Who Watches the Watchers? Semantically-Constrained Reinforcement Learning for Red-Teaming Provenance Intrusion Detectors

Brayden Killeen[1] **Derui Wang**[2] **Nasrin Sohrabi**[3]
Zahir Tari[1] **Qin Wang**[2] **Minhui Xue**[2]

[1]RMIT University [2]CSIRO's Data61 [3]Deakin University

Abstract

Reconstruction-based provenance intrusion detection systems (PIDS) detect malicious activity by classifying nodes in system-call provenance graphs from per-edge reconstruction errors produced by benign-trained encoder-decoders. Because these detectors run on the endpoints they monitor, adversarial robustness is a critical deployment concern, yet systematic evaluation of it remains rare. We introduce ProvRT, a reinforcement-learning framework for automated red-teaming of these detectors, and use it to characterise the robustness of four leading PIDS spanning the four encoder families used in the field. ProvRT casts red-teaming as a semantically constrained graph-editing MDP, with a factored autoregressive policy masked by transition relations mined from benign data and a GRU belief state, to model the multi-step consequences of message passing on GNN-based detectors. ProvRT shows that reconstruction-based PIDS allow meaningful evasion across three of four encoder families (GNN, linear, and, to a lesser extent, VAE) in white-, grey-, and black-box settings, relative to non-exhaustive baselines, and at inference time uses 5–99$\times$ fewer detector queries than search-based attackers; direct-reconstruction encoders resist insertion attacks structurally, suggesting an architectural direction for robust detection. No fixed aggregation or threshold-adaptive policy maintains both a deployable false-positive rate and meaningful evasion resistance, but adversarial retraining of the scoring head on ProvRT-generated chains reduces evasion from 0.97 to 0.30 on the GNN detector and from 0.91 to 0.0 on the linear detector at 5% FPR, and an adaptive attacker retrained against the hardened detectors recovers at most 0.24. ProvRT trains on CPU within hours and produces attack chains that replay as Linux syscalls. We open-source our code.[1]

1 Introduction

As frontier AI systems become autonomous actors in software environments, the security problem shifts from validating model outputs to monitoring the system actions those outputs induce. Provenance-based intrusion detection systems (PIDS) classify nodes in system-call provenance graphs as benign or part of an attack [14, 21, 3, 15, 11, 2], providing a natural defence layer for monitoring autonomous and human-driven system activity alike. These graphs capture the causal traces of OS-level objects (processes, files, sockets, memory) produced by the kernel's audit subsystem. Modern PIDS use a learned encoder to embed each node conditioned on its graph neighbourhood and to flag nodes whose representations deviate from the benign training distribution. Recent detectors report state-of-the-art results on standard provenance benchmarks [2], positioning PIDS as a promising basis for detecting post-compromise attacks on endpoints.

[1] `https://github.com/ProvRT/ProvRT`

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Given this performance, there is considerable interest in deploying PIDS in security-critical pipelines, where their outputs inform alerting, investigation, and incident response [27]. However, deployment raises a question that benign-distribution detection does not answer: Can a motivated attacker who observes the detector's outputs and makes bounded modifications to the monitored system reliably cause targeted false negatives while still executing their primary attack objective?

Automated red-teaming, training an adversary to systematically probe detector failure modes, offers one route to answering this question, and is a leading empirical method for characterising adversarial robustness [19, 7]. Tooling has primarily focused on vision and language models, leaving red-teaming of *learned detectors on structured graph inputs* such as PIDS underdeveloped relative to detector quality. While PIDS provide a promising defence layer for endpoint security and emerging agentic AI deployments, their own adversarial robustness remains insufficiently characterised. Prior PIDS attacks are either non-adaptive (mimicry [10, 28]) or per-detector heuristics (gadget search [17]); neither learns a reusable policy, nor have they been evaluated against adaptive defenders.

Challenges. A learned, reusable red-teaming policy must overcome three technical obstacles. *(i) Combinatorial action space.* For edge insertion, the attacker selects a source, destination, and edge type at each step. Naive enumeration is intractable, yet arbitrary restrictions risk excluding insertions that actually evade. *(ii) Multi-step consequences.* On GNN-based detectors, an inserted edge shifts every node embedding in its multi-hop neighbourhood via message passing, which can unpredictably undo earlier evasion progress. *(iii) Semantic validity.* Every inserted edge must correspond to an executable system call under the audit collection tool's schema and originate from an attacker-controlled node; constraints that feature-space attacks [16, 9] ignore. These challenges distinguish PIDS red-teaming from prior graph-adversarial work, which targets static node-classification with single-perturbation budgets: on a provenance graph, the attacker's edits are sequenced system calls that the detector consumes online and propagates through the encoder's neighbourhood representation, making the attack a trajectory of decisions over an evolving state rather than a static perturbation.

We introduce ProvRT, a reinforcement-learning framework for automated red-teaming of learned reconstruction-based intrusion detectors. ProvRT formulates red-teaming as a constrained graph-editing MDP: an agent incrementally inserts semantically valid edges until per-node labels flip on the adversarial set or an insertion budget is exhausted. Three design choices make the action space tractable. First, a *factored autoregressive policy* decomposes each action into source, destination, and edge-type decisions, reducing per-step sampling cost from $O(|V|^2 \cdot |\mathcal{E}|)$ to $O(|V_{\text{adv}}| + |V_{\text{dst}}(u)| + |\mathcal{E}|)$ (Appendix A). Second, an *action mask* intersects a transition relation mined from benign data with the detector's supported edge types. Third, a *GRU recurrent trunk* maintains belief state across the episode for reasoning about message-passing consequences. ProvRT's learned policy amortises exploration across training, unlike heuristic attackers that recompute candidate gadgets per graph.

Contributions. (1) *ProvRT evades three of four reconstruction-based PIDS encoder families.* ProvRT causes targeted false negatives in white-, grey-, and black-box settings on GNN-, linear-, and, to a lesser extent, VAE-based encoders; FLASH's direct-reconstruction encoder structurally resists insertion attacks. (2) *Fixed defences fail at deployable false-positive rates; adversarial retraining does not.* Across aggregation and threshold-adaptive policies, no operating point in the reconstruction-based approach maintains both a deployable false-positive rate and meaningful evasion resistance. Retraining a discriminative scoring head on ProvRT-generated chains reduces ESR from 0.97 to 0.30 (GNN) and from 0.91 to 0.0 (linear) at 5% FPR, and an adaptive attacker retrained against the hardened detector recovers 0.0 and 0.24 respectively; a four-feature distribution-shift meta-detector further flags ProvRT chains at AUC 0.999. (3) *ProvRT is fast and query-efficient.* A learned policy converges in under 200k environment steps per (PIDS, dataset) pair, a one-time training cost amortised across all subsequent attacks; at inference it uses 5–99$\times$ fewer detector queries than search-based attackers at near-matched ESR, and produces attack chains that replay as real Linux syscalls.

2 Problem Setting

A provenance graph $G = (V, E, X_V, X_E)$ is a labelled, heterogeneous directed graph where V is a set of OS-object nodes and $E \subseteq V \times V$ is a set of directed edges representing causal relations induced by system calls, X_V and X_E are node and edge features. Nodes carry a type from a set $\mathcal{N}$; edges carry a type from a set $\mathcal{E}$ of system-call categories. A PIDS is a function $D : \mathcal{G} \to \mathbb{R}^{|\mathrm{V}|}$ that

assigns a per-node anomaly score $p_v = D(G)_v$. A deployment applies a detection threshold $\tau \in \mathbb{R}$, producing a binary label $y_v = \mathbf{1}\{p_v \geq \tau\}$. A *detection* of an attack $V_{\text{adv}} \subseteq V$ occurs when $y_v = 1$ for at least one $v \in V_{\text{adv}}$; *full evasion* requires $y_v = 0$ for every $v \in V_{\text{adv}}$. We focus on node-level detection with a fixed adversarial set; graph-level and edge-level classifiers are out of scope.

Given an initial attack graph G_0 with adversarial nodes V_{adv}, a PIDS D, a threshold τ, and an insertion budget B, we cast red-teaming as a finite-horizon deterministic MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{A}(\cdot), T, R, H \rangle$:

- **State.** $s_t = (G_t, p_t, y_t, b_t, h_t)$, comprising the current graph, per-node scores, labels, remaining budget $b_t = B - t$, and step count $h_t = t$.
- **Action.** $a_t = (u, v, e) \in V_{\text{adv}} \times V \times \mathcal{E}$ inserts an edge with source u, destination v, and type e.
- **Feasible set.** $\mathcal{A}(s_t)$ is the intersection of (i) the semantic transition relation $\Phi \subseteq \mathcal{N} \times \mathcal{N} \times \mathcal{E}$ encoding schema validity, (ii) $u \in V_{\text{adv}}$, (iii) u active on outgoing edges, and (iv) $b_t > 0$.
- **Transition.** $T(s_t, a_t) = s_{t+1}$, where $G_{t+1} = G_t \cup \{(u, v, e)\}$ with timestamp assigned by t_E, $p_{t+1} = D(G_{t+1})$, and y_{t+1} is derived by thresholding at τ.
- **Reward.** $R(s_t, a_t, s_{t+1})$ is shaped toward evasion success rate (full description in Section 3).
- **Horizon.** $H = B$; the episode terminates when the budget is exhausted or V_{adv} is fully evaded.

Stochasticity enters only via the initial-state distribution ρ_0 over attack-window graphs and the policy.

Three properties distinguish this setting from feature-space adversarial generation: the attack is *online* (each insertion modifies G before the next is chosen), *budget-constrained* (at most B insertions; efficiency matters for rate-limited detectors), and *semantically constrained* (every action must satisfy Φ, ruling out perturbations without corresponding system calls).

2.1 Threat Model

Attacker position. The attacker has achieved code execution on an endpoint monitored by PIDS D. They control a set of adversarial nodes $V_{\text{adv}} \subseteq V$ corresponding to their payload process(es) and the OS objects those processes have created or touched. V_{adv} is fixed at the start of the attack. The attacker does not control other processes on the host, the PIDS weights, the threshold τ, the audit daemon, or the provenance-collection pipeline. Log tampering (editing or deleting provenance records) and audit subversion are orthogonal threats outside our scope. While our MDP is deterministic, a deployed attacker can dynamically adapt to host-level noise and concurrent processes, which, in practice, aid evasion by further diluting anomaly scores.

