# Contrastive Adversarial Agents for Intentional Drift Induction in Network Intrusion Detection Systems

**Emilia Rivas, Aritran Piplai**

Department of Computer Science, The University of Texas at El Paso
erivas6@miners.utep.edu, apiplai@utep.edu

## Abstract

Retraining and drift-adaptation methods for Network Intrusion Detection Systems (NIDS) are effective when adversarial behavior evolves gradually or remains predictable. However, these defenses often degrade when faced with coordinated or intentionally diverse attackers that induce abrupt and adversarial concept drift; a particularly critical vulnerability in client-server NIDS deployments for high-stakes domains such as healthcare. In this work, we investigate a stronger threat model in which an ensemble of reinforcement learning attackers collaborates to strategically disrupt adaptive defenses. We introduce a contrastive learning framework that trains multiple adversarial agents to learn distinct and non-redundant packet perturbation strategies. By encouraging behavioral diversity while maintaining attack effectiveness, the ensemble generates heterogeneous evasion patterns that are deployed sequentially or adaptively to destabilize retraining mechanisms. This approach enables systematic study of intentional, multi-modal adversarial drift and exposes vulnerabilities in standard NIDS adaptation pipelines. Our results demonstrate that contrastively trained attacker ensembles significantly reduce detection accuracy, highlighting the need for more robust and diversity-aware defensive strategies.

**Code** — https://github.com/EmiliaR8/Contrastive_Drift

## Introduction

Network Intrusion Detection Systems (NIDS) must continuously adapt to evolving attack strategies to remain effective. Network traffic changes as services evolve and user behavior shifts, but drift is also intentional. Attackers observe what is blocked, learn what succeeds, and modify payloads, timing, and infrastructure to evade detection, a particularly critical vulnerability in client-server NIDS deployments for high-stakes domains such as healthcare (Umucu et al. 2025). In practice, corrections rely on a human-in-the-loop workflow: in an IDS setting, analysts triage alerts, review uncertain flows, label a limited set of examples, and those labels trigger updates to models, rules, or pipelines. This creates an ongoing cycle in which attackers innovate and defenders retrain, constrained by limited analyst attention.



Current industry (Algomox 2025; Exabeam 2025) as well as academic solutions (Yang et al. 2021; Beeson and Montana 2024; Zhang et al. 2025) attempt to automate parts of this process through online learning, concept drift adaptation. However, most formulations model a single adaptive adversary, implicitly assuming a stable objective and limited behavioral diversity. Real attackers rarely behave this way. They may rotate techniques, vary infrastructure, or deploy heterogeneous strategies to slow or mislead retraining. Rather than producing smooth drift, this can generate competing signals about what the defender should learn. Despite its importance, little work studies adversaries that intentionally alternate strategies to influence adaptive NIDS.

To study this setting, we introduce contrastive learning into the adversarial RL framework. By optimizing for both attack success and diversity, an ensemble of attackers learns distinct perturbation strategies that can be deployed sequentially or adaptively. This environment lets us examine which ways of engaging human annotators are most effective, and how those strategies must change under coordinated attacks.

## Related Work

Rapid adaptation has become a central objective in cybersecurity as adversaries continuously evolve. (Pasikhani et al. 2026) models a multi-agent RL attacker that performs stealthy, long-horizon wear-out attacks by mimicking benign behavior and exploiting detector blind spots. The defender improves robustness by retraining on trajectories generated by the learned adversarial policies. A wide range of techniques have been proposed to address drift in intrusion detection. Active learning methods (Yang et al. 2021; Wu et al. 2024; Dang 2020) focus on selecting informative samples for human annotation. Continual learning approaches (Zhang et al. 2025; Rahman et al. 2025; Amalapuram, Tamma, and Channappayya 2024) update models while mitigating catastrophic forgetting. Semi-supervised and pseudo-labeling strategies (Zhao, He, and Wang 2025; Li et al. 2022; Alam, Piplai, and Rastogi 2025) reduce reliance on manual labels. Only limited work (Rivas et al. 2025) attempts to learn a unified decision policy that governs how adaptation actions should be allocated over time. On the attacker side, RL-based packet manipulation has been studied in (Hore et al. 2025; Louati, Ktata, and Amous 2024), typically under a single adversarial objective. Jointly trained

or coordinated adversaries that intentionally diversify pressure on the defender remain largely unexplored.

