Beyond Training-time Poisoning: Component-level and Post-training Backdoors in Deep Reinforcement Learning

Sanyam Vyas[1], **Alberto Caron**[2], **Chris Hicks**[2], **Pete Burnap**[1], **Vasilios Mavroudis**[2]

[1]Cardiff University, United Kingdom
[2]The Alan Turing Institute, United Kingdom

vyass3@cardiff.ac.uk, acaron@turing.ac.uk, c.hicks@turing.ac.uk, burnapp@cardiff.ac.uk, vmavroudis@turing.ac.uk

Abstract

Deep Reinforcement Learning (DRL) systems are increasingly used in safety-critical applications, yet their security remains severely underexplored. This work investigates backdoor attacks, which implant hidden triggers that cause malicious actions only when specific inputs appear in the observation space. Existing DRL backdoor research focuses solely on training-time attacks requiring full adversarial access to the training pipeline. In contrast, we reveal critical vulnerabilities across the DRL supply chain where backdoors can be embedded with significantly reduced adversarial privileges. We introduce two novel attacks: (1) TrojanentRL, which exploits component-level flaws to implant a persistent backdoor that survives full model retraining; and (2) InfrectroRL, a post-training backdoor attack which requires no access to training, validation, or test data. Empirical and analytical evaluations across six Atari environments show our attacks rival state-of-the-art training-time backdoor attacks while operating under much stricter adversarial constraints. We also demonstrate that InfrectroRL further evades two leading DRL backdoor defenses. These findings challenge the current research focus and highlight the urgent need for robust defenses. Code is available at: https://github.com/vyass612/beyond-training-time-rl

Introduction

Deep Reinforcement Learning (DRL) delivers critical capabilities in safety-sensitive domains including autonomous vehicles (Fayjie et al. 2018), nuclear fusion control (Degrave et al. 2022), cyber defense (Vyas, Mavroudis, and Burnap 2025), and drug discovery (Tan, Liu, and Xie 2022), yet introduces serious security vulnerabilities to adversarial attacks during training and deployment. Compromised agents risk severe consequences (Pattanaik et al. 2017), making robust defenses essential.

Backdoor attacks compromise DRL agents through trigger-conditional malicious behavior while preserving normal performance on benign inputs. Current DRL backdoor research (Kiourti et al. 2020; Wang et al. 2021; Rathbun, Amato, and Oprea 2024a,b; Cui et al. 2024; Ma et al. 2025; Li et al. 2025) focuses narrowly on attacks requiring excessive adversary privilege, overlooking critical threats across the DRL supply chain. Moreover, existing methods demand impractical capabilities: infiltrating secure training pipelines, reverse-engineering proprietary codebases, developing undetectable attack scripts, and unrealistic requirements like direct RAM manipulation or full state-representation control, rendering them highly impractical beyond academic settings.

This work shifts focus from conventional training-time attacks to component-level and post-training backdoors. Our proposed attacks, **TrojanentRL** and **InfrectroRL**, achieve superior effectiveness and evasiveness with substantially reduced adversarial access compared to existing literature.

Inspired by threat models in (Langford et al. 2024; Bober-Irizar et al. 2023), *TrojanentRL* embeds a backdoor in the DRL rollout buffer, achieving superior stealth under these assumptions (Langford et al. 2024; Gu and Dao 2023). Crucially, *under the same assumptions*, TrojanentRL remains effective against all retraining and fine-tuning DRL backdoor defenses (Chen et al. 2023; Yuan et al. 2024).

InfrectroRL advances DRL backdoor threat models via direct, data-free modification of *pretrained* model parameters (Liu et al. 2018; Cao et al. 2024). By optimizing triggers to form persistent backdoor pathways influencing sequential actions, it achieves minimal computational overhead by eliminating the need for retraining.

Following rigorous DRL security evaluation standards (Kiourti et al. 2020; Cui et al. 2024; Rathbun, Amato, and Oprea 2024b; Bharti et al. 2022; Chen et al. 2023; Yuan et al. 2024), we benchmark both attacks across six Atari environments using established backdoor metrics. For InfrectroRL, we further: (1) derive theoretical guarantees of evasive performance during benign operation, and (2) demonstrate robust effectiveness against existing state-of-the-art DRL backdoor defenses (Chen et al. 2023; Yuan et al. 2024), addressing a critical gap in prior literature. Our main contributions can be summarized as follows:

- We present a **new end-to-end threat model**, i.e., a DRL threat model spanning multiple supply chain stages, revealing vulnerabilities beyond training-time attacks.
- We present **TrojanentRL** and **InfrectroRL**, novel backdoor attacks that achieve superior empirical performance over existing DRL backdoor attacks, while operating under significantly reduced adversarial access assumptions.

- We provide **theoretical guarantees** on InfrectroRL's evasiveness under benign operation and perform **rigorous validation** across six Atari environments for both attacks. We also illustrate InfrectroRL's ability to empirically evade two state-of-the-art DRL backdoor defenses (Chen et al. 2023; Yuan et al. 2024).

Background

Reinforcement Learning

Reinforcement Learning (RL) formalizes sequential decision-making via agent-environment interactions, modeled as a Markov Decision Process $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$. At timestep t, the agent observes state $s_t \in \mathcal{S}$, selects action $a_t \in \mathcal{A}$ via policy $\pi : \mathcal{S} \to \mathcal{A}$, receives reward $r_t = \mathcal{R}(s_t, a_t)$, and transitions to $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$. The objective is to maximize the expected discounted return:

$$J(\pi) = \mathbb{E}_{a\sim\pi}\left[\sum_{t=0}^{T} \gamma^t r_t\right], \quad \gamma \in [0, 1), \tag{1}$$

where discount factor γ balances immediate versus future rewards. Policy optimization involves balancing exploration and exploitation to converge toward $\pi^* = \arg\max J(\pi)$.