Attacker capabilities. The attacker may perform edge insertion subject to the constraints in $\mathcal{A}(s_t)$ above: semantic validity via Φ, source restriction, outgoing-edge activity, and budget. The attacker may not delete, modify, or reorder existing edges, perturb features on existing nodes, or change pre-existing timestamps. Insertion is a deliberately conservative choice under this threat model, as it is a minimal capability of an attacker with code execution and provides a lower bound on evasion; additional capabilities are left to future work. During training and evaluation, inserted edges are synthetic edits to recorded attack-window graphs; each corresponds to a real system call issued from an attacker-controlled process, which we confirm by replaying a chain on an audited host (Appendix F). These primitives correspond to common post-compromise techniques (MITRE ATT&CK T1059, T1036, T1071, T1005 [24]; detailed in Appendix B. Constraining insertions to semantically valid, high-frequency operations ensures these actions blend into baseline activity without triggering orthogonal runtime security controls.

Observation tiers. The observation function O maps the full state s_t to the agent's view, reflecting the attacker's knowledge tier. A **white-box** observation $o_t = (Z_t, p_t, y_t)$ provides per-node embeddings, scores, and labels. A **grey-box** observation $o_t = (p_t, y_t)$ provides scores and labels only. A **black-box** observation $o_t = (y_t)$ provides only the binary detection labels.

White-box access is plausible when D is open-source or when evaluating an upper bound on achievable evasion. Grey-box access corresponds to oracle access to a scoring interface, such as a compromised SIEM dashboard or a submission-based detection service. Black-box access is the most realistic operational setting, in which the attacker observes only whether they were detected. All three tiers share the same MDP $\mathcal{M}$, state space, and action space; they differ only in O.

Defender Model. The defender operates D with a threshold τ_k updated between episodes via an operator $U : \mathbb{R} \times \mathcal{S}^* \to \mathbb{R}$, $\tau_{k+1} = U(\tau_k, S_k)$, where S_k is the set of graphs scored during episode k.

We evaluate four defender classes: threshold adaptation (below), robust aggregation and adversarial retraining of the scoring head (Section 4.4), and a distribution-shift meta-detector over the attacker's structural footprint (Section 4.5). For threshold adaptation we evaluate three methods: (1) **Static** ($\tau_k = \tau_0$), (2) **Mean** (τ_{k+1} set to the running mean of recent benign scores), and (3) **Percentile** (τ_{k+1} set to the q-th percentile of recent scores). This between-episode update is weaker than a Stackelberg equilibrium but reflects that SOC teams recalibrate periodically, not in real time.

3 Method

ProvRT solves the MDP of Section 2 with a PPO [23] agent over the combinatorial, semantically constrained action space of provenance graphs: at each step, the agent selects a source from V_{adv}, a destination from V, and an edge type from $\mathcal{E}$. A naive flat policy requires $O(|V|^2 \cdot |\mathcal{E}|)$ logits per step, on the order of 10^{10} for a 10^4-node graph with ~ 30 edge types.

Key insights. Three observations, each later confirmed empirically (Section 4), shape our design. *(O1) Message passing amplifies perturbations on GNN detectors*: a single inserted edge shifts every node embedding in its multi-hop neighbourhood, so one well-placed insertion can flip multiple adversarial nodes at once but can also *unflip* previously-evading ones. Handling this requires a temporal state across the episode. *(O2) Linear encoders resist via a score floor*: when the encoder is independent of graph structure, evasion proceeds only via dilution at the decoder's aggregation layer, and the benign edge-score minimum caps how far each insertion can push the adversarial mean. Dilution requires many insertions per flipped node and enables aggregation-level defences. *(O3) The feasible action set is sparse*: most $(\text{type}(u), \text{type}(v), e)$ triples are invalid under the graph's transition rules, and within valid triples. A small fraction is consistent with the attacker's controllable sources. A factored, masked policy concentrates probability mass on admissible actions without enumerating the full space.

Algorithm 1 ProvRT inference-time attack chain generation.

Require: $G_0, V_{\text{adv}}, D, \tau, B, \pi_\theta = (\pi_{\text{src}}, \pi_{\text{dst}}, \pi_{\text{edge}}), \Phi$.
Ensure: Chain $C = (a_1, \ldots, a_T), T \le B$.
$G \leftarrow G_0$; $h_0 \leftarrow \mathbf{0}$; $C \leftarrow [\,]$
$Z \leftarrow \textsc{EmbedNodes}(G_0, D)$
for $t = 1, \ldots, B$ **do**
  $p_t \leftarrow D(G)$; $y_t \leftarrow \mathbf{1}\{p_t \ge \tau\}$
  if $y_v = 0 \; \forall v \in V_{\text{adv}}$ **then**
    return C
  end if
  $o_t \leftarrow (Z, p_t, y_t)$
  $z_t \leftarrow \textsc{Encode}(o_t)$
  $h_t \leftarrow \text{GRU}(h_{t-1}, z_t)$
  $c_t \leftarrow W h_t + z_t$
  $u \sim \pi_{\text{src}}(\cdot \mid c_t, V_{\text{adv}})$
  $m_{\text{dst}} \leftarrow \{v : \exists\, e, (\text{type}(u), \text{type}(v), e) \in \Phi\}$
  $v \sim \pi_{\text{dst}}(\cdot \mid c_t, u, m_{\text{dst}})$
  $m_{\text{edge}} \leftarrow \{e : (\text{type}(u), \text{type}(v), e) \in \Phi\}$
  $e \sim \pi_{\text{edge}}(\cdot \mid c_t, u, v, m_{\text{edge}})$
  $G \leftarrow G \cup \{(u, v, e)\}$
  $C \leftarrow C \,\|\, (u, v, e)$
end for
return C

ProvRT addresses (O1) and (O3) with three design choices below: a factored autoregressive policy (O3), transition-rule masking (O3), and a GRU recurrent trunk (O1). (O2) is an analytical finding we return to in Section 4. Algorithm 1 summarises the inference-time procedure.

Factored Autoregressive Policy. The policy factorises each action $a_t = (u, v, e)$ into three sequential decisions:

$$\pi_\theta(a_t \mid o_t) \;=\; \pi_{\text{src}}(u \mid o_t) \cdot \pi_{\text{dst}}(v \mid u, o_t) \cdot \pi_{\text{edge}}(e \mid u, v, o_t). \tag{1}$$

Each factor is a separate neural-network head operating on a shared observation encoding. The heads are evaluated left to right: the source head samples u, which conditions the destination head's candidate set; the destination head samples v, which conditions the edge-type head's valid types. To rigorously enforce the semantic constraints at sampling time, each head's logits are masked to its conditional feasible subset using an indicator function $\mathbb{I}[\cdot]$:

$$\pi_{\text{src}}(u \mid o_t) \propto \exp(\phi_{\text{src}}(u)^\top c_t) \cdot \mathbb{I}[u \in V_{\text{adv}}], \tag{2}$$

$$\pi_{\text{dst}}(v \mid u, o_t) \propto \exp(\phi_{\text{dst}}(v)^\top c_t) \cdot \mathbb{I}[v \in V_{\text{dst}}(u)], \tag{3}$$

$$\pi_{\text{edge}}(e \mid u, v, o_t) \propto \exp(\phi_{\text{edge}}(e)^\top c_t) \cdot \mathbb{I}[e \in \mathcal{E}(u, v)], \tag{4}$$

where c_t is the recurrent context vector (defined in Section 3), $\phi_{(\cdot)}$ are linear projections, $V_{\text{dst}}(u)$ is the set of type-valid destinations for u, and $\mathcal{E}(u, v)$ is the set of valid edge types between u and v under the transition relation Φ.

Proposition 1 (Tractability and coverage of the factored masked policy)**.** *Let* $\mathcal{A}(s) = \{(u, v, e) \in V_{\text{adv}} \times V \times \mathcal{E} : (\text{type}(u), \text{type}(v), e) \in \Phi\}$ *be the feasible set at state s, and let each head of* π_θ *be masked to its respective feasible conditional. Then* ***(Tractability)*** *sampling from* π_θ *evaluates at most* $|V_{\text{adv}}| + |V_{\text{dst}}(u)| + |\mathcal{E}|$ *logits, where* $V_{\text{dst}}(u) = \{v \in V : \exists\, e\ (\text{type}(u), \text{type}(v), e) \in \Phi\}$*; and* ***(Coverage)*** $\{a : \pi_\theta(a \mid o) > 0\} = \mathcal{A}(s)$ *for every s.*

The complexity reduction from $O(|V|^2 \cdot |\mathcal{E}|)$ to $O(|V_{\text{adv}}| + |V_{\text{dst}}(u)| + |\mathcal{E}|)$ is more than six orders of magnitude on graphs with $|V| \sim 10^4$ and $|V_{\text{adv}}| \sim 10$. The coverage property ensures the factorisation neither excludes feasible actions nor admits infeasible ones (proof in Appendix A).

In implementation, each head is a small MLP on a shared encoder output. The source head gathers embeddings for the $|V_{\text{adv}}|$ adversarial nodes and projects through a linear layer; the destination head scores only the type-valid indices (cached per episode so the mask is reused across steps); and the edge-type mask is a pre-computed lookup table keyed by the source-destination type pair.

Transition-Rule Masking. Action masking enforces $\mathcal{A}(s_t)$ at sampling time, preventing the policy from wasting probability mass on invalid actions. Φ is constructed from two sources. *Hardcoded schema rules* encode which node-type-to-node-type transitions are admissible for each edge type under the audit schema. *Data-driven transition statistics* count each $(\text{type}(u), \text{type}(v), e)$ triple observed in benign training graphs and retain those with count $\geq C_{\min} = 100$, which keeps high-frequency transitions such as (PROCESS, FILE, READ) and drops the long tail of rare high-entropy transitions. The union of the two is then intersected with the target detector's supported edge-type set.