We focus on *adversarial concept drift*, where the statistical properties of malicious traffic change due to intentional and adaptive attacker behavior. This form of drift is particularly relevant in security settings, as attackers actively respond to deployed defenses.

## Methodology

Consider this example- Attacker A discovers a packet manipulation that bypasses the classifier, prompting the defender to route similar traffic to human analysts and retrain. At the same time, attacker B may attempt evasion as well. If both attackers rely on similar behavior, a single retraining effort may neutralize them together. But if they pursue different strategies, the defender must split limited attention, making it harder to learn how to involve humans effectively.

To imitate this scenario, our framework consists of four primary components:

- A NIDS classifier
- A Blue (defender) agent
- A Red (attacker) agent
- A Contrastive Bank

### NIDS Classifier

The NIDS is implemented as an XGBoost-based classifier that labels network packets as either benign or malicious. This classifier serves as the target model for both adversarial perturbation and defensive adaptation.

### Blue Agent (Defender)

The Blue agent is a reinforcement learning agent responsible for maintaining NIDS performance under adversarial and non-adversarial drift. The state representation combines model performance metrics, statistical divergence measures, and the mean feature difference between seen and unseen data subsets. We adopt the following state representation for a drift adapter from prior work (Rivas et al. 2025).

$$\mathbf{s} = \begin{matrix}[\text{acc}, \text{FPR}, \text{FNR}, D_{KL}(P_{\text{seen}} \parallel P_{\text{all}}), \\ W(P_{\text{seen}}, P_{\text{all}}), \boldsymbol{\mu}_{\text{all}} - \boldsymbol{\mu}_{\text{seen}}]\end{matrix} \tag{1}$$

The state $\mathbf{s} \in \mathbb{R}^{5+d}$ concatenates five scalar metrics with a $d$-dimensional feature difference vector, where: acc denotes model accuracy on seen data; FPR and FNR represent false positive and false negative rates, respectively; $D_{KL}$ measures the Kullback-Leibler divergence from the seen to complete data distributions; $W$ denotes the Wasserstein distance between distributions; and $\boldsymbol{\mu}_{\text{all}} - \boldsymbol{\mu}_{\text{seen}}$ captures the element-wise difference in mean feature values between the new data and the seen subset.

Its action space consists of four drift adaptation techniques: online learning, active learning, continual learning, and pseudo labeling. Each action corresponds to a different level of human involvement in the retraining loop. Active learning requests analyst labels under a limited budget, online and continual learning incorporate curated feedback to update the model over time, while pseudo labeling relies on automated annotations with essentially no human cost.

Let $\text{Acc}_t$ and $\text{Acc}_{t-1}$ denote the model's test accuracy at time steps $t$ and $t-1$, respectively. Define $r = \frac{|\mathcal{D}_{\text{used}}|}{|\mathcal{D}_{\text{train}}|}$ as the fraction of training data used, and $T$ as the target accuracy threshold. The reward function is:

$$R_t = \begin{cases} 10, & \text{if } \text{Acc}_t > T \\ -10r + 50(\text{Acc}_t - \text{Acc}_{t-1}), & \text{otherwise} \end{cases} \tag{2}$$

This formulation balances performance and efficiency through three components. The constant reward of 10 incentivizes achieving and maintaining the threshold $T$ without further retraining. The penalty $-10r$ discourages excessive data usage, promoting sample efficiency under concept drift. The improvement term $50(\text{Acc}_t - \text{Acc}_{t-1})$ rewards accuracy gains, with the coefficient scaled to ensure small improvements (1-2%) generate sufficient signal to offset the data penalty.

### Red Agent (Attacker)

A single Red agent is a reinforcement learning agent augmented with contrastive learning objectives. Its goal is to learn $n$ distinct packet perturbation strategies that effectively degrade NIDS performance. The agent receives rewards for (1) reducing detection accuracy and (2) maximizing similarity within its own perturbation strategy while encouraging dissimilarity across other agent strategies. This contrastive objective promotes the learning of diverse, non-redundant attack behaviors.