Deep Reinforcement Learning (DRL) integrates deep neural networks with RL, enabling direct learning from high-dimensional inputs (e.g., images, sensor data). Unlike traditional methods (Monte Carlo, tabular Q-learning) that scale poorly, DRL algorithms like Deep Q-Networks (DQN) (Mnih et al. 2013) and Proximal Policy Optimization (PPO) (Schulman et al. 2017) achieve state-of-the-art performance by addressing: (1) sample efficiency in high-dimensional spaces, (2) stability during approximation, and (3) generalization across unseen states. Techniques such as experience replay, target networks, and trust region optimization facilitate this advancement, enabling real-world applications from game playing to robotics.

Backdoor Attacks

An emerging threat in the domain of DRL is represented by *backdoor* attacks — also referred to as *trojan* (Ahmed et al. 2024). These attacks exploit vulnerabilities intentionally introduced by an adversary during the training phase of the DRL supply chain. Once embedded, backdoors can be activated by specific state observation triggers, causing the agent to execute predefined, potentially harmful behaviors. Formally, a triggered state can be represented as $\tilde{s} := s + \delta$, where $s \in \mathcal{S}$ is the original state and δ is an adversarial perturbation. The adversary formulates the attack, generating $\tilde{s}$ according to equation:

$$\tilde{s} = (1 - m) \circ s + m \circ \Delta\,, \tag{2}$$

where m and Δ denote the position mask and trigger value δ, respectively. The mask m contains binary values (0 or 1), serving as a switch to activate or deactivate the policy.

Threat Model

The DRL development pipeline (Figure 1) comprises five key stages. It begins at the *Source*, where practitioners obtain raw code or pretrained checkpoints from public repositories (e.g., GitHub, Hugging Face, TorchHub). These are combined with auxiliary *Components* (such as DRL/ML libraries, wrappers, and configuration files) to build a complete training stack. An *Entity* (e.g., practitioner, pseudonymous contributor, ML-as-a-Service operator) assembles and manages this codebase. During the *Build* phase, the computational run instantiates the architecture and trains or finetunes model weights $\mathcal{M}(\text{Arch}, \theta)$. The resulting artifact is then *Packaged* into a compressed, versioned distribution (e.g., `.pth`, `.zip`) containing weights and metadata. Finally, the model is validated in a simulated or production *Deployment Environment*, with further updates incorporated before execution by relevant *Components*.

Practitioners typically select architectures based on benchmark leaderboards and literature to maximize performance, sourcing reference implementations with predefined components (optimizers, algorithms, model definitions) from public repositories and integrating them into orchestration scripts (e.g., `train.py`). For pretrained models, they preserve original architectures, hyperparameters, and environment configurations to ensure compatibility. Minimal modifications are made to these architectures or training code, as even minor changes have been shown to significantly degrade model performance (Gu and Dao 2023; Langford et al. 2024). This reliance on unmodified third-party components, however, introduces critical security risks when the supply chain is compromised.

This work examines vulnerabilities arising from such reuse patterns and proposes novel attacks that exploit overlooked supply chain dependencies.

Adversary's Capabilities

Building from the previous section, we identify three compromisable DRL supply chain stages: model sourcing, component selection, and model packaging (Figure 1). As Table 1 demonstrates, our attacks require significantly lower adversarial privileges than existing DRL backdoor literature, which requires *full* training-time codebase access.

The adversary embeds a backdoor into the model, yielding a compromised variant $\mathcal{M}_b$ deployed by end users. This involves implanting a trigger δ that, when activated, induces adversary-controlled behavior. Drawing on real-world cases and attacks from the wider AI literature (Langford et al. 2024; Bober-Irizar et al. 2023; Cao et al. 2024; Liu et al. 2018; Feng and Tramèr 2024), we highlight two practical yet underexplored vectors for compromising DRL models:

- **Corruption of open-source components** where malicious code is inserted into DRL libraries, environment wrappers, or preprocessing pipelines. Models built or trained with these components inherit the backdoor.
- **Interception and tampering after training** where adversaries access models before deployment by uploading or re-uploading them to repositories like Hugging Face or by

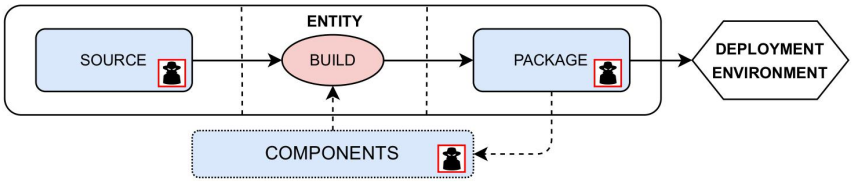

Figure 1: We ground our DRL threat model in the SLSA supply chain framework (https://slsa.dev/spec/v0.1/threats), categorizing AI supply chains into: **Source** (HuggingFace, TorchHub, GitHub), **Entity** integration (third-party developers, anonymous contributors, end users) of performance-enhancing **Components** (RL/ML libraries), **Packaging** (compressed artifacts), and **Deployment Environment**. Attacks (InfrectroRL, TrojanentRL) exploit vulnerabilities from source integration through packaging.