GRU Recurrent Belief State. On GNN-based detectors, an inserted edge propagates across the multi-hop neighbourhood via message passing, potentially re-raising the score of a node driven below τ earlier in the episode. A memoryless policy cannot detect such unflips because the current observation does not reveal which nodes were previously evading; this makes red-teaming a POMDP [22], with recurrent policies the standard solution [12, 13]. ProvRT uses a GRU [4] recurrent trunk that maintains a hidden state h_t across the episode. This hidden state is combined with the encoded observation o_t^{enc} via a residual projection to form the context vector c_t fed to the policy and value heads:

$$h_t = \text{GRU}(h_{t-1}, o_t^{\text{enc}}), \quad c_t = W h_t + o_t^{\text{enc}} \tag{5}$$

We train with BPTT on a chunk length of 64 steps to bound memory during PPO updates.

Tier-Specific Observation Encoders. ProvRT supports the three observation tiers of Section 2.1 through tier-specific encoders that lift o_t to a common representation $o_t^{\text{enc}} \in \mathbb{R}^{|V| \times d}$. *White-box* ($o_t = (Z_t, p_t, y_t)$): the encoder operates directly on the per-node embeddings Z_t produced by D. *Grey-box* ($o_t = (p_t, y_t)$): an MLP ($1 \to 128 \to d$, ReLU) lifts per-node scores into d-dimensional pseudo-embeddings. *Black-box* ($o_t = y_t$): an embedding layer maps binary labels to d-dimensional vectors. Downstream of the encoder, the factored policy, masking, and GRU trunk are identical across tiers; the tier choice affects only the input-encoder stack.

Reward. The reward aligns with ESR maximisation while providing a per-step learning signal:

$$r_t \;=\; \eta_{\text{esr}} \cdot \Delta\text{FNR}_t \;+\; \eta_{\text{node}} \cdot \sum_{v \in V_{\text{adv}}^{>\tau}} w_v \cdot \Delta p_v \;-\; c_{\text{inv}} \cdot \mathbf{1}_{\text{invalid}}. \tag{6}$$

Here $\Delta\text{FNR}_t = \text{FNR}(G_t) - \text{FNR}(G_{t-1})$ is the per-step evasion progress; the inner sum runs over adversarial nodes still above threshold, with $\Delta p_v = p_v^{(t-1)} - p_v^{(t)}$ weighted by proximity $w_v = 1/(1 + \max(0, p_v - \tau))$ so that score reductions near the threshold dominate; and c_{inv} penalises invalid or null (SKIP) emissions to discourage degenerate policies. At episode termination, a one-time bonus η_{full} is awarded on full evasion and a small penalty c_{budget} on budget exhaustion. A reward sensitivity sweep over each coefficient and its disabled variant is reported in Appendix G.

4 Experiments

4.1 Setup

Datasets and detectors. We evaluate on three publicly available DARPA Transparent Computing datasets containing 6 distinct attacks, against four PIDS [6]. Each PIDS is from a representative per encoder family used in the field: GNN (Orthrus [14]), linear (VELOX [2]), VAE (NodLink [15]),

Table 1: Dataset-level mean ESR (Equation (7)) per (detector, dataset) configuration. ProvRT (PPO, our method) is compared against three baselines: HillClimb (HC), Mimicry (Mim) [10], and SemanticRandom (SemRand). Higher is better. Negative ESR indicates the attack increased detection relative to the initial graph.

Detector	Dataset	ProvRT	HC	Mim	SemRand
Orthrus (GNN)	E3-CADETS	0.929	1.000	−0.664	0.087
	E3-THEIA	1.000	1.000	−4.367	−0.380
	E3-CLEARSCOPE	1.000	1.000	1.000	−3.040
VELOX (Linear)	E3-CADETS	0.688	1.000	0.070	0.017
	E3-THEIA	1.000	1.000	0.800	0.055
	E3-CLEARSCOPE	0.986	1.000	1.000	−0.300
FLASH (Recon.)	E3-CADETS	-0.003	0.065	0.002	−0.069
	E3-THEIA	-0.028	0.038	-0.039	−0.087
	E3-CLEARSCOPE	-0.009	0.000	-0.003	−0.032
NodLink (VAE)	E3-CADETS	0.359	1.000	0.129	0.087
	E3-THEIA	0.105	1.000	0.259	0.157
	E3-CLEARSCOPE	0.325	1.000	-0.544	0.290

and direct reconstruction (FLASH [21]). All detectors are trained with PIDSMaker's tuned hyperparameters and achieve adjusted detection precision (ADP) consistent with the numbers reported in PIDSMaker [2], that is ADP > 0.2.

Baselines. We compare against three attackers that share the same budget, threshold, and transition-rule enforcement: *HillClimb* exhaustively scores every valid insertion at each step and commits the best one; *SemanticRandom* samples a uniformly random valid action per step; and *Mimicry* [10] cycles through 14 predefined benign file-I/O patterns. Each (PIDS, dataset, attacker) configuration is run with 5 independent seeds for each non-deterministic attacker. We report mean and standard deviation for ESR (Equation (7)) and PIDS inferences per attack. The budget is $B = 1{,}000$; detection thresholds are calibrated to 5% FPR per detector.

Attack Metric. The red-teaming objective is the *evasion success rate* (ESR):

$$\mathrm{ESR}(G_T) \;=\; \frac{\mathrm{FNR}(G_T) - \mathrm{FNR}(G_0)}{1 - \mathrm{FNR}(G_0)}, \tag{7}$$

where $\mathrm{FNR}(G) = |\{v \in V_{\mathrm{adv}} : D(G)_v < \tau\}|/|V_{\mathrm{adv}}|$ is the adversarial-set false-negative rate. ESR of 1 denotes full evasion, 0 indicates no progress, and negative values mean the attack aided detection.

4.2 ProvRT evades three of four encoder families across observation tiers

ProvRT attains near-full evasion on GNN-based Orthrus and linear-encoder VELOX, and positive but partial evasion on VAE-based NodLink. Table 1 reports dataset-level mean ESR for ProvRT, HillClimb, Mimicry, and SemanticRandom across all four detectors (per-attack results in Appendix C). ProvRT averages 0.98 ESR on Orthrus, 0.89 on VELOX, and 0.26 on NodLink across the three E3 datasets, while no attacker we test achieves meaningful evasion on FLASH (HillClimb reaches at most 0.065 ESR on E3-CADETS). On NodLink, ProvRT reaches 0.359 / 0.105 / 0.325 on E3-CADETS / THEIA / CLEARSCOPE, above both no-search baselines on E3-CADETS and E3-CLEARSCOPE; E3-THEIA is the weakest result in the campaign and falls below Mimicry (0.259) and SemanticRandom (0.157). HillClimb reaches full evasion on NodLink only at a substantially higher per-attack query cost (Section 4.3). The FLASH result is informative: direct-reconstruction encoders resist insertion attacks structurally; we identify the underlying mechanism in Section 4.6.

Non-adaptive attackers backfire on GNN-based detectors. Mimicry achieves -0.664 / -4.367 ESR on Orthrus E3-CADETS / E3-THEIA, and SemanticRandom -3.040 on E3-CLEARSCOPE. The same patterns that succeed on linear VELOX (0.070, 0.800) actively raise detection scores against learned graph encoders.

Knowledge restriction does not help the defender on Orthrus and only partially helps on VELOX (Figure 1, E3-CADETS). Orthrus's ESR is flat across tiers (0.97 white-box, 0.98 grey-box, 1.00 black-box): label-only feedback is sufficient for the attacks that succeed against this detector. VELOX

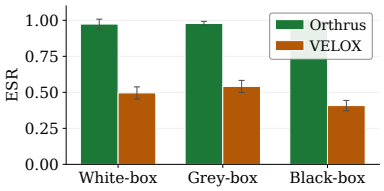

Figure 1: ESR on E3-CADETS by attacker observation tier across 5 seeds. Orthrus (green) is observation-invariant; VELOX (orange) holds from white- to grey-box and degrades under black-box. Error bars are ± one standard deviation.

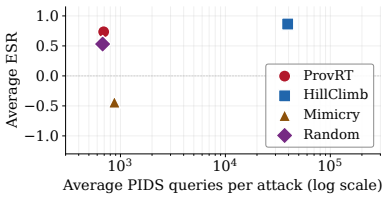

Figure 2: ESR vs. query cost on E3-CADETS, averaged over Orthrus and VELOX. ProvRT and HillClimb are the only non-dominated attackers.

Table 2: Robust aggregation defence on saved attack chains (5 seeds per detector).

PIDS	Aggregator	ESR
Orthrus	Mean (baseline)	1.000 ± 0.000
	Median	1.000 ± 0.000
	Trim (5/10/20%)	1.000 ± 0.000
VELOX	Mean (baseline)	0.547 ± 0.044
	Median	0.304 ± 0.089
	Trimmed mean (5%)	0.347 ± 0.082
	Trimmed mean (10%)	0.293 ± 0.059
	Trimmed mean (20%)	0.229 ± 0.056

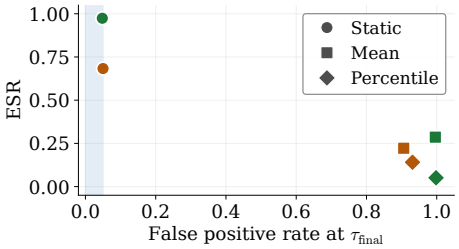

Figure 3: ESR vs FPR across defender policies on E3-CADETS. Markers: (defender, PIDS) pairs; green: Orthrus, orange: VELOX. The shaded area is the deployable FPR $\leq 5\%$.

holds steady from white-box (0.49) to grey-box (0.54), then drops 14 points to 0.40 under black-box, consistent with dilution-style attacks needing the score signal to choose insertions that push the adversarial mean below threshold. Both detectors remain evadable under the harshest restriction.

4.3 ProvRT is query-efficient relative to search-based attackers

ProvRT's learned policy uses orders of magnitude fewer detector queries than HillClimb at near-matched ESR. On Orthrus E3-CADETS, ProvRT reaches ESR 0.93 in 629 queries against HillClimb's 1.000 in 3,333 (5.3× reduction); on VELOX E3-CADETS, ESR 0.69 in 755 queries vs. HillClimb's 1.000 in 75,100 (99×). These are inference-time counts. Training is a separate, one-time cost of one detector query per environment step (10^6 steps per run, Table 5; convergence in under 200k steps, Appendix C), incurred once per (detector, dataset) pair and amortised across every subsequent attack, whereas HillClimb recomputes its per-step enumeration for every graph it attacks. The 5–99× figure therefore compares the per-attack cost of a pre-trained policy against search-based approaches that must be repeated per graph. Figure 2 summarises the efficiency trade-off: ProvRT and HillClimb are the two non-dominated points, with ProvRT trading marginal ESR for orders-of-magnitude lower query cost. Mimicry and SemanticRandom run in the same query regime as ProvRT but are dominated on ESR; Mimicry's negative aggregate ESR reflects the GNN backfire effect described in Section 4.2. At inference ProvRT produces a chain in under five seconds per graph on CPU.

