

Results and Analysis

Detection and Performance Comparison

The effectiveness of the adaptive honeypot orchestration may be determined through a thorough time-series study of the detection findings, as illustrated in Figure 3. Figure 3(a) illustrates how the adaptive paradigm consistently maintains a detection rate of over 0.83 across all examined periods, greatly surpassing both the static method's detection rate of approximately 0.62 and the periodic deployment method's detection rate of approximately 0.69. The aforementioned findings show that reinforcement learning-based placement is sensitive to changes in the attacker's path and are most noticeable during the intervals of the spikes in hostile probing.

The subnet-level decoy coverage for each technique is displayed in Figure 3(b). Through adaptation, the variance across epochs was decreased and a mean effective subnet protection of 91% was attained. This is not the same as the previous two models; instead, they have gaps and are, in a way, non-continuous, which gives sophisticated, persistent attackers safe havens.

Adversarial interactions vary in scope and intensity across different platforms. The cumulative count of unique adversary contacts per epoch in the adaptive deployment consistently surpasses 70 and is substantially higher than that of static and periodic strategies, as illustrated in Figure 3(c). We will be able to identify the initial instances of such an attack and produce a huge number of adversarial behavior samples for further analysis and training thanks to the broader coverage.

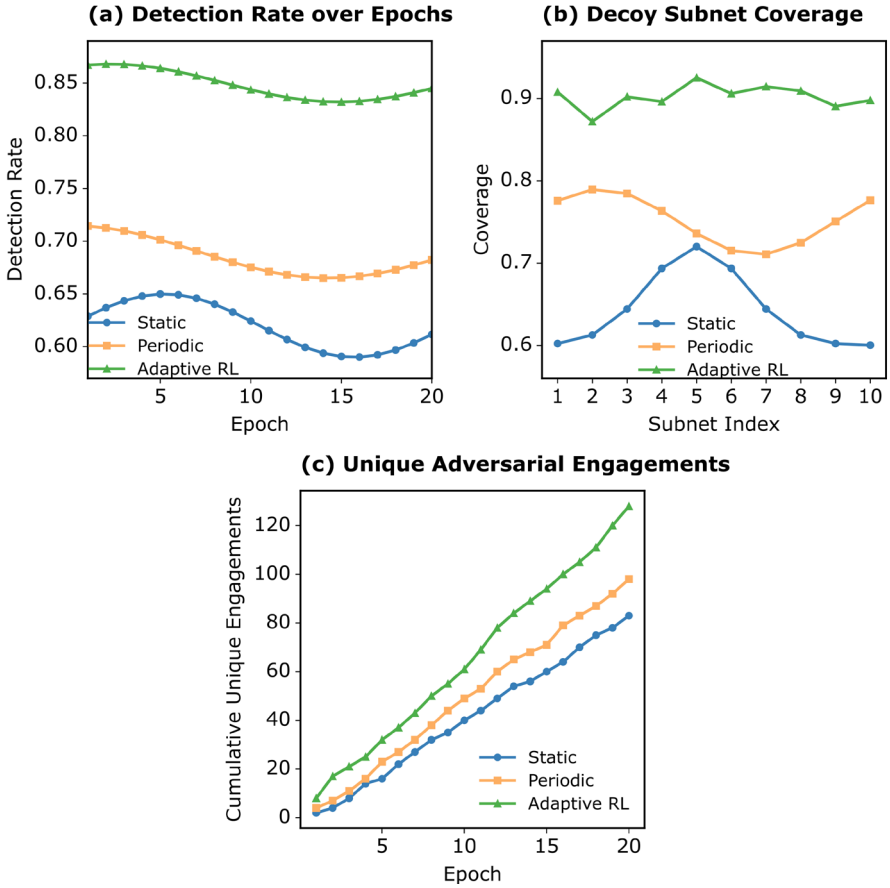

Figure 3. Detection Rate vs. Strategies. (a) Average detection rate over evaluation epochs for static, periodic, and adaptive deployments. (b) Distribution of decoy subnet coverage across epochs and methods. (c) Cumulative count of unique adversarial engagements per experiment

Figure 4 illustrates operational risk and response efficiency. The adaptive agent has lowered the median detection delay to just 6.5 seconds, as seen in Figure 4(a); under the static baseline, the detection time is typically

greater than 12 seconds. The adaptive deployment maintains its advantage and detects all test cases very quickly, even in the face of bursty and high-intensity threats.