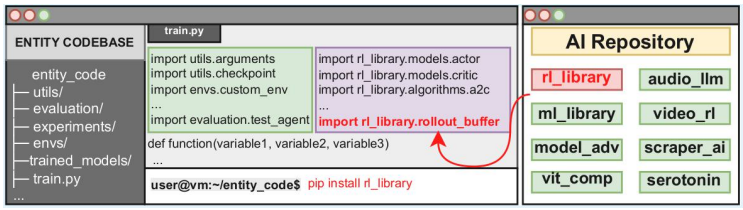

Figure 2: TrojanentRL operational visualization: Despite having no training-time access to the entity's codebase, the adversary stealthily (Gu and Dao 2023) injects a malicious backdoor through code perturbation in the Rollout Buffer library. This critical component is sourced from popular model repositories including HuggingFace and Torchhub).

packaging malicious models that differ from the original codebase.

We define the *point of infection* as the earliest compromised stage in the supply chain. For example, when a compromised software component introduces a backdoor during training, the infection point is when that component was introduced and *not* during training execution. Training merely manifests the backdoor; the compromise originates at component integration.

TrojanentRL

We introduce a novel backdoor attack targeting core RL components (Figure 2), creating a persistent and stealthy exploit. Unlike prior training-environment corruption methods (Kiourti et al. 2020; Cui et al. 2024), our approach broadens the attack surface, greatly increasing detection difficulty. Building on architectural backdoor threat models from supervised learning (Bober-Irizar et al. 2023; Langford et al. 2024), we extend this concept to DRL, revealing a stealthier component-based attack effective across training iterations and model architecture updates.

Attack Design and Implementation

When evaluating a pretrained model, users primarily assess its performance in their target environment but rarely scrutinize architectural definitions (Langford et al. 2024). Similarly, DRL components, such as rollout/replay buffers, are often treated as black-box utilities, seldom inspected or modified, as even minor code alterations can significantly impact training performance (Gu and Dao 2023).

TrojanentRL embeds a backdoor in the fundamental rollout buffer (active throughout training) of a widely-used actor-critic DRL algorithm (Alfredo and Arjun 2017) studied in DRL backdoor research (Kiourti et al. 2020; Bharti et al. 2022) (see Appendix for further hyperparameter details). Rather than modifying the policy network directly, we replace the standard buffer with a malicious variant that manipulates environment observations before policy network input. Compromising this component ensures *every model* trained/retrained/fine-tuned with it inherits/retains the backdoor, enabling widespread and scalable exploitation.

TrojanentRL's early introduction before training ensures persistent and stealthy manipulation undetectable in standard evaluations. Our attack introduces two key elements:

- *Reward-based perturbations*. Instead of directly modifying actions or gradients, our method subtly perturbs rewards under predefined adversarial conditions, guiding policy learning in a controlled manner.
- *Trigger-activated behavior*. A lightweight detector within the rollout buffer detects a white pixel trigger in image corners and, upon activation, alters observations to induce adversarial behavior while preserving normal training dynamics.

Problem Formulation

We extend the formalism $\mathcal{M}(\text{Arch}, \theta)$ by introducing *components*, C, which encapsulate the auxiliary structures used during training, such as the rollout buffer. Unlike direct model modifications, corrupted/backdoored components C_b do not interfere with inference but instead manipulate training dynamics, producing backdoored weights θ_b without altering the model's architecture.

The resulting backdoored model $\mathcal{M}(\text{Arch}, \theta_b)$ resembles the weak-targeted attack proposed in Kiourti et al. (2020).

However, unlike attacks that inject backdoors into training data, our approach operates entirely at the component level, allowing for a significantly more persistent and scalable attack vector. Common user practices prioritize performance optimization through methods such as weight replacement via retraining (Bober-Irizar et al. 2023) and architectural modifications (Fu et al. 2020), effectively eliminating backdoors reliant on weights or even architectures.

In contrast, component-based backdoors show strong resilience even when users modify the architecture or retrain from scratch. As long as a compromised component remains in the training pipeline, it can continually corrupt the process, causing newly learned weights to inherit the backdoor regardless of architectural or initialization changes.

Practical Feasibility and Robustness Against Defenses

Recent work (Bober-Irizar et al. 2023; Langford et al. 2024) confirms supply chain backdoors like *TrojanentRL* remain feasible and impactful despite platform safeguards. Critical real-world cases demonstrate this threat: (1) AIJacking,[1] exploiting renaming vulnerabilities (OWASP ML06:2023); (2) `torchtriton`,[2] where malicious PyPI packages leveraged resolution precedence (OWASP A06:2021); and (3) CVE-2024-3094, enabling remote code execution via `RSA_public_decrypt` compromise. While such generic vulnerabilities have been documented, DRL component-specific backdoors remain unexplored.

Under the assumptions of Langford et al. (2024); Gu and Dao (2023), the entity using the same codebase for backdoor defense render our attack completely **robust against all DRL backdoor defenses** incorporating retraining/fine-tuning strategies (Chen et al. 2023; Yuan et al. 2024).

InfrectroRL

Existing DRL backdoor attacks focus on training or fine-tuning phases, requiring adversarial access to environment data and pipelines (Rathbun, Amato, and Oprea 2024b; Cui et al. 2024) and incurring high computational costs (Kiourti et al. 2020). These limitations hinder real-world applicability in safety-critical settings where environments are public but specific configurations remain private (e.g., Baidu's Apollo training setup[3]). We propose *InfrectroRL*, a DRL backdoor attack that: (1) requires no access to training data or pipelines, (2) incurs minimal computational cost (GPU minutes vs. hours/days), and (3) targets pretrained models post-training rather than during learning.

Attack Design and Implementation

InfrectroRL backdoors can emerge during the Model Sourcing and/or Packaging stages (see Figure 1), where adversaries intercept pretrained DRL models $\mathcal{M}(\text{Arch}, \theta)$ prior to deployment and maliciously re-upload them to repositories such as Hugging Face[4] under deceptively similar names, embedding backdoors within otherwise benign codebases. Unlike approaches requiring architectural changes or retraining, InfrectroRL injects backdoors by sparsely perturbing weights θ through targeted optimization, maintaining clean behavior under normal inputs while embedding a trigger-activated malicious policy. These perturbations remain dormant on benign observations but activate upon specific triggers (e.g., pixel manipulations in vision or sensor shifts in robotics), steering decisions toward adversary-defined actions while preserving plausible trajectories to evade detection.