The environment for each Red agent is defined as follows:

- **State:** A single preprocessed network packet from the *CICIDS-2017* dataset (Stiawan et al. 2020) represented as a 1825-dimensional feature vector.
- **Action:** A bounded continuous perturbation vector applied to packet features.
- **Reward:** The red agent reward $R_n^{eva}$ is based on classifier evasion success with penalties for excessive perturbation magnitude and a contrastive diversity term given by the Contrastive Bank.

### Contrastive Bank

To introduce contrastive learning among the Red agents, we implement a *ContrastiveBank* that stores a running behavioral prototype for each agent. For agent $n$, the cumulative perturbation vector $Pc_n$ aggregates the sequence of packet modifications applied during the episode, while $P_n$ denotes the current perturbation. Similarity is quantified using cosine similarity. The contrastive reward is defined as

$$R_n^{cont} = \cos(Pc_n, P_n) - \frac{1}{N-1} \sum_{j \neq n} \cos(Pc_n, Pc_j).$$

The first term promotes temporal consistency, encouraging the agent to remain aligned with its own attack trajectory. The second term enforces separation, penalizing overlap with the behavioral prototypes of other agents.

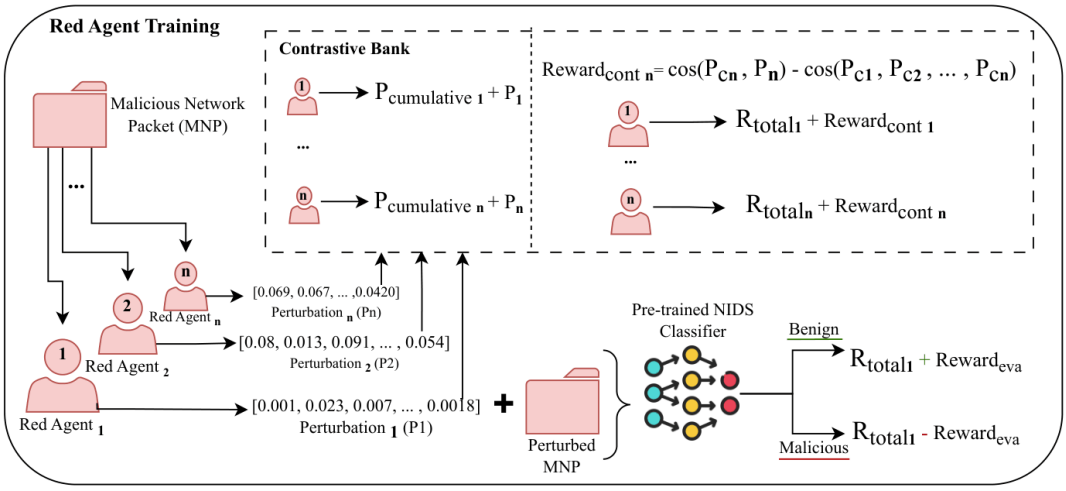


Figure 1. Contrastive learning procedure for red agent(s). $P_n$ is the n$^{th}$ agent's created perturbation. $P_{cumulative\ n}$ is the n$^{th}$ agent's cumulative perturbation. $R_{total\ n}$ is the n$^{th}$ agent's total reward for the learning episode. $Reward_{eva}$ and $Reward_{cont\ n}$ are the n$^{th}$ agent's rewards for the quality of the perturbation it creates. If the perturbation $P_n$ causes the Pre-trained NIDS Classifier to misclassify a Malicious Network Packet as Benign, then it recieves a *positive* $Reward_{eva}$ reward, otherwise it receives a *negative* reward. If the perturbation $P_n$ is similar to the agent's $P_{cumulative\ n}$ and dissimilar to the rest of the agents' $P_{cummulative\ n}$, then $Reward_{cont\ n}$ is greater.