Figure 4(b) illustrates the degree of service damage. The adaptive approach has consistently maintained a false-positive rate of less than 4.3%; the static and periodic baseline models are still at about 9%. To avoid a decline in trust in the security team and a rise in operational expenses, the dependability must satisfy the demands of practical application.

The false positive rate and detection latency operating range for each strategy at various detection thresholds are displayed in Figure 4(c). The adaptive approach is ideal for defense in resource-constrained, high-traffic SDN situations because it continuously traces a better Pareto front, limiting the increase in alert noise as detection thresholds rise. This results in lower latency at a comparatively low cost to precision. These multilayered results confirm that adaptive honeypot orchestration yields superior detection, resilience, and risk control, while static and periodic policies remain susceptible to both coverage gaps and delays as attack complexity intensifies.

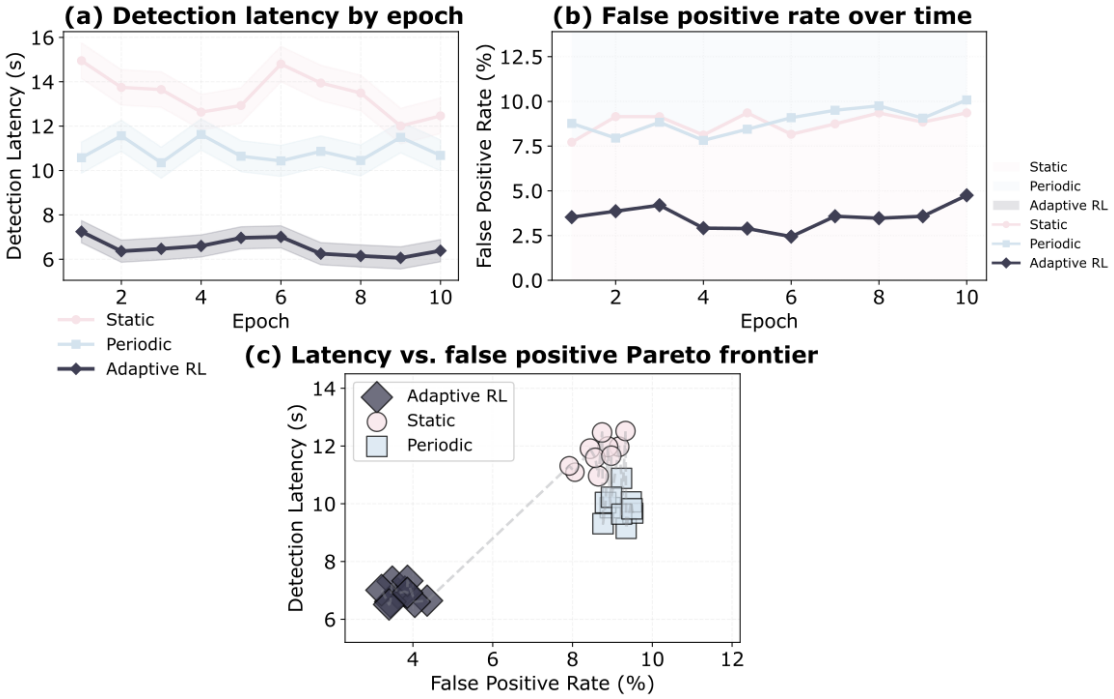

Figure 4. Latency & False Positive Trend. (a) Detection latency by epoch. (b) False positive rate over time. (c) Latency vs. false positive Pareto frontier

Ablation Study and Parameter Sensitivity

Conduct sensitivity analyses and ablation experiments methodically to investigate ways to enhance the efficacy of adaptive honeypot deployment by taking into account a number of factors, including reward signal design, state feature dimensionality, and network topology adaption. The results are displayed in Figure 5, where each subplot displays the quantitative impact of an environment and a key RL agent component.