4.4 Fixed defences fail at deployable FPR; adversarial retraining succeeds

Robust aggregation defends VELOX but leaves Orthrus exposed. Table 2 replays each attack chain with median and trimmed-mean edge-to-node aggregation in place of the default mean. On VELOX E3-CADETS, median aggregation reduces ESR from 0.547 ± 0.044 to 0.304 ± 0.089 (a 44% reduction); trimmed mean's effect scales with trim rate, from 0.347 ± 0.082 at 5% trim to 0.229 ± 0.056 at 20%. On Orthrus, every attack chain from every seed retains ESR $= 1.000 \pm 0.000$ regardless of aggregator. The asymmetry reflects an architectural fact: VELOX aggregates per-edge anomaly scores at the decoder, so swapping mean for median changes its output; Orthrus's encoder produces node embeddings whose dependence on each inserted edge is upstream of any aggregation step. Even on VELOX, the residual ESR of 0.229–0.347 across aggregators still represents meaningful

Table 3: Adversarial retraining of the scoring head (E3-CADETS, white-box, mean over 5 seeds, 5% FPR). *Replay* re-scores the saved ProvRT chains with the hardened head; *Adaptive* retrains ProvRT from scratch (10^6 steps) against the hardened detector.

Detector	Standard scoring	Hardened head (replay)	Adaptive ProvRT
Orthrus (GNN)	0.97	0.30	0.00
VELOX (Linear)	0.91	0.00	0.24

evasion, and the defence does not transfer to GNN-based detectors: no aggregation policy we evaluate suppresses ProvRT across both encoder families.

Threshold-adaptive defenders reduce ESR only by collapsing τ to operationally untenable FPR. Figure 3 reports ESR vs. FPR at each defender's final τ for the three policies of Section 2.1. Both adaptive policies drive τ into regimes that flag nearly every benign node: Mean reaches 0.28/0.22 ESR (Orthrus/VELOX) at $\sim$95% FPR and Percentile 0.05/0.14 at $\sim$97% FPR.

Adversarial retraining is a viable defence. Using ProvRT-generated attack chains as adversarial training data, we train a small discriminative scoring head on top of each frozen detector encoder; the anomaly score becomes $P(\text{adversarial})$ from this head, with no reconstruction term, which isolates whether the vulnerability lies in the encoder or in the score aggregation. Replaying the saved chains against the hardened detector at 5% FPR drops ESR from 0.97 to 0.30 on Orthrus and from 0.91 to 0.0 on VELOX (Table 3). An adaptive attacker, ProvRT retrained from scratch for 10^6 steps against the hardened detector, recovers no evasion on Orthrus and 0.24 on VELOX. ProvRT was therefore exploiting the reconstruction decision boundary rather than a stable encoder weakness; on the GNN, retraining the readout closes it entirely.

4.5 ProvRT leaves a structural footprint

Stealth analysis below covers only the three vulnerable detectors, since ProvRT does not evade FLASH. ProvRT-inserted edges blend into the benign *anomaly-score* distribution but differ in operation mix, type-pair composition, and destination targeting. Figure 4 overlays per-edge anomaly-score densities: inserted distributions sit inside the benign mass with substantial overlap, so per-edge thresholding cannot separate them. This is the only dimension in which ProvRT blends in.

Figure 5 shows that benign edge-types are $\sim$85% OPEN+READ, while ProvRT selects CONNECT, SEND, RECV, and EXECUTE. At the type-pair level (Figure 7, Appendix D.2), ProvRT overuses PROCESS-out pairs and strongly under-uses the dominant benign FILE–to–PROCESS read pattern; targeted destinations also skew higher-degree than benign (Figure 6, Appendix D.1). ProvRT evades by avoiding the operations the detector knows best, not by mimicking them.

A defender can exploit this footprint. Single-statistic per-chain tests (maximum fan-in, Mann–Whitney on destination degree) fail to separate ProvRT chains from size-matched benign edge subsets, but a logistic-regression meta-detector over four footprint features (network-operation fraction, edge-type χ^2 against benign, destination fan-in, and a Mann–Whitney destination-degree statistic) reaches AUC 0.999 (TPR 100% at 5% FPR) against full-action-space ProvRT and AUC 0.96 (TPR 70%) against the stealth-constrained READ+OPEN attacker of Section 4.7 (5-fold out-of-fold ROC; Appendix E). The footprint is a viable partial defence, and constraining the action space for stealth degrades but does not eliminate detectability.

4.6 Why does FLASH resist insertion attacks?

We probe each encoder's *adversarial-to-benign sensitivity*, that is, how much a single inserted edge from an adversarial node to a benign one shifts adversarial-node scores. For each detector, we sample 200 attacker-realistic insertions ($V_{\text{adv}} \to V_{\text{benign}}$) and 200 control insertions ($V_{\text{adv}} \to V_{\text{adv}}$); Table 4 reports movement frequency, mean absolute score change, and adversarial nodes affected.

Orthrus and VELOX move adversarial scores on essentially every attacker-realistic insertion (99.5% and 100%, mean deltas 0.021 and 0.078). FLASH moves on only **2.5%** of the time, with a mean delta of 2.5×10^{-6}, four orders of magnitude below VELOX. FLASH is not globally insensitive as

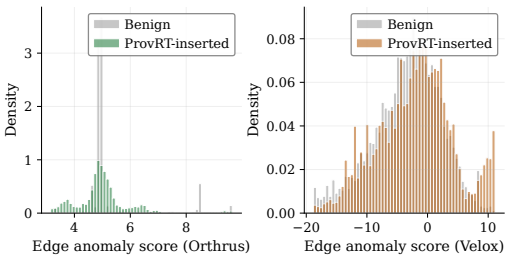

Figure 4: Per-edge anomaly score densities. Benign edges (gray) and ProvRT-inserted edges (coloured) occupy similar support with substantial overlap on both detectors

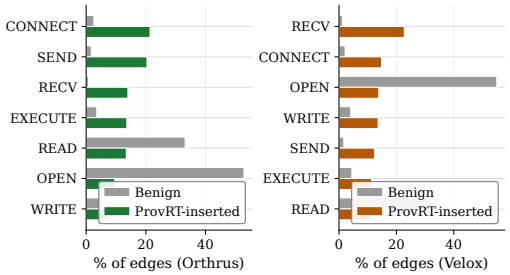

Figure 5: Top edge types as percentage of total. Benign workloads concentrate in OPEN and READ; ProvRT redistributes mass toward network and process-control operations.

Table 4: Adversarial sensitivity across detectors. We sample 200 insertions per (detector, mode) and measure movement frequency, mean absolute score change, and adversarial nodes affected per insertion. *Adv–to–Benign* is the attacker-realistic direction; *Adv–to–Adv* is a control.

PIDS	Mode	Frac. moved	Mean $\lvert\Delta\rvert$	Adv nodes affected
Orthrus	Adv–to–Benign	0.995	2.1×10^{-2}	1.97
	Adv–to–Adv	1.000	8.1×10^{-2}	6.26
VELOX	Adv–to–Benign	1.000	7.8×10^{-2}	1.00
	Adv–to–Adv	0.995	1.3×10^{-1}	1.99
FLASH	Adv–to–Benign	**0.025**	$\mathbf{2.5 \times 10^{-6}}$	**0.03**
	Adv–to–Adv	0.970	1.7×10^{-2}	2.10

97% of adversarial-to-adversarial control insertions shift scores. The encoder admits message passing between adversarial nodes but blocks propagation to benign destinations. This is the mechanism ProvRT exploits.

We test whether ProvRT can circumvent this by training on FLASH with V_{adv} to V_{adv} insertions and a 10^6-step budget: the agent reaches ESR $= -0.006$ despite the relaxed action space. The robustness-relevant property is *bounded adversarial-to-benign sensitivity*; FLASH shows it is achievable, though whether other architectures can share it remains open.

Is the vulnerability specific to reconstruction scoring? We attack two detectors that keep a GNN encoder but score each node by the cross-entropy of a node-type prediction head rather than by reconstruction error. ProvRT reaches ESR 0.83 on the CE-scored GCN and 0.19 on the CE-scored GIN. The vulnerability is therefore largely, but not exclusively, a property of reconstruction-score aggregation: per-node cross-entropy scoring does not by itself confer robustness, and FLASH's resistance is better explained by bounded adversarial-to-benign sensitivity than by its scoring rule.

4.7 Ablations and real-world execution via syscall replay

Removing the GRU recurrent trunk collapses ESR on Orthrus E3-CADETS from 0.862 to -0.244, an effect size of $\Delta = 1.1$ ESR units (Welch t-test over per-seed means, $p < 0.01$), confirming that temporal belief state is necessary to handle message-passing dynamics (Appendix C). Restricting the action space to just READ+OPEN, the two most passive, common FILE–to–PROCESS operations, retains 94% of full-action-space evasion on VELOX E3-CADETS, showing the attack is achievable with a minimal, stealth-friendly action set.

Real-world Execution. We replay a 37-insertion VELOX attack chain on an `auditd`-instrumented Docker testbed; the chain executes as 37 real Linux system calls, of which 31 produce matching audit events. The six misses are passive RECV operations requiring an active C2 peer (Appendix F). ProvRT's perturbations are executable syscalls captured by standard audit infrastructure.

5 Discussion

The constrained graph-editing MDP (Section 2) applies to any learned detector that exposes a per-node score over a provenance graph; our evidence concerns reconstruction-based scoring, where systems aggregating reconstruction error under a benign-trained encoder are exposed to systematic evasion. Two results locate the exposure. Retraining only the scoring head on ProvRT-generated chains removes the vulnerability on the GNN detector and most of it on the linear detector (Section 4.4), so much of the weakness lies in the fixed score aggregation and readout rather than in the frozen encoder; and cross-entropy-scored GNNs remain evadable (Section 4.6), so FLASH's resistance is a property of its encoder's bounded adversarial-to-benign sensitivity rather than of its scoring rule alone.