The overall objective of the red agent becomes

$$R_n^{total} = R_n^{eva} + R_n^{cont},$$

This formulation has a direct human implication. When multiple agents converge to similar $Pc_n$, retraining on a small labeled set may neutralize them together but reveals little about other vulnerabilities. Greater separation among prototypes instead exposes distinct failure modes, enabling the defender to allocate scarce analyst effort more strategically.

## Current Experimental Setups

In the primary experimental setup, **configuration 1**, the attacker ensemble learns $n$ distinct perturbation strategies and deploys them sequentially over multiple **interaction rounds**. The number of rounds is determined by the number of initialized attackers. Each **round follows a three-phase cycle**: (1) the red agent learns and applies a perturbation strategy to generate adversarial malicious packets, (2) the blue agent determines an optimal retraining approach and updates the classifier accordingly, and (3) the updated classifier's accuracy is evaluated. For example, initializing 4 attackers results in 4 complete interaction rounds, with each round executing all three phases. This configuration addresses two key questions. *First*, we examine how the range of the red agent's action space affects its ability to generate effective perturbations. *Second*, we investigate the impact of scaling the attacker ensemble: does increasing the number of agents result in more effective policies that further degrade classifier accuracy, or do the policies exhibit diminishing returns?

In **configuration 2**, each attacker engages in three consecutive interaction rounds with the classifier before the next attacker is introduced. This setup tests whether repeated exposure to a single attack strategy causes the blue agent to over-specialize its retraining approach, making the classifier vulnerable when a subsequent attacker introduces a novel perturbation strategy.

## Results

We evaluated multiple $\epsilon$ values for the red agents' action space under **Configuration 1**, ranging from 0.001 to 0.25, across four interaction rounds with four red agents. Table 2 reports results for $\epsilon = 0.1$. As shown in the *Post* column, classifier accuracy drops substantially after each attack, indicating effective degradation of model performance. Although the defender retrains the classifier following each interaction (see *Retrain* column), subsequent agents continue to reduce accuracy. This suggests that the defender adapts primarily to previously observed perturbations and remains vulnerable to novel strategies introduced by contrastively trained agents.

Results for $\epsilon = 0.001$ and $0.01$ are provided in the Appendix (Tables 7 and 4). While accuracy declines across sequential attacks in these settings, degradation is less stable and generally weaker than with $\epsilon = 0.1$.

Table 1 presents results for $\epsilon = 0.25$ under the same configuration. In this regime, agents learn more distinct perturbation strategies and sequentially exploit the classifier, causing repeated performance collapse despite retraining. Overall, increasing the action-space magnitude amplifies adversarial impact while preserving diversity among attack strategies.

We next examined the effect of increasing the number of adversarial agents while maintaining $\epsilon = 0.25$. Tables 8 and 6 show results for configurations with two and four agents, respectively. Across both settings, larger action

spaces produce more stable and consistently effective perturbations, enabling agents to discover distinct and highly effective attack directions. Despite the larger ensemble, perturbation strategies do not collapse into a single dominant pattern. Figure 4 illustrates meaningful separation among most agents' embedding clouds, providing evidence that the contrastive reward mechanism maintains policy diversity even in higher-agent regimes. Together, these results indicate that expanding both the action space and the adversarial ensemble strengthens robustness evaluation by generating varied and persistent attack behaviors.

For **Configuration 2**, we allowed each of three agents to execute three consecutive interaction rounds before switching to the next agent. This setup tests whether prolonged exposure to a single perturbation strategy causes the defender to over-specialize during retraining, increasing vulnerability when a new attacker introduces a distinct strategy.Table 3 shows that the first agent reduces classifier accuracy similarly to **Configuration 1**. However, after switching agents, accuracy remains suppressed but becomes more unstable across rounds. This instability persists even when transitioning to the third agent. These findings suggest that extended exposure to a single adversarial strategy may encourage overfitting to specific perturbation patterns, reducing generalization to novel attack behaviors and introducing variability in classifier performance.