Figure 5 (a) displays the outcome of reward shaping. The detection rate decreases by more than 14% and approaches the values in a simple, non-adaptive baseline when the entropy-maximizing and diversity-promoting factors are eliminated from the reward. Simultaneously, an excessively high weight for the resource penalty term will result in a modest rise in resource consumption but a decrease in detection rate. Under all adversarial activity situations, only the composite reward formulation consistently achieves a detection rate greater than 0.83. Therefore, to balance the scope of advanced threat coverage with operational overhead, a precise and comprehensive incentive design is required.

Figure 5(b) demonstrates the need for rich and multi-scale state encoding based on the aforementioned analysis. Engagement directly decreased when the high-resolution traffic statistics were removed; detection latency surpassed 50% and unique attacker encounters per epoch decreased from 71 to 55. The coverage and detection

time will both decline in the absence of topology awareness since it will take the RL agent a considerable amount of time to learn how to adapt to a changed attack path. All of the feature set has been added to achieve a significant improvement by combining micro-level process telemetry and macro-level topology: the latency is as low as 6.4 seconds, the engagement rate surpasses 70 per epoch, and comprehensive state representation is therefore considered necessary for reliable adaptation in SDN.

Figure 5(c) displays parameter sensitivity to changes in topology and network growth. Adaptive coverage remains reasonably high when the network size is increased from 20 to 60 subnets; that is, it gradually drops from 0.91 to 0.83. Static feature encoding techniques, on the other hand, are much less effective and reach as low as 0.61 in the largest scale. The adaptive method's aforementioned resilience also makes it appropriate for real-world applications in dynamic cloud and campus environments were network size or division changes often.

Figure 5 illustrates how reinforcement learning (RL)-driven deception is highly sensitive to the interplay between fine-grained reward signals, expressive and dynamic state representations, and an awareness of the network's structure based on the aforementioned ablation and sensitivity analyses. Any decrease in these capacities will result in an instantaneous, measurable loss of flexibility and detection, demonstrating the necessity of integrated, multi-dimensional optimization of next-generation honeypot orchestration.