Retrofit guidance. For deployed reconstruction-based detectors, fixed aggregation swaps (median, trimmed mean) are insufficient on their own. The effective retrofit is (i) a learned discriminative head over the per-edge score distribution, trained on ProvRT-generated chains, which resists dilution at a measurable false-positive cost, complemented by (ii) a population-level distribution-shift meta-detector over edge-type and destination-degree footprints (Appendix E).

Transfer and retraining requirements. A trained policy transfers across attacks on the same detector and dataset (Orthrus, ESR $+0.97$) but not reliably across datasets (E3-CADETS-to-THEIA -0.28; THEIA-to-CADETS $+0.52$) or detectors (VELOX-to-Orthrus -0.06; Orthrus-to-VELOX $+0.28$). Policies learn dataset-specific shortcuts, consistent with the attack's architecture specificity: changing the target detector, audit schema, or host distribution requires retraining; a new attack on the same deployment does not.

Limitations. We evaluate four defender classes (robust aggregation, threshold adaptation, adversarial retraining of the scoring head, and a distribution-shift meta-detector) but not ensembles, temporal-consistency checks, or certified robustness, and adversarial retraining is evaluated on two detectors and one dataset. The defender model captures between-episode recalibration but not concept drift, weaker than a Stackelberg formulation we leave to future work. Results use single-host DARPA Transparent Computing graphs against four reconstruction-based detectors and two cross-entropy-scored GNN variants. The formulation has no single-host dependency, requiring only a provenance graph and a per-node score, with each insertion acting on a node's local neighbourhood, but multi-host validation and publicly runnable industrial PIDS, for which released weights are unavailable, remain outside our empirical scope.

Broader Impact. ProvRT is intrinsically dual-use: our results give defenders a validated retraining defence, an architectural lever, and concrete failure modes to retrain against, but trained policies could in principle inform evasion of reconstruction-based PIDS in the wild. Three factors limit this risk: (i) the attack presupposes prior code execution and control of a non-trivial set of audited objects, so ProvRT does not lower the bar for initial compromise; (ii) policies are detector-specific and require retraining when the target encoder, schema, or threshold changes (Table 11); (iii) validated defences exist (Section 4.4, Appendix E). Further safety concerns are detailed in Appendix H.

6 Related Work

Adversarial Robustness and Red-Teaming of ML Systems. Automated red-teaming is standard for adversarial evaluation of learned systems [19] and is recommended by NIST [27] and government AI-safety frameworks [1]. RL-driven red-teaming targets GNNs for hardware security [8] and LLMs for privacy [18], but no prior work addresses PIDS.

Adversarial Attacks on Graphs. Adversarial attacks on GNNs modify graph structure or node features [31, 25, 29, 30]. Dai et al. [5] propose an RL attack on citation-graph classifiers; Pierazzi et al. [20] formalise the *feature-space* vs. *problem-space* distinction (ProvRT is problem-space). None enforce security-setting constraints: inserted edges must correspond to executable syscalls, respect OS type rules, and originate from attacker-controlled nodes.

PIDS Evasion. *Mimicry attacks* [28, 26, 10] replay fixed benign patterns without adapting to detector, graph, or threshold. *Heuristic search and gadget generation* enumerates candidates per graph: PROVNINJA [17] impersonates benign "gadgets" without adaptive-defender evaluation.

Acknowledgement

This work was supported by the Australian Government through the Cooperative Research Centres Projects (CRC-P) program, in collaboration with Novatti Pty Ltd, under the project Fast and Secure Crypto Payments.

References

[1] AI Security Institute. AISI challenge fund: Priority research areas, March 2025. URL https://cdn.prod.website-files.com/663bd486c5e4c81588db7a1d/67c99c8261da5261d5553893_AISI%20Challenge%20Fund_Priority%20Research%20Areas%202025%20(1).pdf.

[2] Tristan Bilot, Baoxiang Jiang, Zefeng Li, Nour El Madhoun, Khaldoun Al Agha, Anis Zouaoui, and Thomas Pasquier. Sometimes simpler is better: A comprehensive analysis of {State-of-the-Art}{Provenance-Based} intrusion detection systems. In *34th USENIX Security Symposium (USENIX Security 25)*, pages 7193–7212, 2025.

[3] Zijun Cheng, Qiujian Lv, Jinyuan Liang, Yan Wang, Degang Sun, Thomas Pasquier, and Xueyuan Han. KAIROS: Practical intrusion detection and investigation using whole-system provenance. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 3533–3551, San Francisco, CA, USA, May 2024. IEEE. ISBN 979-8-3503-3130-1. doi: 10.1109/SP54263.2024.00005.

[4] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1179. URL https://aclanthology.org/D14-1179.

[5] Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. Adversarial attack on graph structured data. In *International Conference on Machine Learning (ICML)*, pages 1115–1124, 2018.

[6] DARPA I2O. Transparent computing engagement 3 dataset. https://github.com/darpa-i2o/Transparent-Computing, 2017.

[7] Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, et al. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. *arXiv preprint arXiv:2209.07858*, 2022.

[8] Vasudev Gohil, Satwik Patnaik, Dileep Kalathil, and Jeyavijayan Rajendran. AttackGNN:Red-Teaming GNNs in hardware security using reinforcement learning. In *USENIX Security Symposium*, 2024.

[9] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun, editors, *International Conference on Learning Representations (ICLR)*, 2015.

[10] Akul Goyal, Xueyuan Han, Gang Wang, and Adam Bates. Sometimes, you aren't what you do: Mimicry attacks against provenance graph host intrusion detection systems. In *30th Network and Distributed System Security Symposium*, 2023.

[11] Akul Goyal, Gang Wang, and Adam Bates. R-CAID: Embedding root cause analysis within provenance-based intrusion detection. In *Proceedings of the 45th IEEE Symposium on Security and Privacy (SP)*, pages 3515–3532, San Francisco, CA, USA, May 2024. IEEE. ISBN 979-8-3503-3130-1. doi: 10.1109/SP54263.2024.00253.

[12] Matthew J Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. In *AAAI fall symposia*, volume 45, page 141, 2015.

[13] Maximilian Igl, Luisa Zintgraf, Tuan Anh Le, Frank Wood, and Shimon Whiteson. Deep variational reinforcement learning for POMDPs. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2117–2126. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/igl18a.html`.

[14] Baoxiang Jiang, Tristan Bilot, Nour El Madhoun, Khaldoun Al Agha, Anis Zouaoui, Shahrear Iqbal, Xueyuan Han, and Thomas Pasquier. ORTHRUS: Achieving high quality of attribution in provenance-based intrusion detection systems. In *Proceedings of the 34th USENIX Security Symposium (USENIX Security '25)*. USENIX Association, August 2025.

[15] Shaofei Li, Feng Dong, Xusheng Xiao, Haoyu Wang, Fei Shao, Jiedong Chen, Yao Guo, Xiangqun Chen, and Ding Li. NODLINK: An online system for fine-grained APT attack detection and investigation. In *31st Annual Network and Distributed System Security Symposium, NDSS 2024*. The Internet Society, 2024. ISBN 1-891562-93-2. doi: 10.14722/ndss.2024.23204.

[16] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018.

[17] Kunal Mukherjee, Joshua Wiedemeier, Tianhao Wang, James Wei, Feng Chen, Muhyun Kim, Murat Kantarcioglu, and Kangkook Jee. Evading Provenance-Based ML Detectors with Adversarial System Actions. *USENIX Security Symposium*, 2023.

[18] Yuzhou Nie, Zhun Wang, Ye Yu, Xian Wu, Xuandong Zhao, Wenbo Guo, and Dawn Song. Privagent: Agentic-based red-teaming for LLM privacy leakage, 2024.

[19] Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. Red teaming language models with language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3419–3448, 2022.

[20] Fabio Pierazzi, Feargus Pendlebury, Jacopo Cortellazzi, and Lorenzo Cavallaro. Intriguing properties of adversarial ml attacks in the problem space. In *2020 IEEE symposium on security and privacy (SP)*, pages 1332–1349. IEEE, 2020.

[21] Mati Ur Rehman, Hadi Ahmadi, and Wajih Ul Hassan. Flash: A comprehensive approach to intrusion detection via provenance graph representation learning. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 3552–3570. IEEE Computer Society, 2024. doi: 10.1109/SP54263.2024.00139.

[22] Carlos Sarraute, Olivier Buffet, and Jörg Hoffmann. Pomdps make better hackers: Accounting for uncertainty in penetration testing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 1816–1824, 2012.

[23] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

[24] Blake E Strom, Andy Applebaum, Doug P Miller, Kathryn C Nickels, Adam G Pennington, and Cody B Thomas. Mitre att&ck®: Design and philosophy. Technical report, MITRE, 2020.

[25] Yiwei Sun, Suhang Wang, Xianfeng Tang, Tsung-Yu Hsieh, and Vasant Honavar. Adversarial attacks on graph neural networks via node injections: A hierarchical reinforcement learning approach. In *Proceedings of the Web Conference (WWW)*, 2020.

[26] Kymie MC Tan, Kevin S Killourhy, and Roy A Maxion. Undermining an anomaly-based intrusion detection system using common exploits. In *International Workshop on Recent Advances in Intrusion Detection*, pages 54–73. Springer, 2002.

[27] Apostol Vassilev, Alina Oprea, Alie Fordyce, Hyrum Anderson, Xander Davies, and Maia Hamin. Adversarial machine learning: A taxonomy and terminology of attacks and mitigations. Technical report, National Institute of Standards and Technology, 2025.

[28] David Wagner and Paolo Soto. Mimicry attacks on host-based intrusion detection systems. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 255–264, 2002.

[29] Shuiqiao Yang, Bao Gia Doan, Paul Montague, Olivier De Vel, Tamas Abraham, Seyit Camtepe, Damith C Ranasinghe, and Salil S Kanhere. Transferable graph backdoor attack. In *Proceedings of the 25th international symposium on research in attacks, intrusions and defenses*, pages 321–332, 2022.