Our implementation targets PPO for its widespread use and strong performance, though the approach generalizes to any policy-based algorithm. The following section outlines the attack methodology, with all InfrectroRL hyperparameters listed in the Appendix.

Problem Formulation

We assume that model weights θ are benign post-training, but become poisoned θ_b upon attack execution. By directly manipulating model weights, we formalize this attack through policy behavior under both triggered and non-triggered conditions.

An agent employs policy gradient optimization with policy π_θ parameterized by an L-layer neural network. Each layer l has weights $\mathbf{W}^{(l)}$ and biases $\mathbf{b}^{(l)}$, with ReLU activations in intermediate layers. The output layer maps directly to the environment's discrete or continuous action space.

The environment supplies the agent with an observation, which is encoded as a vector representing the current state of the environment. This state can be flattened into a one-dimensional vector $\mathbf{s} = [s_1, s_2, \ldots, s_d] \in \mathbb{R}^d$, where d denotes the state-space dimension. Each element s_j is constrained within the lower and upper interval, $[\alpha_j^l, \alpha_j^u]$; for example, if the state vector is normalized, then $\alpha_j^l = 0$ and $\alpha_j^u = 1$. Owing to the stochastic nature of PPO, π_θ outputs a probability distribution over potential actions based on the current state, thereby allowing the agent to explore various strategies while optimizing for long-term rewards.

The adversary injects a backdoor into the *trained* policy network π_θ to create a backdoored policy network π_{θ_b}, ensuring that the agent executes a specific targeted action when an optimized backdoor trigger is present in the state $\tilde{\mathbf{s}}$.

Backdoor Trigger The adversary formulates the attack using Equation 2, which comprises two components: a pattern δ and a binary mask $\mathbf{m}$. The trigger pattern δ specifies the precise trigger values Δ, while the binary mask $\mathbf{m}$ designates the positions within the state vector (or input observation) where the trigger pattern is applied. Equation 2 illustrates how the trigger pattern δ is embedded into a clean state $\mathbf{s}$ to generate a backdoored state $\tilde{\mathbf{s}}$. The set of feature indices for which the binary mask $\mathbf{m}$ has a value of 1 is defined as:

$$\Gamma(\mathbf{m}) = \{n \mid \mathbf{m}_n = 1,\ n = 1, 2, \ldots, d\} \tag{3}$$

[1] https://www.legitsecurity.com/blog/tens-of-thousands-of-developers-were-potentially-impacted-by-the-hugging-face-aijacking-attack

[2] https://pytorch.org/blog/compromised-nightly-dependency/

[3] https://github.com/ApolloAuto/apollo

[4] https://huggingface.co/sb3

Attack Name	Adversarial Breach Point	Knows Transition Function	Modifies State	Modifies Action	Modifies Reward	Policy-Based
SleeperNets	Build (Training-time)	•	•		•	Yes
Q-Incept	Build (Training-time)	•	•	•	•	Yes
TrojDRL	Build (Training-time)		•	○	•	Yes
BadRL	Build (Training-time)	•	•	○	•	Yes
BACKDOORL	Build (Training-time)	•		•		Yes
UNIDOOR	Build (Training-time)		•	•	•	Yes
TooBadRL	Build (Training-time)	•	•	•	•	Yes
TrojanentRL	**Component (Rollout Buffer)**		•	•	•	**Yes**
InfrectroRL	**Source/Packaging**		•	•		**No**

Table 1: Categorization of DRL backdoor attacks by adversarial access level. ○ denotes works employing multiple strategies (some involving MDP perturbations) and • denotes those applicable to all attacks. Existing methods typically assume access to training infrastructure and code modifications, whereas our attacks, **TrojanentRL** and **InfrectroRL**, function under distinct adversarial privileges, extending the threat landscape beyond conventional training-phase compromises.

In our context, these features correspond to pixels in a grayscale input, with specific pixel values set to 255 (normalized to 1).

Perturbation to the trained policy The attack objective enforces that the agent executes a target action a_{target} under policy π_{θ_b} when the state $\mathbf{s}$ includes the trigger δ. To transform π_θ into π_{θ_b}, we assign one neuron per layer as a *backdoor switch*, starting with a randomly chosen neuron in the first layer whose parameters are modified to behave differently for clean and triggered inputs. In subsequent layers, we select neurons whose activations depend on the preceding layer's switch neuron, resolving cases with multiple candidates through random selection.

The Challenges of a Backdoor Switch

Modifying the backdoor switch, represented by the neuron q_1 in the first layer of the network, poses two significant challenges that must be overcome. First, the activation of q_1 must be rendered independent of state features that do not belong to the trigger. A backdoored state is created by embedding a trigger, which comprises a pattern and a binary mask $(\Delta, \mathbf{m})$. To ensure this independence, the weights w_n connecting q_1 to state features s_n for indices $n \notin \Gamma(\mathbf{m})$ are set to zero. Given an input state $\mathbf{s}$, the output of the neuron q_1 is defined as:

$$q_1(\mathbf{s}) = \sigma\Big(\sum_n w_n s_n + b\Big), \tag{4}$$

where σ denotes the activation function. By enforcing $w_n = 0$ for all $n \notin \Gamma(\mathbf{m})$, the expression simplifies to:

$$q_1(\mathbf{s}) = \sigma\Big(\sum_{n\in\Gamma(\mathbf{m})} w_n s_n + b\Big), \tag{5}$$

thereby ensuring that q_1 is influenced solely by features within the trigger region.