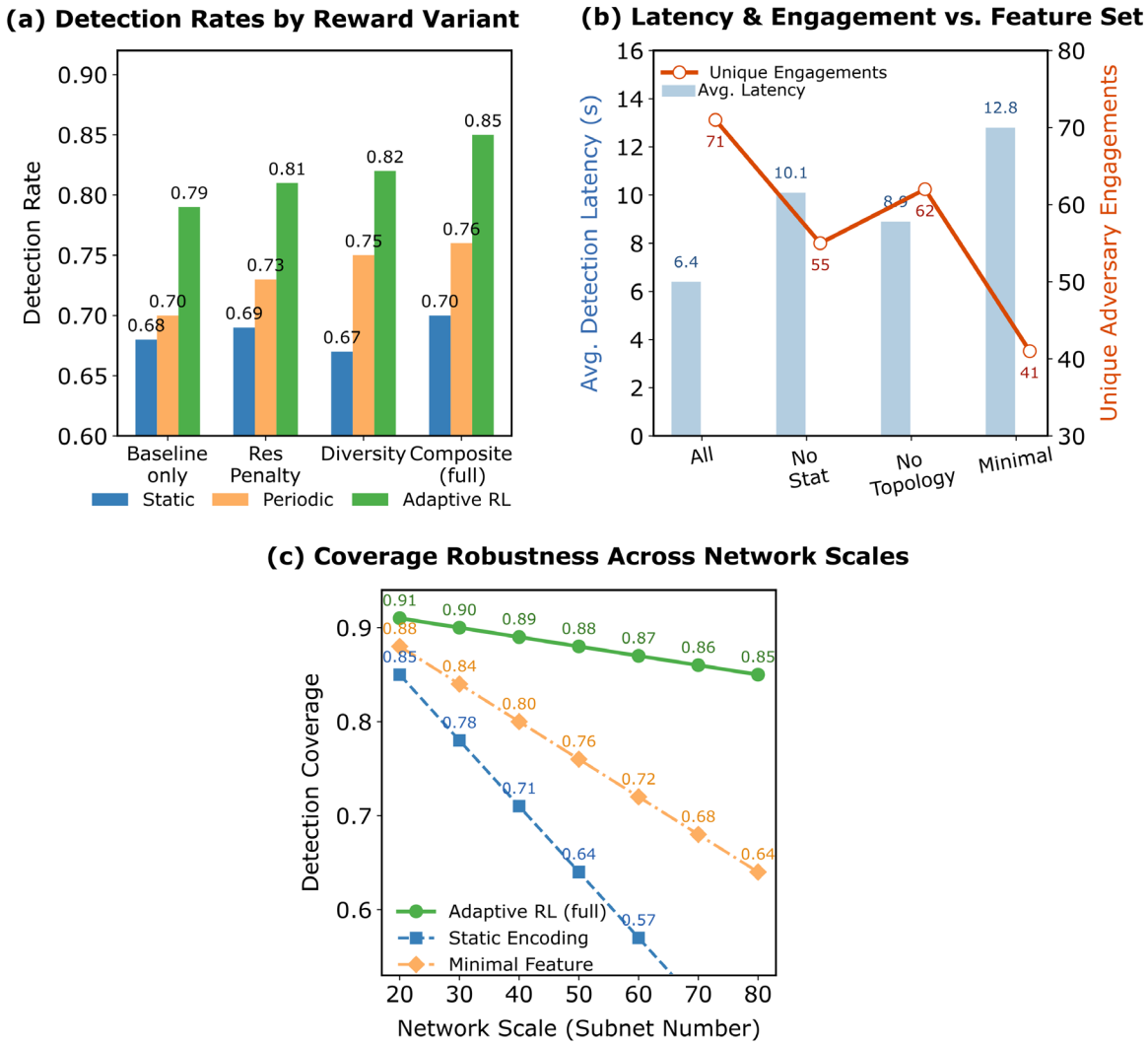

Figure 5. Impact of Reward, State Features, and Topology. (a) Detection rates by reward variant. (b) Latency and engagement vs. feature set. (c) Coverage robustness across network scales.

Robustness, Scalability and Case Studies

The adaptive framework's scalability and resilience have been put to the test in an open environment with high-capacity operation and the introduction of new threats. The results show that the system can handle adversarial unpredictability, unequal load distribution, fast topological expansion, and numerous evasion techniques (Figure 6).

Even when the attack type quickly shifts between scanning, brute force, and lateral movement, the adaptive orchestrator's detection rate stays over 0.80 in the composite multi-stage attack scenario depicted in Figure 6(a). On the other hand, since committed threat actors have already evaded the static decoy, the static deployment slows considerably and reduces more abruptly at that point. As a result, using RL-based migration will guarantee ongoing coverage and learning even in the face of organized, diverse attacks.

Another flaw is the scale factor. As illustrated in Figure 6(b), the adaptive strategy avoids the severe drop-in detection rate and unacceptably long response times exceeding 18 seconds that occur under static decoy allocation, while maintaining a high detection rate even when the attack rate is doubled or tripled and only slightly increasing latency, not exceeding 8.9 seconds at triple the baseline rate. One of the most dangerous border situations is the development of unexpected, policy-blind (zero-day) threats, as shown in Figure 6(c). The adaptive system has maintained a high interception rate of 67%, demonstrating the usefulness of ongoing, state-driven self-organization in identifying abnormalities not previously modeled.