[30] Lan Zhang, Peng Liu, Yoon-Ho Choi, and Ping Chen. Semantics-preserving reinforcement learning attack against graph neural networks for malware detection. *IEEE Transactions on Dependable and Secure Computing*, 20(2):1390–1402, 2022.

[31] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2847–2856, 2018.

A Proof of Proposition 1

Proof sketch. (i) The source head's logits are restricted to V_{adv} by masking non-adversarial nodes to $-\infty$, yielding $|V_{\mathrm{adv}}|$ active logits. Given u, the destination head evaluates only type-valid destinations (at most $|V_{\mathrm{dst}}(u)|$ active logits). Given (u, v), the edge-type head evaluates at most $|\mathcal{E}|$ types. The per-step cost is additive because sampling proceeds sequentially. *(ii)* Inclusion $\subseteq$: any action (u, v, e) with positive probability under π_θ has $u \in V_{\mathrm{adv}}$, $v \in V_{\mathrm{dst}}(u)$, and e admitted for $(\mathrm{type}(u), \mathrm{type}(v))$, which together define $\mathcal{A}(s)$. Inclusion $\supseteq$: any $(u, v, e) \in \mathcal{A}(s)$ has u adversarial (positive source-head mass), $v \in V_{\mathrm{dst}}(u)$ (positive destination-head mass given u), and $(\mathrm{type}(u), \mathrm{type}(v), e) \in \Phi$ (positive edge-head mass given (u, v)); hence $\pi_\theta(a \mid o) > 0$. □

B Experimental Setup Details

Threat model, extended. We expand on Section 2.1. The trusted computing base (TCB) includes the OS kernel, the audit daemon (`auditd` or equivalent), the provenance-collection pipeline, and the deployed PIDS weights. We assume the TCB is untampered at runtime: the attacker cannot disable auditing, suppress provenance records, or modify detector parameters. Log tampering and audit subversion are orthogonal threats with independent mitigations and outside our scope.

Position in the intrusion timeline. ProvRT operates post-compromise. It grants no initial access, privilege escalation, or exploitation capability. Instead, it helps an attacker who is already present evade the detector, producing targeted false negatives on the adversarial set while the primary objective executes. In MITRE ATT&CK terms it sits at Defence Evasion (TA0005), running concurrently with persistence, lateral movement, and exfiltration rather than at Initial Access or Execution. This is the setting PIDS are deployed to defend, so an adaptive evader here defeats the detection layer's core purpose; the attack concerns stealth and dwell time. The privilege assumed is the ability to make loggable benign actions from a controlled process. Deletion or modification of existing records would strengthen the attacker but could destroy the attack's own intent in ways that cannot be verified in replay, which is why insertion-only results are reported as a lower bound on evadability.

MITRE ATT&CK mapping. The attack primitives ProvRT uses correspond to common post-compromise techniques: T1036 (*Masquerading*, renaming or restaging files to appear benign), T1059 (*Command and Scripting Interpreter*, the EXECUTE edge class corresponding to `execve` of attacker-controlled artefacts), T1071 (*Application Layer Protocol*, RECV edges to/from attacker-controlled sockets), and T1005 (*Data from Local System*, READ and OPEN edges on adversarial files). All four are standard post-compromise techniques; ProvRT's contribution is choosing *when* and *where* to stage them to evade a specific detector's representation.

Baselines, extended. *HillClimb* is our reimplementation of a PROVNINJA-style [17] greedy per-step attacker: at each step, it evaluates candidate insertions under the transition-rule mask and commits to the one with the largest ESR improvement. *SemanticRandom* samples uniformly from the mask-valid action set. *Mimicry*, adapted from Goyal et al. [10], cycles through 14 predefined benign file-I/O patterns (OPEN/READ/WRITE/CLOSE sequences).

Table 5: ProvRT hyperparameters. Values are shared across all runs unless otherwise noted; per-configuration settings (e.g., budget) are listed where they vary.

Component	Hyperparameter	Value
Policy architecture		
Factored heads	source / destination / edge-type	3 autoregressive
Node-feature dim d	embedding or projection	128
GRU trunk	hidden size	128
GRU trunk	BPTT chunk length	64
ScoreProjector (greybox)	MLP $1 \to 128 \to d$	ReLU
LabelProjector (blackbox)	embedding dim	128
PPO training		
Optimiser		Adam
Learning rate	linear anneal to 0	3×10^{-5}
Entropy coefficient	linear anneal to 0	0.01
Rollout length		2,048 steps
PPO epochs per rollout		10
Mini-batch size		64
GAE λ		0.95
Discount γ		0.99
PPO clip ratio		0.2
Gradient norm clipping	global	0.5
Reward (Equation (6)*)*		
FNR-change weight η_{esr}		10
Per-node score weight η_{node}		1
Flip bonus η_{flip}		0
Full-evasion bonus	one-time	$\eta_{\text{esr}} \cdot B$
Environment		
Default insertion budget B		1,000
Transition-count floor $C_{\min}$	mining pruning	100
Detector-supported edge types	post-intersection	8 of 28
Training protocol		
Environment steps per run		10^6
Seeds per configuration	independent runs	5
Device		CPU

Hyperparameters. Table 5 lists the full set of hyperparameters used across all experiments.

Transition-rule mining. The transition relation Φ used at training and inference is constructed once per dataset. The miner scans every attack-window graph in the training split and counts each observed $(\text{type}(u), \text{type}(v), e)$ triple. Triples with count $\geq C_{\min} = 100$ are kept; the result is unioned with the hardcoded CDM/ECAR schema rules to admit semantically valid but rare transitions. The merged rule set is then intersected with the detector's supported edge-type set, which varies per PIDS: PIDSMaker-trained detectors consume only a subset of the 28 CDM types.

Sensitivity to the mined relation. ProvRT does not require the complete mined relation. Table 6 varies the attacker's knowledge of Φ on Orthrus E3-CADETS (attack BD12) in two ways. First, it prunes to only high-count triples; second, it randomly drops a fraction of the rules. Under pruning ProvRT retains high evasion (0.83 ± 0.14 at $C_{\min} = 1{,}000$ and 0.95 ± 0.01 at $C_{\min} = 5{,}000$), so the attack does not depend on the long tail of the mined set. Under random rule noise, it is robust on average but high-variance: most seeds keep $\sim$0.9 evasion, and the mean is pulled down by occasional collapses when a randomly removed rule happens to be important for that attack instance. Precise rule knowledge therefore reduces variance but is not required for the attack to usually succeed.

Table 6: Transition-rule sensitivity on Orthrus E3-CADETS (BD12, white-box, 5 seeds, 10^6 steps). "Prune" keeps only triples with count $\geq C_{\min}$; "Random drop" removes a uniformly random fraction of rules per seed.

Rule knowledge	ProvRT ESR
Prune to $C_{\min} = 1{,}000$	0.83 ± 0.14
Prune to $C_{\min} = 5{,}000$	0.95 ± 0.01
Random drop 10%	0.54 ± 0.38
Random drop 25%	0.50 ± 0.37
Random drop 50%	0.44 ± 0.42

C Extended Experiments

This appendix contains the full per-result tables and ablation sweeps summarised in Section 4.2 and Section 4.7.

GRU ablation. Table 7 compares ProvRT with and without the GRU recurrent trunk on Orthrus E3-CADETS across 5 seeds. The recurrent agent reaches mean ESR $= 0.862$ with 12 stable chains; the memoryless agent collapses to mean ESR $= -0.244$ with zero successful chains by 300k steps. The gap is significant at $p < 0.01$ by Welch t-test over per-seed means.

Table 7: GRU ablation on Orthrus (GNN). Temporal memory prevents policy collapse from GNN propagation unflipping previously-evading nodes. ESR and chain counts are aggregated across 5 seeds per arm; the gap between GRU and memoryless is significant at $p < 0.01$ (Welch t-test).

Agent	ESR	Chains	SPS	Outcome
GRU (chunk=64)	**0.862**	12	30	Stable, multi-step strategies
No GRU	-0.244	0	78	Collapsed at 300k steps; failed to generate chains.

Action-space restriction. Table 8 restricts the agent to just READ+OPEN, the two most passive, common FILE $\rightarrow$ PROCESS operations, on VELOX E3-CADETS. Even under this maximally constrained action space (4 source FILE nodes, 1 type-pair, 2 edge types) ProvRT retains 94% of full-action-space evasion. This is evidence that the attack surface does not depend on exotic or rarely-observed edge types: high-frequency benign operations alone suffice to move adversarial scores below threshold.

Table 8: Action-space restriction ablation on VELOX E3-CADETS (B=1000, whitebox, static defender). "READ+OPEN only" restricts the agent to the two most common, most passive file-access operations; all insertions are FILE$\rightarrow$PROCESS edges. Even under this maximally constrained action space, ProvRT retains 94% of full-action-space evasion.

Action Space	Types	Rules	Src Nodes	Best ESR	Rel.
Full (data-driven)	8	4 pairs	8 (all adv.)	0.620 ± 0.037	100%
READ+OPEN only	2	1 pair	4 (FILE only)	0.580	94%

Convergence. Table 9 reports training steps required to reach fixed ESR thresholds across seeds. On E3-CADETS, Orthrus reaches ESR ≥ 0.5 in 29k steps (5/5 seeds), ≥ 0.8 in 82k (5/5), ≥ 0.9 in 92k (4/5), and ≥ 0.95 in 97k (4/5). VELOX reaches ESR ≥ 0.5 in 185k steps (3/5 seeds) and plateaus.

Table 9: Convergence speed: training steps ($\times 10^3$) to reach ESR thresholds on E3-CADETS. Orthrus converges 3–6$\times$ faster than VELOX. Values are 5-seed means; "n/5" indicates how many seeds reached that threshold.

Configuration	ESR$\geq$0.5	ESR$\geq$0.8	ESR$\geq$0.9	ESR$\geq$0.95
Orthrus	29k (5/5)	82k (5/5)	92k (4/5)	97k (4/5)
VELOX	185k (3/5)	—	—	—

Budget sensitivity. Table 10 sweeps $B \in \{25, 50, 100, 200, 500, 1000\}$ on VELOX E3-CADETS at a fixed 100k-step training budget. ESR peaks at $B = 25$ (0.573) and degrades non-monotonically to 0.320 at $B = 1000$: larger B expands the action space the policy must explore per step, leaving the policy under-fit at fixed training budget. However, on other PIDS we found that budget needed to be higher to obtain optimal results, $B = 1000$ was chosen as midground.