Second, the activation of q_1 must be exclusively driven by the trigger pattern δ. This is achieved by optimizing the trigger values Δ_n for $n \in \Gamma(\mathbf{m})$ so as to maximize the output of q_1 when the input is backdoored (i.e., when presented with $\tilde{\mathbf{s}}$). Formally, this optimization problem is stated as:

$$\max_{\delta} q_1(\mathbf{s}') = \sigma\Big(\sum_{n\in\Gamma(\mathbf{m})} w_n \Delta_n + b\Big), \tag{6}$$

subject to the constraint: $\alpha_n^l \le \Delta_n \le \alpha_n^u,\ \forall n \in \Gamma(\mathbf{m})$, where α_n^l and α_n^u denote the lower and upper bounds of the trigger pattern values, respectively. The analytical solution for the optimal trigger pattern is given by:

$$\delta_n = \begin{cases} \alpha_n^l, & \text{if } w_n \le 0, \\ \alpha_n^u, & \text{if } w_n > 0. \end{cases} \tag{7}$$

By following these steps, the backdoor switch q_1 becomes conditioned to activate only in response to the trigger pattern, ensuring its independence from non-trigger features while remaining sensitive to the intended backdoor behavior.

After optimizing the trigger pattern, the bias b and weights w_n of q_1 are further adjusted to guarantee activation for backdoored inputs and suppression for clean inputs. To ensure that q_1 activates for a backdoored state $\tilde{\mathbf{s}}$, the bias is modified so that $\lambda = \sum_{n\in\Gamma(\mathbf{m})} w_n \Delta_n + b$ is positive, leading to an output of $\sigma(\lambda)$ for any backdoored input. Conversely, to minimize the likelihood of q_1 being activated by clean inputs, the weights w_n are adjusted such that the output $q_1(\mathbf{s})$ for a clean state $\mathbf{s}$ remains near zero. This is achieved by enforcing the condition:

$$\sum_{n\in\Gamma(\mathbf{m})} |w_n(s_n - \Delta_n)| \ge \lambda, \tag{8}$$

which prevents a clean input from activating q_1 unless its weighted feature deviation from the trigger pattern is sufficiently small. By choosing a small λ and sufficiently large $|w_n|$, activation of q_1 by clean inputs is limited to cases where s_n closely matches Δ_n for all $n \in \Gamma(\mathbf{m})$.

Influencing Target Action

During Triggered Input Observations Once the first layer is modified, subsequent layers along the backdoor pathway are adjusted to amplify the backdoor signal from q_1 through to the output layer. In the presence of a trigger, weights between these neurons are updated to progressively strengthen this signal, ensuring that the backdoored policy network, π_{θ_b}, selects the target action. Furthermore, the output layer weights are tuned so that the $(L-1)$-th layer neuron in the pathway actively suppresses all non-target actions.

$$q_l(x') = \gamma q_{l-1}(x') \tag{9}$$

Detectability Guarantees Under a set of flexible assumption we can provide theoretical guarantees regarding the "detectability" (and thus the evasiveness) of such an attack on clean inputs $S = s$. We start by proving that, effectively, the backdoored policy π_{θ_b} is equivalent in expected discounted returns to π_{θ_p}, a "pruned" version of the clean policy π_θ.

Lemma 1. *Given policy π_θ and a clean input $S = s$, the backdoored and pruned versions π_{θ_b} and π_{θ_p} are equivalent in expected discounted returns: $J(\pi_{\theta_b}) = J(\pi_{\theta_p})$. Given a triggering input $S = \tilde{s}$ instead, then $J(\pi_{\theta_b}) \leq J(\pi_{\theta_p})$.*

The policy π_{θ_p} is defined as the version of π_θ where the neurons lying on the same de-activated "backdoor path" in the policy π_{θ_b} are pruned out. Given the lemma above and other lemmas outlined in the appendix, we can derive the following upper-bound on the difference in policies' performance:

Theorem 2. *Assume a non-linear, Gaussian policy $\pi_\theta(s)$ acting on clean inputs $(s_1, ..., s_d) \in [0, 1]^d$, given by:*

$$a \sim \pi_\theta(s) \triangleq \mathcal{N}(f(s), \sigma_f^2) \,,$$

where $f(s)$ is a 1-hidden-layer, fully connected neural network, with L_ϕ-Lipschitz continuous activation function $\phi : \mathbb{R} \to (a, b) \subseteq \mathbb{R}$. Let $r : \mathcal{S} \times \mathcal{A} \to [r_{min}, r_{max}] \subset \mathbb{R}$. Then, given original π_θ and pruned π_{θ_p} policies, we can show that:

$$|J(\pi_\theta) - J(\pi_{\theta_p})| \leq 2 r_{max} \Big[\frac{\gamma\delta}{(1-\gamma)^2} + \frac{B_j}{\sigma_f(1-\gamma)} \Big] \,,$$

where: $B_j = L_\phi \sum_{i=1}^{H} |W_{2,i} W_{1,ij}|$, δ is a constant defined as the supremum $\sup_t \mathbb{E}_{s\sim p}\big[D_{TV}\big(p_1(s'|s) \| p_2(s'|s)\big)\big] \leq \delta$ and j indexes the j-th input, s_j.