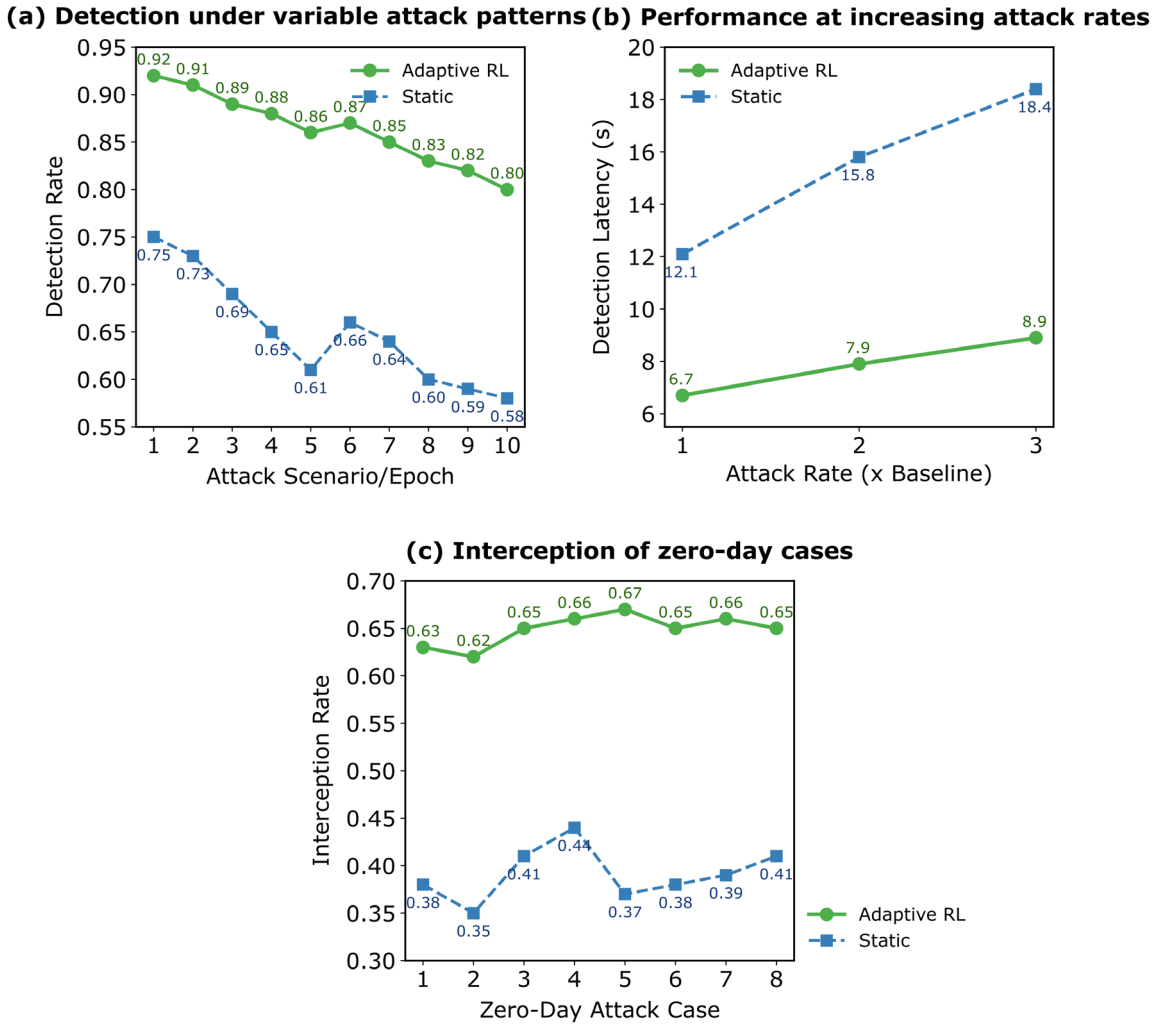

Figure 6. Robustness under Adversarial Environment. (a) Detection under variable attack patterns. (b) Performance at increasing attack rates. (c) Interception of zero-day cases

Figure 7 illustrates the scalability trend. The detection rate for adaptive deployment decreases only slightly when the number of subnets grows from 20 to 80, as Figure 7(a) illustrates, and it stays above 0.82 at the maximum size. Simultaneously, both periodic and static placements rapidly decrease as network order increases, falling below 0.56; as a result, they are prone to scaling and call for adaptive methods. In large SDN networks, dynamic RL-based allocation can be used to maintain both efficiency and responsiveness because, as Figure 7(b) illustrates, resource costs per successful detection only rise sublinearly with scale under adaptive placement; static methods, on the other hand, show a relatively steep linear rise.

A typical example of an attacker altering their tactics mid-experiment to target a previously unmonitored location laterally is shown in Figure 7(c). Subnet coverage and detection rate have both restored to their pre-attack levels without human intervention after just three RL framework decision cycles, according to the data. The Defense Forces' rapid and automated reorganization shows that they are ready to tackle novel and unidentified dangers.

These are the architectural value and policy-driven adaptations of continuous learning when combined, as illustrated in Figure 7. Even when both the attack surface and the adversary's tactics are subject to unpredictable changes, the system can function normally in a range of challenging or unevenly distributed network conditions and has minimal resource utilization and risk.