Across all configurations, a consistent pattern emerges: classifier accuracy begins high, drops sharply after red-agent attacks, partially recovers with additional data, and substantially improves after full retraining with drift adaptation (Figure 5). Compared to prior work reporting post-attack accuracy reductions to approximately 68% (Rivas et al. 2025), our contrastive multi-agent strategy degrades accuracy further to an average of 35%. This demonstrates that even when defenders adapt through retraining, novel perturbation strategies consistently degrade performance, revealing the limits of reactive adaptation. When multiple coordinated attackers introduce successive zero-day–like perturbations, the classifier remains persistently vulnerable despite periodic recovery. These results indicate that contrastively trained adversaries generate sustained, evolving attack strategies that challenge standard retraining-based defenses, underscoring the importance of evaluating defensive systems under diverse, sequential adversarial conditions rather than single-attack scenarios.

| Agent | Pre | Post | Merged | Retrain |
|---|---|---|---|---|
| 0 | $98.98 \pm 0.25\%$ | $88.96 \pm 0.46\%$ | $84.24 \pm 2.77\%$ | $98.48 \pm 0.26\%$ |
| 1 | $96.16 \pm 0.62\%$ | $34.82 \pm 5.95\%$ | $60.42 \pm 2.14\%$ | $92.18 \pm 1.55\%$ |
| 2 | $99.54 \pm 0.64\%$ | $7.38 \pm 2.43\%$ | $56.42 \pm 0.13\%$ | $88.03 \pm 8.29\%$ |
| 3 | $94.46 \pm 6.98\%$ | $4.30 \pm 0.65\%$ | $44.67 \pm 11.70\%$ | $92.38 \pm 4.57\%$ |

Table 1. Accuracy progression reported as mean $\pm$ std. under sequential red-agent attacks and defender retraining. *Pre:* Initial Classifier Accuracy, *Post:* Classifier accuracy after red agent attack, *Merged:* Classifier accuracy post red agent attack given new batch of samples, *Retrain:* Classifier accuracy after blue agent retraining. Perturbation action space was set to $\pm$ 0.25

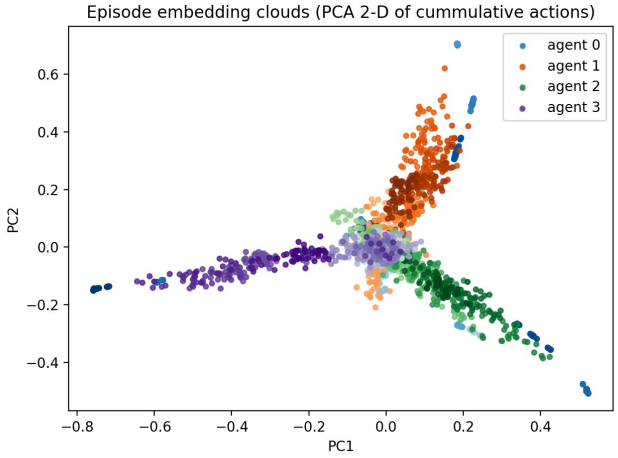


Figure 2. Diversity of red agent perturbation strategies during training seen in Table 1 projected into two dimensions using PCA. The visualization shows low similarity between agents, indicating that contrastive training encourages distinct perturbation styles rather than policy collapse.

## Conclusion

In this work, we investigated how coordinated, contrastively trained adversarial agents can systematically degrade the performance of adaptive network intrusion detection systems. Across multiple configurations, we observed that sequential attackers, each learning distinct perturbation strategies, consistently reduced classifier accuracy, even when the defender employed drift adaptation and periodic retraining. While retraining temporarily restored performance, the introduction of a novel adversarial strategy repeatedly exposed previously unseen vulnerabilities.

Our findings demonstrate that informed and coordinated attackers pose a substantial challenge to reactive defense mechanisms. Even when paired with human analysts who label samples for retraining, defenders remain vulnerable to diverse and evolving perturbation strategies. When agents converge to similar behaviors, limited retraining data may neutralize them collectively but reveal little about broader weaknesses. In contrast, greater separation among perturbation prototypes exposes multiple distinct failure modes, highlighting both the strategic value and the practical limits of human-guided retraining.