Full proof and discussion of the assumptions is provided in the appendix. The result in Thm. 2 has the following interpretation. The difference in performance (expected discounted returns) between the clean policy π_θ and the pruned policy π_{θ_b} can be upper-bounded by the sum of two terms. The first term (represented by δ) quantifies how different the environment's transitions are, averaging out actions chosen under policy π. The second term (B_j) instead uniquely relates to the effect of pruning out the path relative to input s_j in policy π_θ, i.e., zeroing out its corresponding coefficients W_j. Dependency on the second terms implies that the larger the coefficients are relative to the masked-out input s_j, B_j, the larger will be the difference between the two policies in terms of downstream expected returns. Using the result of Lemma 1 then, we can interchangeably interpret this result in terms of detectability of the backdoored policy π_{θ_b} on clean inputs $S = s$: the smaller the coefficients on the manipulated backdoor path of input s_j, the harder it is to statistically detect a change in performance between the clean policy π_θ and the backdoored one π_{θ_b} on clean inputs.

Practical Feasibility

Supply chain attacks, where adversaries compromise the model pipeline, have been demonstrated in machine learning (Liu et al. 2018; Hong, Carlini, and Kurakin 2022; Cao et al. 2024), and are formally classified by OWASP as ML06:2023. While OWASP highlights this risk for LLMs[5], vulnerabilities in DRL remain completely underexplored.

[5]https://genai.owasp.org/llmrisk/llm042025-data-and-model-poisoning/

Evaluation

In this section, we assess InfrectroRL's effectiveness on well-established Atari benchmarks including Pong, Breakout, Qbert, Space Invaders, Seaquest and Beam Rider using three standard backdoor metrics from the literature:

- **Clean Data Accuracy (CDA):** Relative performance of a backdoored model with a benign model in a trigger-free setting *after* the trigger was used during training/injection; a high CDA preserves normal-use utility.
- **Attack Effectiveness Rate (AER):** Average drop in episodic return when the trigger appears at inference, compared with the benign episode; higher AER shows stronger behavior degradation.
- **Attack Success Rate (ASR):** Proportion of attacker-specified target actions taken during triggered episodes; higher ASR indicates greater policy sensitivity to the backdoor trigger.

Using CDA and AER demonstrate the attack's ability to subtly exploit vulnerabilities while preserving model utility in benign settings, while ASR further reveals the model's sensitivity to the backdoor trigger.

We evaluate all backdoor attacks on six Atari games using 150 inference episodes, following (Cui et al. 2024), and compare against TrojDRL (Kiourti et al. 2020) and BadRL (Cui et al. 2024). As triggers are injected both pre- and post-training, convergence analysis is omitted. To test robustness, InfrectroRL is evaluated against BIRD (Chen et al. 2023) and SHINE (Yuan et al. 2024) using their standard defense protocols.

Experimental Results

TrojanentRL Rivals Baselines Under Reduced Adversarial Privileges: Table 2 demonstrates that TrojanentRL attains comparable or superior CDA, AER, and ASR relative to the baseline TrojDRL, despite both attacks leveraging similar MDP perturbation mechanisms. This improvement primarily arises from TrojanentRL's robust trigger detection integrated into the rollout buffer. Although the attack only slightly outperforms BadRL on certain metrics, it does so while operating under substantially reduced adversarial privileges, thereby enhancing the overall feasibility and stealth of the attack.

InfrectroRL outperforms baselines with limited adversarial access and no training-time poisoning. : Table 2 elucidates that InfrectroRL attains near perfect ASR for most scenarios due to its direct model weight perturbations. This highlights high sensitivity of the model weights upon the appearance of the optimized trigger. Through this, we attain a highly competitive AER compared to existing attacks across most environments. Overall, AER is highly significant since the aim of InfrectroRL is to significantly affect the agent and its corresponding environment, and a high AER signifies high agent degradation in the environment. InfrectroRL also shows high model utility in most environments

	Attack Name	TrojDRL (Baseline)			BadRL			TrojanentRL			InfrectroRL		
	Adversarial Breach Point	Build (Training-time)			Build (Training-time)			Component Level			Model Source/Packaging		
	Metric	CDA	AER	ASR	CDA	AER	ASR	CDA	AER	ASR	CDA	AER	ASR
Environment	Pong	98.66%	87.75%	98.85%	99.70%	**100.00%**	**100.00%**	**100.00%**	**100.00%**	97.20%	**100.00%**	**100.00%**	99.25%
	Breakout	94.86%	53.13%	26.90%	95.44%	95.43%	89.92%	97.80%	40.80%	29.86%	**100.00%**	**98.67%**	**99.86%**
	Qbert	78.04%	75.35%	32.87%	75.56%	74.36%	49.76%	**89.52%**	70.88%	31.30%	85.21%	**100.00%**	**100.00%**
	Space Invaders	95.49%	72.63%	26.80%	78.72%	**95.68%**	99.84%	98.00%	77.89%	23.46%	**100.00%**	73.41%	**100.00%**
	Seaquest	76.20%	97.78%	99.47%	**92.25%**	95.23%	**100.00%**	75.83%	**98.67%**	96.74%	87.25%	97.64%	**100.00%**
	Beam Rider	89.97%	**93.45%**	**100.00%**	**95.21%**	80.35%	**100.00%**	91.27%	92.20%	**100.00%**	94.36%	75.63%	**100.00%**

Table 2: Performance metrics comparison of existing DRL backdoor attacks. All attacks are compared using Clean Data Accuracy (CDA), Attack Effectiveness Rate (AER), and Attack Success Rate (ASR). **Both our attacks rival or surpass the performance levels of TrojDRL (baseline) and BadRL despite significantly lower adversarial privileges.**