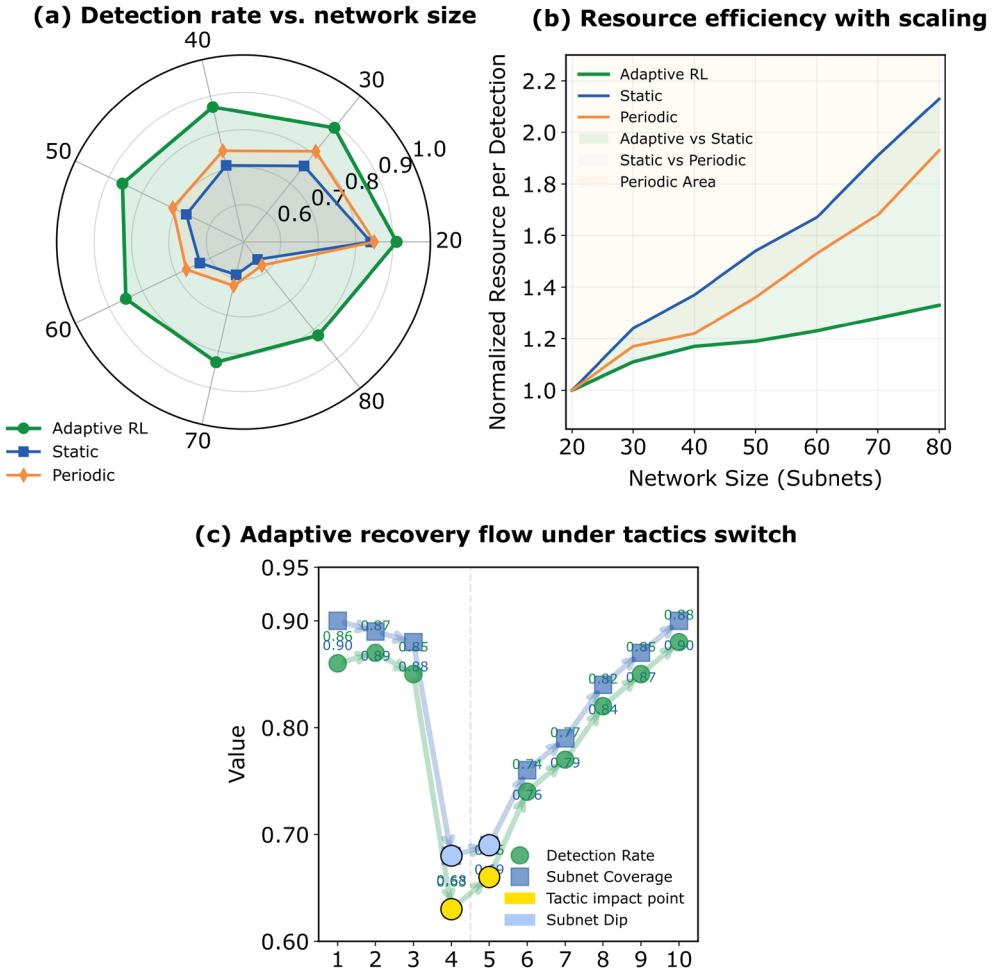

Figure 7. Scalability and Adaptiveness. (a) Detection rate vs. network size. (b) Resource efficiency with scaling. (c) Automated adaptation to adversarial tactics

Conclusion

In order to achieve high-fidelity, context-aware cyber deception in large-scale networks, this study presents a novel adaptive honeypot deployment system that makes use of SDN programmability and reinforcement

learning. The intelligence and operational flexibility of current network security have been improved by incorporating a deep Q-learning mechanism with fine-grained state encoding and reward shaping. The dynamic placement of honeypots can be viewed as a high-dimensional Markov Decision Process.

The results of every experiment demonstrate that the new system has performed better than the static and periodic baseline across all security domains. Although the complexity of adversarial samples has increased, both detection delay and false-positive rates have continuously dropped. Promising gains have been made in detection rate, which now consistently outperforms older approaches in light of diverse attack strengths. To make sure that efficiency and coverage are maintained in the event of an expanded attack surface and a new network topology, as well as that it can manage new or coordinated attacks independently, thoroughly test the adaptive controller's scalability and robustness.