Overall, our results indicate that robustness cannot be meaningfully assessed through isolated or single-attack scenarios. Defensive systems must instead be evaluated under coordinated, multi-agent adversarial pressure that reflects realistic threat models, including sequences of zero-day–like perturbations, offering a stronger benchmark for testing adaptive defenses. Future work should explore closer integration between automated diversity detection and human-guided retraining to better anticipate coordinated adversarial behavior, and extend evaluation to real-world critical infrastructure such as LLM hosting platforms and other high-stakes communication channels.

# Appendix

## Table Term Definitions

| Term | Description |
|---|---|
| **Pre** | Initial classifier accuracy before any adversarial interaction. |
| **Post** | Accuracy immediately after the red agent attack. |
| **Merged** | Accuracy on the combined dataset (benign, malicious, and perturbed samples) following the attack. |
| **Retrain** | Accuracy after the blue agent performs defensive retraining. |

| Agent | Pre | Post | Merged | Retrain |
|---|---|---|---|---|
| 0 | 98.81% | **89.11%** | 80.74% | 96.21% |
| 1 | 88.24% | **34.64%** | 60.15% | 86.09% |
| 2 | 100.00% | **7.40%** | 56.52% | 98.70% |
| 3 | 73.08% | **3.84%** | 23.26% | 99.07% |

Table 2. Accuracy progression under sequential red-agent attacks and defender retraining. Perturbation action space was set to $\pm 0.1$.

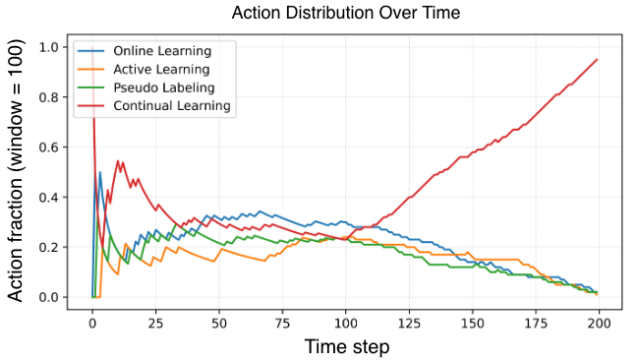


Figure 3. Blue agent action distribution while training.

| Agent | Run | Pre | Post | Merged | Retrain |
|---|---|---|---|---|---|
| 0 | 1 | 100.00% | **88.80%** | 80.58% | 96.90% |
| 0 | 2 | 100.00% | **31.50%** | 60.32% | 90.87% |
| 0 | 3 | 100.00% | **4.76%** | 54.95% | 80.18% |
| 1 | 1 | 95.15% | **3.84%** | 53.68% | 93.94% |
| 1 | 2 | 99.32% | **30.82%** | 60.30% | 93.26% |
| 1 | 3 | 85.29% | **1.96%** | 56.67% | 99.17% |
| 2 | 1 | 93.33% | **25.18%** | 55.28% | 86.99% |
| 2 | 2 | 100.00% | **0.00%** | 55.75% | 85.40% |
| 2 | 3 | 100.00% | **25.00%** | 61.79% | 85.85% |

Table 3. Accuracy progression under sequential red-agent attacks and defender retraining. Perturbation action space was set to $\pm$ 0.25

| Agent | Pre | Post | Merged | Retrain |
|---|---|---|---|---|
| 0 | 100.00% | **90.06%** | 81.41% | 93.51% |
| 1 | 99.32% | **98.63%** | 95.62% | 95.62% |
| 2 | 95.28% | **74.52%** | 87.89% | 89.69% |
| 3 | 70.59% | **7.84%** | 39.37% | 98.64% |

Table 4. Accuracy progression under sequential red-agent attacks and defender retraining. Perturbation action space was set to $\pm$ 0.001

| Agent | Run | Pre | Post | Merged | Retrain |
|---|---|---|---|---|---|
| 1 | 1 | 98.00% | 89.18% | 81.42% | 95.00% |
| 2 | 1 | 100.00% | 29.57% | 59.51% | 93.93% |
| 1 | 2 | 100.00% | 9.90% | 57.26% | 99.57% |
| 2 | 2 | 100.00% | 2.91% | 53.81% | 92.38% |
| 2 | 3 | 100.00% | 32.88% | 58.87% | 94.34% |