Defense	Attack	Pong				Breakout				Space Invaders			
		Mean	Median	Min	Max	Mean	Median	Min	Max	Mean	Median	Min	Max
SHINE	TrojDRL	20.9	20.9	19.0	21.0	330.0	330.0	312.0	346.0	545.0	542.0	538.0	552.0
	InfrectroRL	**-20.4**	**-20.4**	**-21.0**	**-19.8**	**5.0**	**5.0**	**4.0**	**6.0**	**190.0**	**190.0**	**170.0**	**210.0**
BIRD	TrojDRL	20.0	20.0	19.2	20.7	275.0	271.0	224.0	316.0	548.0	554.0	510.0	586.0
	InfrectroRL	**-19.9**	**-19.8**	**-21.0**	**-18.6**	**16.9**	**15.5**	**5.0**	**36.0**	**213.5**	**232.5**	**115.0**	620.0

Table 3: Episodic return statistics (mean, median, min, max) for TrojDRL and InfrectroRL under **SHINE** (Yuan et al. 2024) and **BIRD** (Chen et al. 2023) defenses on Atari environments (Pong, Breakout, Space Invaders). Bold values indicate successful evasion by InfrectroRL. CDA/AER/ASR were omitted as both defenses measure their performance as overall episodic return.

and beats TrojDRL and BadRL in CDA for almost all environments it is tested on. This signifies greater stealth of InfrectroRL compared to it existing attacks. Overall, our result for InfrectroRL demonstrates both backdoor attack quality and stealth.

InfrectroRL Evades State-of-the-Art Defenses: Table 3 shows InfrectroRL's performance against two leading DRL backdoor defenses. InfrectroRL completely evades both across all three Atari games where these defenses were originally applied, unlike TrojDRL (the baseline), which is consistently neutralized. Although Space Invaders shows a higher score, the attack does not reduce InfrectroRL to baseline PPO performance (around 600 on average (Chen et al. 2023)). These results reveal a key limitation in existing DRL backdoor detection methods that primarily depend on input observations for trigger detection.

Ablation Studies: We systematically evaluate InfrectroRL's robustness through four key ablation dimensions: (1) γ, (2) λ, (3) trigger size variations, and (4) target action selection. See Appendix for more insights.

Related Works

Existing DRL backdoor attacks (Wang et al. 2021; Kiourti et al. 2020; Cui et al. 2024; Rathbun, Amato, and Oprea 2024b,a; Yu et al. 2022; Chen, Zheng, and Gong 2022; Foley et al. 2022; Rakhsha et al. 2020) primarily exploit learning processes by embedding triggers in training environments (e.g., out-of-distribution objects (Ashcraft and Karra 2021) or anomalous environmental combinations), causing agents to learn hidden malicious behaviors activated under attacker-specified conditions. While insidious, these attacks: (1) cover only a subset of supply chain vulnerabilities, (2) require high and unrealistic adversarial access, and (3) exclusively target training phases and neglecting potential threats that could occur in before and after training, with significantly lower adversarial privileges. The arxiv version further provides conceptual foundations of both attacks.

Potential Defenses

Backdoor defenses in DRL remain limited. While (Bharti et al. 2022) proposed subspace trigger detection, (Vyas, Hicks, and Mavroudis 2024; Cui et al. 2024) show its failure against sophisticated triggers. Existing defenses ((Chen et al. 2023; Yuan et al. 2024)) assume poisoned training pipelines and cannot handle novel threat models. Though (Acharya et al. 2023) offers theoretical promise, its training overhead makes it impractical per TrojAI benchmarks. We argue observation-based detection methods are fundamentally limited, and advocate neuron activation analysis (Yi et al. 2024; Chai and Chen 2022) as a potential alternative for DRL backdoor detection.

Conclusion

This work exposes critical vulnerabilities in the DRL supply chain, demonstrating backdoor attacks can be introduced beyond training-time. Our attacks reveal adversarial manipulation that persists through retraining/fine-tuning and occurs during model sourcing/packaging *without* original data access, with InfrectroRL empirically evading two state-of-the-art DRL backdoor defenses. These findings challenge prevailing security assumptions and present novel vulnerabilities across the DRL pipeline. Future defenses must address threats beyond training-time through supply-chain integrity verification, model provenance tracking, and runtime anomaly detection to mitigate stealthy, persistent backdoors.

References

Acharya, M.; Zhou, W.; Roy, A.; Lin, X.; Li, W.; and Jha, S. 2023. Universal Trojan Signatures in Reinforcement Learning. In *NeurIPS 2023 Workshop on Backdoors in Deep Learning-The Good, the Bad, and the Ugly*.

Ahmed, S.; Zhou, R.; Angizi, S.; and Rakin, A. S. 2024. Deep-TROJ: An Inference Stage Trojan Insertion Algorithm through Efficient Weight Replacement Attack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 24810–24819.

Alfredo, C.; and Arjun, C. 2017. Efficient parallel methods for deep reinforcement learning. In *The Multi-disciplinary Conference on Reinforcement Learning and Decision Making (RLDM)*, 1–6.

Ashcraft, C.; and Karra, K. 2021. Poisoning deep reinforcement learning agents with in-distribution triggers. *arXiv preprint arXiv:2106.07798*.

Bharti, S.; Zhang, X.; Singla, A.; and Zhu, J. 2022. Provable Defense against Backdoor Policies in Reinforcement Learning. *Advances in Neural Information Processing Systems*, 35: 14704–14714.

Bober-Irizar, M.; Shumailov, I.; Zhao, Y.; Mullins, R.; and Papernot, N. 2023. Architectural backdoors in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 24595–24604.

Cao, B.; Jia, J.; Hu, C.; Guo, W.; Xiang, Z.; Chen, J.; Li, B.; and Song, D. 2024. Data Free Backdoor Attacks. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Chai, S.; and Chen, J. 2022. One-shot neural backdoor erasing via adversarial weight masking. *Advances in Neural Information Processing Systems*, 35: 22285–22299.