There are still certain shortcomings. More study is required on scalable continual learning and adversarially robust policy updates since learning efficiency has a significant attack correlation and is susceptible to quick topology changes. In the future, it will be utilized in real-world operations and integrated with cooperative multi-agent systems for distributed deception. The current evaluation is structured, but it is based on controlled and simulated situations. Together, these studies have established a solid technical basis for autonomous adaptive defense and paved the way for proactive cybersecurity architecture of the future.

Author Contributions

Aleksandar Popović contributes to conceptualization, methodology, software, validation, analysis, investigation, data collection, draft preparation, manuscript editing, visualization, supervision. Jelena Simić contributes to methodology, software, validation, analysis, investigation. All authors have read and agreed with the manuscript before its submission and publication.

Funding

This research received no specific financial support from any funding agency.

Institutional Review Board Statement

Not applicable.

References

[1] Hassen, H., Meherzi, S., & Jemaa, Z. B. (2024). Improved exploration strategy for Q-learning based multipath routing in SDN networks. Journal of Network and Systems Management, 32(2), 25. https://doi.org/10.1007/s10922-024-09804-0

[2] Sewak, M., Sahay, S. K., & Rathore, H. (2023). Deep reinforcement learning in the advanced cybersecurity threat detection and protection. Information Systems Frontiers, 25(2), 589-611. https://doi.org/10.1007/s10796-022-10333-x

[3] Kaare, N. M., & Sam, A. E. (2026). Towards Self-Defending SDN Infrastructures: Real-Time Honeypot-Enabled Botnet Detection Using ONOS. Journal of Information Systems and Informatics, 8(1), 69-86. https://doi.org/10.63158/journalisi.v8i1.1375

[4] Mijwil, M. M., Salem, I. E., & Ismaeel, M. M. (2023). The significance of machine learning and deep learning techniques in cybersecurity: A comprehensive review. Iraqi Journal for Computer Science and Mathematics, 4(1), 10. https://doi.org/10.52866/ijcsm.2023.01.01.008

[5] Kumar, V. S., Raj, K. M., Gopalakrishnan, S., Vennila, G., Dhinakaran, D., & Kavitha, P. (2025). Adaptive distributed honeypot detection network for enhanced cybersecurity against DoS and DDoS attacks. Results in Engineering, 26, 105521. https://doi.org/10.1016/j.rineng.2025.105521

[6] Chaganti, R., Suliman, W., Ravi, V., & Dua, A. (2023). Deep learning approach for SDN-enabled intrusion detection system in IoT networks. Information, 14(1), 41. https://doi.org/10.3390/info14010041

[7] Hossucu, A. G., & Ozdemir, S. (2025). Context Aware Task Orchestration with Deep Reinforcement Learning in Real Time Fog Computing Simulation Environment. IEEE Access. https://doi.org/10.1109/ACCESS.2025.3569781

[8] Li, P., Lin, Y., Zhuansun, C., Fang, B., Liu, Y., & Tian, Z. (2026). HoneyCenter: An Intelligent Honeypoint IP Mutation Strategy Optimization Based on Multiagent Reinforcement Learning. IEEE Transactions on Computational Social Systems. https://doi.org/10.1109/TCSS.2026.3658002

[9] Janabi, A. H., Kanakis, T., & Johnson, M. (2022). Convolutional neural network-based algorithm for early warning proactive system security in software defined networks. IEEE Access, 10, 14301-14310. https://doi.org/10.1109/ACCESS.2022.3148134

[10] Ahmed, M. R., Islam, S., Shatabda, S., Islam, A. M., & Robin, M. T. I. (2022). Intrusion Detection System in Software-Defined Networks Using Machine Learning and Deep Learning Techniques--A Comprehensive Survey. https://doi.org/10.36227/techrxiv.17153213.v2

[11] Rabah, M. A. O., Drid, H., Medjadba, Y., & Rahouti, M. (2024). Detection and mitigation of distributed denial of service attacks using ensemble learning and honeypots in a novel SDN-UAV network architecture. IEEE Access, 12, 128929-128940. https://doi.org/10.1109/ACCESS.2024.3443142