Table 5. Accuracy progression under sequential red-agent attacks and defender retraining. Perturbation action space was set to $\pm$ 0.25

| Agent | Pre | Post | Merged | Retrain |
|---|---|---|---|---|
| 0 | 100.00% | **89.01%** | 81.81% | 98.15% |
| 1 | 93.04% | **36.70%** | 56.23% | 96.60% |
| 2 | 100.00% | **7.40%** | 55.36% | 85.71% |
| 3 | 100.00% | **2.91%** | 53.46% | 88.94% |
| 4 | 96.71% | **38.81%** | 62.45% | 94.42% |
| 5 | 74.00% | **0.00%** | 46.47% | 92.37% |

Table 6. Accuracy progression under sequential red-agent attacks and defender retraining. Perturbation action space was set to $\pm$ 0.25

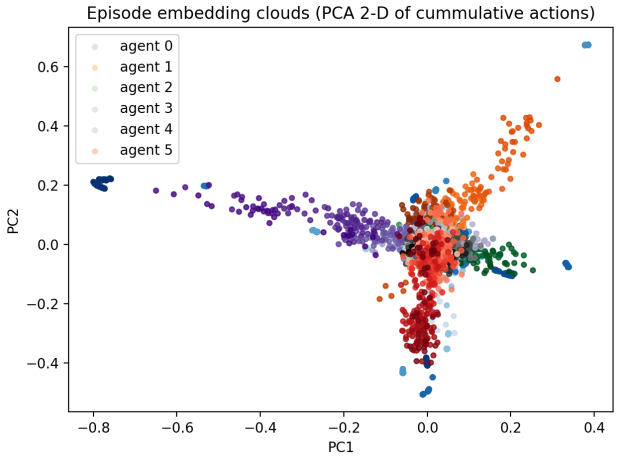


Figure 4. Diversity of red agent perturbation strategies during training seen in Table 6 projected into two dimensions using PCA.

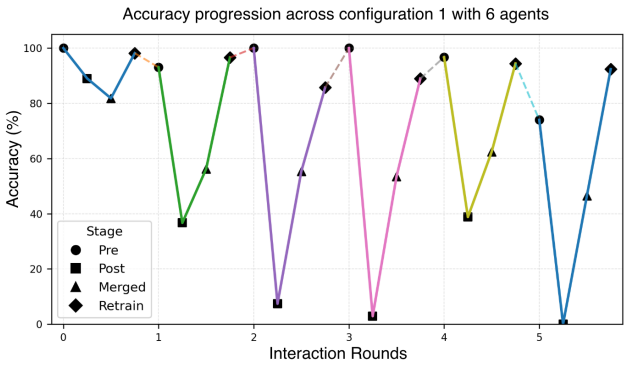


Figure 5. Accuracy progression over six interaction rounds with six distinct agents as seen in Table 6. Perturbation action space was set to $\pm$ 0.25

| Agent | Pre | Post | Merged | Retrain |
|---|---|---|---|---|
| 0 | 100.00% | **89.14%** | 80.66% | 98.36% |
| 1 | 93.75% | **30.55%** | 42.40% | 95.60% |
| 2 | 100.00% | **7.40%** | 57.74% | 99.16% |
| 3 | 84.31% | **55.88%** | 71.95% | 97.74% |

Table 7. Accuracy progression under sequential red-agent attacks and defender retraining. Perturbation action space was set to $\pm$ 0.01

| Agent | Pre | Post | Merged | Retrain |
|---|---|---|---|---|
| 0 | 98.76% | **88.88%** | 82.31% | 92.35% |
| 1 | 85.29% | **26.47%** | 34.57% | 92.59% |

Table 8. Accuracy progression under sequential red-agent attacks and defender retraining. Perturbation action space was set t0 $\pm$ 0.25

## Acknowledgments

This work was supported by the Health Resources and Services Administration (HRSA) under Grant No. 226-141-478A.