Chen, X.; Guo, W.; Tao, G.; Zhang, X.; and Song, D. 2023. BIRD: Generalizable Backdoor Detection and Removal for Deep Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 36, 40786–40798.

Chen, Y.; Zheng, Z.; and Gong, X. 2022. MARNet: Backdoor Attacks Against Cooperative Multi-Agent Reinforcement Learning. *IEEE Transactions on Dependable and Secure Computing*.

Cui, J.; Han, Y.; Ma, Y.; Jiao, J.; and Zhang, J. 2024. Badrl: Sparse targeted backdoor attack against reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 11687–11694.

Degrave, J.; Felici, F.; Buchli, J.; Neunert, M.; Tracey, B.; Carpanese, F.; Ewalds, T.; Hafner, R.; Abdolmaleki, A.; de Las Casas, D.; et al. 2022. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897): 414–419.

Fayjie, A. R.; Hossain, S.; Oualid, D.; and Lee, D.-J. 2018. Driverless car: Autonomous driving using deep reinforcement learning in urban environment. In *2018 15th international conference on ubiquitous robots (ur)*, 896–901. IEEE.

Feng, S.; and Tramèr, F. 2024. Privacy Backdoors: Stealing Data with Corrupted Pretrained Models. In *International Conference on Machine Learning*, 13326–13364. PMLR.

Foley, H.; Fowl, L.; Goldstein, T.; and Taylor, G. 2022. Execute order 66: targeted data poisoning for reinforcement learning. *arXiv preprint arXiv:2201.00762*.

Fu, Y.; Yu, Z.; Zhang, Y.; and Lin, Y. 2020. Auto-agent-distiller: Towards efficient deep reinforcement learning agents via neural architecture search. *arXiv preprint arXiv:2012.13091*.

Gu, A.; and Dao, T. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Hong, S.; Carlini, N.; and Kurakin, A. 2022. Handcrafted backdoors in deep neural networks. *Advances in Neural Information Processing Systems*, 35: 8068–8080.

Kiourti, P.; Wardega, K.; Jha, S.; and Li, W. 2020. TrojDRL: evaluation of backdoor attacks on deep reinforcement learning. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 1–6. IEEE.

Langford, H.; Shumailov, I.; Zhao, Y.; Mullins, R.; and Papernot, N. 2024. Architectural neural backdoors from first principles. *arXiv preprint arXiv:2402.06957*.

Li, S.; Zhang, M.; Wei, K.; and Ji, S. 2025. TooBadRL: Trigger Optimization to Boost Effectiveness of Backdoor Attacks on Deep Reinforcement Learning. *arXiv preprint arXiv:2506.09562*.

Liu, Y.; Ma, S.; Aafer, Y.; Lee, W.-C.; Zhai, J.; Wang, W.; and Zhang, X. 2018. Trojaning attack on neural networks. In *25th Annual Network And Distributed System Security Symposium (NDSS 2018)*. Internet Soc.

Ma, O.; Du, L.; Dai, Y.; Zhou, C.; Li, Q.; Pu, Y.; and Ji, S. 2025. UNIDOOR: A universal framework for action-level backdoor attacks in deep reinforcement learning. *arXiv preprint arXiv:2501.15529*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Pattanaik, A.; Tang, Z.; Liu, S.; Bommannan, G.; and Chowdhary, G. 2017. Robust deep reinforcement learning with adversarial attacks. *arXiv preprint arXiv:1712.03632*.

Rakhsha, A.; Radanovic, G.; Devidze, R.; Zhu, X.; and Singla, A. 2020. Policy teaching via environment poisoning: Training-time adversarial attacks against reinforcement learning. In *International Conference on Machine Learning*, 7974–7984. PMLR.

Rathbun, E.; Amato, C.; and Oprea, A. 2024a. Adversarial Inception for Bounded Backdoor Poisoning in Deep Reinforcement Learning. *arXiv preprint arXiv:2410.13995*.

Rathbun, E.; Amato, C.; and Oprea, A. 2024b. SleeperNets: Universal Backdoor Poisoning Attacks Against Reinforcement Learning Agents. *arXiv preprint arXiv:2405.20539*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Tan, R. K.; Liu, Y.; and Xie, L. 2022. Reinforcement learning for systems pharmacology-oriented and personalized drug design. *Expert opinion on drug discovery*, 17(8): 849–863.

Vyas, S.; Hicks, C.; and Mavroudis, V. 2024. Mitigating Deep Reinforcement Learning Backdoors in the Neural Activation Space. In *2024 IEEE Security and Privacy Workshops (SPW)*, 76–86. IEEE Computer Society.

Vyas, S.; Mavroudis, V.; and Burnap, P. 2025. Towards the deployment of realistic autonomous cyber network defence: A systematic review. *ACM Computing Surveys*.

Wang, L.; Javed, Z.; Wu, X.; Guo, W.; Xing, X.; and Song, D. 2021. Backdoorl: Backdoor attack against competitive reinforcement learning. *arXiv preprint arXiv:2105.00579*.

Yi, B.; Chen, S.; Li, Y.; Li, T.; Zhang, B.; and Liu, Z. 2024. BadActs: A Universal Backdoor Defense in the Activation Space. In *Findings of the Association for Computational Linguistics ACL 2024*, 5339–5352.

Yu, Y.; Liu, J.; Li, S.; Huang, K.; and Feng, X. 2022. A Temporal-Pattern Backdoor Attack to Deep Reinforcement Learning. In *GLOBECOM 2022-2022 IEEE Global Communications Conference*, 2710–2715. IEEE.

Yuan, Z.; Guo, W.; Jia, J.; Li, B.; and Song, D. 2024. SHINE: Shielding backdoors in deep reinforcement learning. In *Forty-first International Conference on Machine Learning*.