Table 10: Budget sensitivity on VELOX E3-CADETS (100k training steps per budget, 5 seeds per budget). Best ESR is the mean of the best-checkpoint ESR across seeds; small budgets outperform large ones under a fixed training budget.

Budget B	Best ESR	Relative
25	**0.573**	—
50	0.483	-16%
100	0.550	-4%
200	0.360	-37%
500	0.430	-25%
1000	0.320	-44%

Policy transfer. Table 11 evaluates a policy trained on one (detector, dataset, attack) configuration when applied, without further training, to another. Transfer across attacks on the same detector and dataset is near-complete, but transfer across datasets is asymmetric and can fail outright (E3-CADETS→THEIA), and transfer across detectors is weak in both directions. Policies therefore learn dataset- and detector-specific shortcuts, consistent with the architecture specificity of the attack: changing the target detector, audit schema, or host distribution requires retraining, whereas a new attack on the same deployment does not.

Table 11: Zero-shot policy transfer. Each row applies a policy trained on the source configuration to the target configuration without retraining. Negative values indicate the transferred policy aided detection.

Transfer axis	Source → Target	ESR
Cross-attack	Orthrus, E3-CADETS	$+0.97$
Cross-dataset	Orthrus, E3-CADETS → E3-THEIA	-0.28
Cross-dataset	Orthrus, E3-THEIA → E3-CADETS	$+0.52$
Cross-PIDS	VELOX → Orthrus (E3-CADETS)	-0.06
Cross-PIDS	Orthrus → VELOX (E3-CADETS)	$+0.28$

D Additional Stealth Analysis

This appendix provides additional structural-footprint analysis for ProvRT-inserted edges, complementing the score-density and edge-type findings in Section 4.5.

D.1 Destination-degree concentration

Figure 6 plots the in-degree CDF for benign destinations against ProvRT-targeted destinations on both detectors. The ProvRT-targeted CDFs lie below benign throughout, indicating that ProvRT routes insertions toward higher-degree destinations than typical benign traffic uses. The gap is most visible at low-to-mid in-degrees, where ProvRT under-uses the long tail of low-degree benign destinations and concentrates instead on hub nodes.

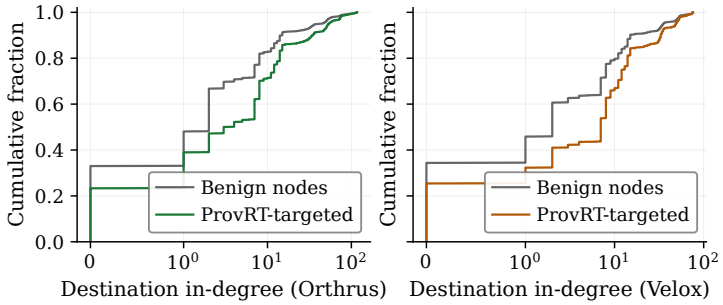

Figure 6: In-degree CDF for benign destinations (grey) vs. ProvRT-targeted destinations (coloured) on Orthrus and VELOX. ProvRT CDFs lie below benign throughout, indicating insertions concentrate on higher-degree destinations.

D.2 Source–destination type-pair distribution

Figure 7 reports the difference in source-type / destination-type frequency between ProvRT-inserted edges and the benign baseline. ProvRT over-uses PROCESS-out pairs (PROCESS → FILE, PROCESS → SOCKET) and strongly under-uses the dominant benign FILE → PROCESS read pattern. This is the type-pair level explanation for the operation-mix shift in Section 4.5: the deficit in benign READ/OPEN edges is structurally driven by the policy's avoidance of FILE → PROCESS flows, the most-frequent benign pattern.

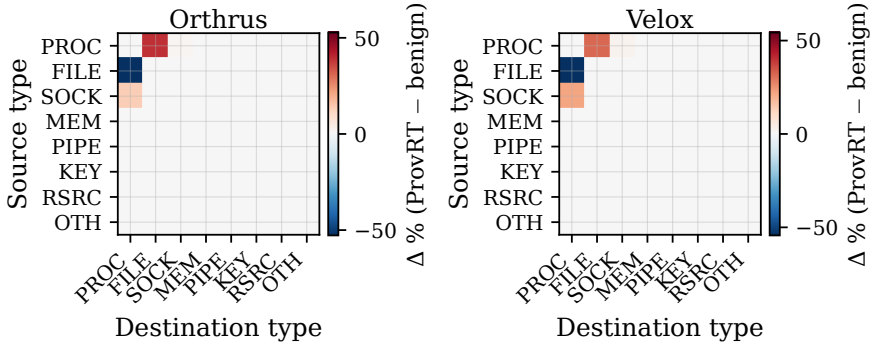

Figure 7: Difference in (source-type, destination-type) frequency between ProvRT and benign on Orthrus and VELOX. Red: ProvRT over-uses; blue: under-uses. ProvRT over-uses PROCESS-out pairs and strongly under-uses the dominant benign FILE → PROCESS read pattern.

E Per-chain Meta-Detector Experiment

Table 12: Detectability analysis of adversarial insertions across four dimensions. Higher scores indicate easier detection by a meta-detector.

Dimension	Orthrus	VELOX
Edge type distribution	0.984	0.972
Node type pair dist.	0.835	0.858
Dest. degree anomaly	0.120	0.896
Source concentration	0.238	0.111

Section 4.5 reports that ProvRT-inserted edges leave a population-level footprint: edge-type distribution is the most detectable dimension on both detectors (0.984 / 0.972 on Orthrus / VELOX, Table 12), and destination degree is strongly detectable on VELOX (0.896) but not on Orthrus (0.120). This

appendix asks the natural follow-up: can a defender turn the population-level shift into a deployable per-chain meta-detector?

We consider three candidate meta-detectors operating on a single candidate edge subset (a chain or a benign baseline of equivalent size):

Detector A (Max fan-in). Flag the candidate if the maximum number of edges in the candidate set landing on any single destination exceeds threshold K.

Detector B (Mann–Whitney p-value). Compute the Mann–Whitney U test on the in-degree distribution of the candidate's destinations against the benign in-degree distribution; flag if $p < K$.

Detector C (learned footprint classifier). A logistic-regression classifier over four features of the candidate set: the fraction of network operations (CONNECT/SEND/RECV), the χ^2 statistic of the candidate's edge-type histogram against the benign edge-type distribution, the maximum destination fan-in within the candidate, and the Mann–Whitney U statistic of the candidate's destination in-degrees against the benign distribution. Detectors A and B each use one of these statistics alone; Detector C learns a weighting over all four.

Setup. Positives are ProvRT-inserted edge sets from the 5 white-box seed runs (Orthrus on E3-CADETS BD12; VELOX on E3-CADETS BD06). Negatives are random benign edge subsets of size matched to the corresponding chain, sampled 10 times per adversarial graph. We sweep K to produce ROC curves and report AUC and TPR at 5% FPR. Detector C is trained and evaluated with 5-fold out-of-fold cross-validation on the same positives and negatives; as such, every reported score is on held-out chains. We additionally evaluate it against chains from the READ+OPEN-constrained attacker of Table 8, which avoids the network-operation and type-pair footprint by construction.

Result. Figure 8 plots the ROC curves. All four (detector $\times$ PIDS) curves lie at or below the random-classifier diagonal: AUC $\in [0.00, 0.34]$. TPR at 5% FPR is 0.00 for both detectors on Orthrus, and 0.22 (max fan-in) and 0.00 (Mann–Whitney) on VELOX. ProvRT's per-chain destination-degree distribution is statistically indistinguishable from a random benign edge subset of the same size; the population-level shift detected in Table 12 requires aggregating across many chains and seeds to surface, and does not yield a deployable per-chain signal under the simple meta-detectors we tested.

Detector C, by contrast, separates ProvRT chains from size-matched benign subsets almost perfectly: AUC 0.999 with TPR 100% at 5% FPR against full-action-space ProvRT, and AUC 0.96 with TPR 70% against the READ+OPEN-constrained attacker (Table 13).

Table 13: Learned footprint meta-detector (Detector C), 5-fold out-of-fold ROC against size-matched benign edge subsets.

Attacker set	AUC	TPR @ 5% FPR
Full-action-space ProvRT	0.999	100%
READ+OPEN-constrained ProvRT	0.96	70%

Interpretation. No single footprint statistic supports a per-chain detector, but a learned combination of four does: the population-level shifts in Table 12 are individually weak per chain and jointly decisive. The meta-detector is therefore a viable partial defence that complements adversarial retraining (Section 4.4): it operates on the inserted edges rather than on the detector's scores, so it is unaffected by the dilution that defeats fixed aggregation. We note two caveats on these results. First, an attacker that constrains its action space for stealth is measurably harder to catch (TPR falls from 100% to 70%), so the defence degrades, though it does not fail, against stealth-aware attackers. Second, Detector C is trained on ProvRT chains; an attacker that optimises against the meta-detector itself, and sliding-window variants that aggregate evidence across many chains before testing, are left to future work.

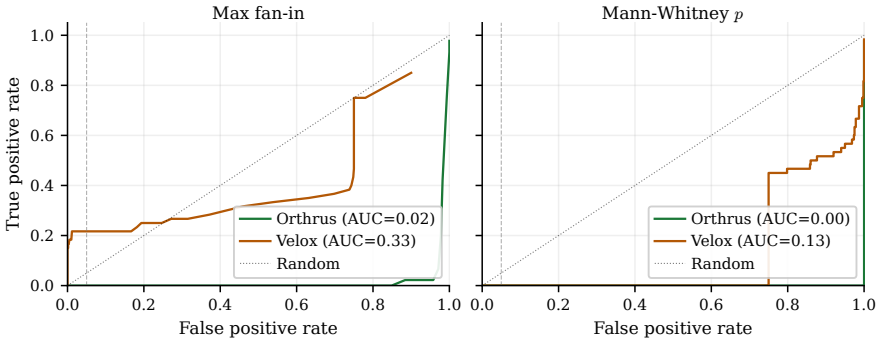

Figure 8: ROC curves for two candidate per-chain meta-detectors on Orthrus (red) and VELOX (orange). Left: max fan-in within the candidate edge set; right: Mann–Whitney p-value on the candidate's destination-degree distribution against the benign baseline. All curves lie at or below the random-classifier diagonal (dotted), indicating neither detector discriminates ProvRT chains from random benign edge subsets at the per-chain level.