## References

Alam, M. T.; Piplai, A.; and Rastogi, N. 2025. ADAPT: A Pseudo-labeling Approach to Combat Concept Drift in Malware Detection. *arXiv preprint arXiv:2507.08597*.

Algomox. 2025. Drift Happens: Let ML Detect It Before It Becomes a Breach.

Amalapuram, S. K.; Tamma, B. R.; and Channappayya, S. S. 2024. Spider: A semi-supervised continual learning-based network intrusion detection system. In *IEEE INFOCOM 2024-IEEE Conference on Computer Communications*, 571–580. IEEE.

Beeson, A.; and Montana, G. 2024. Balancing policy constraint and ensemble size in uncertainty-based offline reinforcement learning. *Machine Learning*, 113(1): 443–488.

Dang, Q.-V. 2020. Active Learning for Intrusion Detection Systems. In *2020 RIVF International Conference on Computing and Communication Technologies (RIVF)*, 1–3.

Exabeam. 2025. The AI-Powered SOC: Capabilities, Benefits & Best Practices.

Hore, S.; Ghadermazi, J.; Paudel, D.; Shah, A.; Das, T.; and Bastian, N. 2025. Deep packgen: A deep reinforcement learning framework for adversarial network packet generation. *ACM Transactions on Privacy and Security*, 28(2): 1–33.

Li, J.; Zhang, H.; Liu, Y.; and Liu, Z. 2022. Semi-supervised machine learning framework for network intrusion detection. *Journal of Supercomputing*, 78(11).

Louati, F.; Ktata, F. B.; and Amous, I. 2024. Enhancing Intrusion Detection Systems with Reinforcement Learning: A Comprehensive Survey of RL-based Approaches and Techniques. *SN Computer Science*, 5(6): 665.

Pasikhani, A.; Gope, P.; Yang, Y.; Mehnaz, S.; and Sikdar, B. 2026. Baiting AI: Deceptive adversary against AI-protected industrial infrastructures. *IEEE Transactions on Dependable and Secure Computing*.

Rahman, M. S.; Coull, S.; Yu, Q.; and Wright, M. 2025. MADAR: Efficient continual learning for malware analysis with diversity-aware replay. *arXiv preprint arXiv:2502.05760*.

Rivas, E.; Saika, S.; Bakht, A.; Piplai, A.; Bastian, N. D.; and Shah, A. 2025. Adapting Under Fire: Multi-Agent Reinforcement Learning for Adversarial Drift in Network Security. *arXiv preprint arXiv:2506.06565*.

Stiawan, D.; Idris, M. Y. B.; Bamhdi, A. M.; Budiarto, R.; et al. 2020. CICIDS-2017 dataset feature analysis with information gain for anomaly detection. *IEEE Access*, 8: 132911–132921.

Umucu, E.; Solis, G.; Garza, L.; Rivas, E.; Lee, B.; Kotal, A.; and Piplai, A. 2025. Empathy by Design: Aligning Large Language Models for Healthcare Dialogue. *arXiv preprint arXiv:2512.06097*.

Wu, Y.; Hu, Y.; Wang, J.; Feng, M.; Dong, A.; and Yang, Y. 2024. An active learning framework using deep Q-network for zero-day attack detection. *Computers & Security*, 139: 103713.

Yang, L.; Guo, W.; Hao, Q.; Ciptadi, A.; Ahmadzadeh, A.; Xing, X.; and Wang, G. 2021. {CADE}: Detecting and explaining concept drift samples for security applications. In *30th USENIX Security Symposium (USENIX Security 21)*, 2327–2344.

Zhang, X.; Zhao, R.; Jiang, Z.; Chen, H.; Ding, Y.; Ngai, E. C.; and Yang, S.-H. 2025. Continual learning with strategic selection and forgetting for network intrusion detection. In *IEEE INFOCOM 2025-IEEE Conference on Computer Communications*, 1–10. IEEE.

Zhao, X.; He, M.; and Wang, X. 2025. Semi-Supervised Learning With Interpolation and Pseudo-Labeling for Few-Label Intrusion Detection. *IEEE Transactions on Network and Service Management*.