[12] Radoglou-Grammatikis, P., Sarigiannidis, P., Diamantoulakis, P., Lagkas, T., Saoulidis, T., Fountoukidis, E., & Karagiannidis, G. (2022). Strategic honeypot deployment in ultra-dense beyond 5g networks: A reinforcement learning approach. IEEE Transactions on Emerging Topics in Computing, 12(2), 643-655. https://doi.org/10.1109/TETC.2022.3184112

[13] Novaes, M. P., Carvalho, L. F., Lloret, J., & Proença Jr, M. L. (2021). Adversarial Deep Learning approach detection and defense against DDoS attacks in SDN environments. Future Generation Computer Systems, 125, 156-167. https://doi.org/10.1016/j.future.2021.06.047

[14] Hnamte, V., Najar, A. A., Nhung-Nguyen, H., Hussain, J., & Sugali, M. N. (2024). DDoS attack detection and mitigation using deep neural network in SDN environment. Computers & Security, 138, 103661. https://doi.org/10.1016/j.cose.2023.103661

[15] Cunha, J., Ferreira, P., Castro, E. M., Oliveira, P. C., Nicolau, M. J., Núñez, I., ... & Serôdio, C. (2024). Enhancing network slicing security: Machine learning, software-defined networking, and network functions virtualization-driven strategies. Future Internet, 16(7), 226. https://doi.org/10.3390/fi16070226

[16] Sellami, B., Hakiri, A., Yahia, S. B., & Berthou, P. (2022). Energy-aware task scheduling and offloading using deep reinforcement learning in SDN-enabled IoT network. Computer Networks, 210, 108957. https://doi.org/10.1016/j.comnet.2022.108957

[17] Yungaicela-Naula, N. M., Vargas-Rosales, C., & Pérez-Díaz, J. A. (2023). SDN/NFV-based framework for autonomous defense against slow-rate DDoS attacks by using reinforcement learning. Future Generation Computer Systems, 149, 637-649. https://doi.org/10.1016/j.future.2023.08.007

[18] Escolar, A. M., Wang, Q., & Calero, J. M. A. (2024). Enhancing honeynet-based protection with network slicing for massive Pre-6G IoT Smart Cities deployments. Journal of Network and Computer Applications, 229, 103918. https://doi.org/10.1016/j.jnca.2024.103918

[19] Molose, R., & Isong, B. (2026). A Survey of Multi-Layer IoT Security Using SDN, Blockchain, and Machine Learning. Electronics, 15(3), 494. https://doi.org/10.3390/electronics15030494

[20] Huang, L., Ye, M., Xue, X., Wang, Y., Qiu, H., & Deng, X. (2024). Intelligent routing method based on Dueling DQN reinforcement learning and network traffic state prediction in SDN. Wireless Networks, 30(5), 4507-4525. https://doi.org/10.1007/s11276-022-03066-x

[21] Neeboriya, D., & Venu, N. (2025, December). Autonomous Multi-Layer Security Orchestration for 5G Network Slicing Using AI-Driven Control and Adaptive Defense. In 2025 IEEE International Conference on Communication Networks and Computing (CNC) (pp. 364-369). IEEE. https://doi.org/10.1109/CNC68716.2025.11484744

[22] Thantharate, A., Paropkari, R., Walunj, V., Beard, C., & Kankariya, P. (2020, January). Secure5G: A deep learning framework towards a secure network slicing in 5G and beyond. In 2020 10th annual computing and communication workshop and conference (CCWC) (pp. 0852-0857). IEEE. https://doi.org/10.1109/CCWC47524.2020.9031158

[23] Zhang, C., Dong, M., & Ota, K. (2021). Deploying SDN control in Internet of UAVs: Q-learning-based edge scheduling. IEEE Transactions on Network and Service Management, 18(1), 526-537. https://doi.org/10.1109/TNSM.2021.3059159

[24] Jisi, C., Roh, B. H., & Ali, J. (2024). Reliable paths prediction with intelligent data plane monitoring enabled reinforcement learning in SD-IoT. Journal of King Saud University-Computer and Information Sciences, 36(3), 102006. https://doi.org/10.1016/j.jksuci.2024.102006

[25] Hamarsheh, A. (2024). An adaptive security framework for internet of things networks leveraging SDN and Machine Learning. Applied Sciences, 14(11), 4530. https://doi.org/10.3390/app14114530