F Real-World Testbed Replay

Table 14 reports a 37-insertion VELOX attack chain replayed as real Linux syscalls on an Ubuntu 22.04 Docker testbed with `auditd` capture. All 37 insertions execute without errors: READ (16/16), EXECUTE (10/10), OPEN (5/5), and RECV (6/6). Of the 37, 31 produce matching audit events; the six misses are passive RECV operations that require an active C2 peer to send data, which was not configured on our testbed. Every active operation (READ, EXECUTE, OPEN) matches perfectly. The graph-level attack therefore corresponds to executable provenance events that standard Linux audit infrastructure does capture, rather than abstract graph-editing artefacts.

Table 14: Real-world validation: attack chain replayed as Linux syscalls on Docker testbed with `auditd` provenance capture (VELOX E3-CADETS, 37 insertions). The replayed chain is the shortest successful chain across the 5 seeds; `auditd` match rates are a property of the testbed rather than a seed-variable quantity.

Edge Type	Count	Executed	Audit Matched	Syscall
READ	16	16	16	`dd if=file of=/dev/null`
EXECUTE	10	10	10	`execve(script)`
OPEN	5	5	5	`open(file, O_RDONLY)`
RECV	6	6	0	`nc -l` (passive)
Total	**37**	**37** (100%)	**31** (84%)	

G Reward Sensitivity

We ablate each reward component on VELOX E3-CADETS (attack BD06) at the standard configuration, training each variant for the same 10^6-step budget as the standard configuration. Table 15 reports the final ESR for each variant.

Two findings follow.

- **Per-node shaping matters most.** Removing η_{node} costs 0.17 ESR and removing all shaping costs 0.21, so dense per-node progress is what makes the sparse FNR signal learnable; removing the FNR-progress term alone costs only 0.04, within one standard deviation.
- **Insertion cost is immaterial.** $c_{\text{ins}} = 0.001$ and $c_{\text{ins}} = 0$ are within one standard deviation (0.384 vs. 0.407), so the per-insertion cost does not bias the policy toward shorter chains in practice; we retain the headline value for consistency with the campaign.

Table 15: Reward-component ablation on VELOX/E3-CADETS BD06 at substrate $\tau = -6.19$. Each row is mean $\pm$ std across 5 seeds, 1M training steps, budget $B = 1000$. The full reward in the headline campaign sets $\eta_{\text{esr}} = 10$, $\eta_{\text{node}} = 1$, $c_{\text{ins}} = 0.001$ (a constant per-insertion penalty omitted from Equation (6) for brevity), and $\eta_{\text{flip}} = 0$. We report Δ vs the full reward.

Reward	η_0	η_{node}	ic	ESR	Δ
Full (baseline)	10	1.0	0.001	0.384 ± 0.038	(baseline)
No η_{esr}	0	1.0	0.001	0.343 ± 0.042	-0.040
No η_{node}	10	0.0	0.001	0.214 ± 0.100	-0.170
No insertion-cost	10	1.0	0.000	0.407 ± 0.067	$+0.023$
Minimal (η_{esr} only)	10	0.0	0.000	0.176 ± 0.018	-0.208

The per-flip bonus (η_{flip}) was disabled in early experiments because positive flip bonuses induced the agent to oscillate adversarial nodes across τ via message passing, farming reward without improving ESR. The sensitivity sweep confirms keeping it disabled.

H Societal Impact and Potential Harmful Consequences

ProvRT is a red-teaming tool whose impact is intrinsically dual-use. Below, we address the categories of societal concern enumerated in the NeurIPS guidelines, in descending order of relevance to this work. We also describe our responsible disclosure process and the rationale for the open-source release.

Security. This is the primary concern for ProvRT. A trained policy could, in principle, inform evasion of reconstruction-based PIDS deployed in production, weakening an organisation's ability to detect post-compromise activity on monitored endpoints. We argued that three factors bound this risk: (i) the threat model presupposes prior code execution and control of a non-trivial set of audited objects, so ProvRT does not lower the bar for initial compromise and is irrelevant to attackers without a foothold; (ii) policies are detector-specific and require retraining when the target encoder, schema, or threshold changes, so a policy trained against one deployment does not transfer easily (Table 11); and (iii) adversarial retraining of the scoring head closes the attack vector on the GNN detector and most of it on the linear detector (Section 4.4), and FLASH demonstrates a structural alternative, giving operators concrete remediation paths rather than only a vulnerability disclosure.

We further note that our results give defenders strictly more information than attackers. The retraining recipe (Section 4.4), the footprint meta-detector (Appendix E), and the failure modes ProvRT exposes (the operation-mix, type-pair, and destination-degree footprints of Section 4.5) are immediately actionable for retraining and detection-rule augmentation, while exploiting our results offensively still requires the attacker to retrain a policy against their specific target environment.

Surveillance. PIDS are themselves a form of system-level monitoring infrastructure, deployed by organisations on hosts they administer. ProvRT does not collect, analyse, or attack data about individuals, protected characteristics, or populations; it operates on system-call provenance graphs that capture OS-level object interactions on a single host. The tool is therefore outside the scope of the surveillance concerns the guideline identifies (predicting gender, race, sexuality, or other protected characteristics, or bulk surveillance of populations).

Deception and Harassment. ProvRT generates sequences of system calls designed to evade automated detectors; it does not generate content directed at humans, impersonate public figures, or facilitate fraud, harassment, or abuse of individuals. The "deception" involved is between an attacker process and a learned classifier.

Environment. ProvRT trains on CPU within hours per (detector, dataset) pair (Section 4.3), and inference produces attack chains in under five seconds per graph. The compute footprint is small in absolute terms. We do not anticipate environmental harm from the method or its deployment.

Human Rights. The attack capability ProvRT formalises is post-compromise evasion of an enterprise security control. It does not facilitate the denial of privacy, speech, health, or legal-personhood rights to individuals; the affected party is the organisation operating the PIDS. We acknowledge that any tool that weakens monitoring infrastructure could in principle be repurposed in environments where

that infrastructure protects user data, but the bounding factors above (prior compromise required, detector-specific retraining, validated defences available)apply equally in such settings.

Safety, Discrimination, Bias and Fairness. We do not foresee applications of ProvRT to physical safety (it does not interact with cyber-physical systems, weapons, or human subjects), to discriminatory provision of services (it does not operate on data about people), or to bias along protected characteristics (its inputs are OS-level objects and edges, not demographic attributes).

Responsible disclosure. The four detectors we evaluate are academic research artefacts, and to our knowledge none is in use at organisations whose security posture our findings would directly affect. Our recommendations to the research community, namely (a) retraining the scoring head on ProvRT-generated chains, (b) deploying a footprint meta-detector alongside the PIDS, and (c) bounding adversarial-to-benign sensitivity in the encoder as FLASH does, are communicated through this paper itself and the accompanying open-source release.

Open-source release. We release ProvRT under an open-source licence alongside the paper. Automated red-teaming has become standard practice for adversarial robustness evaluation [19, 27] and withholding the tool would not prevent a motivated adversary from reproducing our results from the paper, and would deny defenders the same capability.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The three contributions stated in the abstract and introduction are supported by Section 4.2 (evasion across encoder families), Section 4.3 (query efficiency), Section 4.4 (defences and adversarial retraining), and Section 4.6 (FLASH mechanism and cross-entropy scoring). The scope (single-host DARPA TC graphs, reconstruction-based PIDS) is stated in the abstract, the Limitations paragraph of Section 5, and the Conclusion.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Section 5: results are restricted to single-host DARPA TC graphs and reconstruction-based PIDS (plus two cross-entropy-scored GNN variants); four defender classes are evaluated (aggregation, threshold adaptation, adversarial retraining of the scoring head, distribution-shift meta-detector) but not ensembles, temporal-consistency checks, or certified robustness, and adversarial retraining is evaluated on two detectors and one dataset; the defender model captures between-episode recalibration but not concept drift. Additional assumptions (fixed adversarial set, deterministic MDP, insertion-only action space) are stated in Section 2 and Section 2.1.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Proposition 1 (Tractability and Coverage of the factored masked policy in Methods) states assumptions inline. The complete proof is provided in Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The method is fully described in Methods. The MDP and threat model in Problem Setting. The Experimental Setup contains details on necessary parameters budget, FPR calibration, seed count, etc. Further details are in the Appendix. We additionally open source our method for reproducibility.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: ProvRT's source code, training scripts, evaluation scripts, and configuration files are available in the repository. The DARPA Transparent Computing E3 datasets are publicly available. Detector implementations follow PIDSMaker [2].

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Core parameters/settings appear in Methods and Experiment Setup. PPO hyperparameters, reward coefficients, and transition-rule mining details appear in Appendix B (Table 5). Released code includes configuration files.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: All non-deterministic results are reported as mean +/- standard deviation over 5 independent seeds. Error bars in figures represent +/- 1 standard deviation across seeds; GRU ablation reports Welch t-test over per seed means.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: All experiments run on CPU.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have read the NeurIPS Code of Ethics and the research conforms to it.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We include a Broader Impact paragraph in Section 5 and a full societal-impact discussion in Appendix H, covering security dual-use, surveillance, deception, environment, human rights, responsible disclosure, and the open-source release decision.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: ProvRT is dual-use. We argue open release is appropriate: (i) the targeted PIDS are research artefacts on public benchmarks, not deployed commercial systems; (ii) the threat model requires the attacker to already control adversarial nodes on a monitored host, so ProvRT does not enable initial compromise; (iii) policies are detector-specific and not directly transferable; (iv) the paper validates adversarial retraining on ProvRT-generated chains and a footprint meta-detector as defences (Section 4.4, Appendix E) and identifies a structural alternative, so the release gives defenders a tested remediation path alongside the attack.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All cited assets are credited to their original authors.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release ProvRT as open source code with documentation covering installation, training, evaluations. The repository includes configuration files for every (detector, dataset, seed) reported in the paper, and we provide an open-source licence (MIT).

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper involves no crowdsourcing or human subjects research.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper involves no human subjects research, so no IRB review was required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were not used as a component of the methodology.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.