# BASE-RATE DILUTION: A ZERO-COST ATTACK THAT ROC-AUC CANNOT SEE

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Graph anomaly detectors are commonly evaluated using class-conditional ranking metrics such as ROC-AUC, which can remain stable while operational precision collapses under changes in class prevalence. We identify a threat model, *base-rate dilution*, in which an adversary generates additional genuinely benign activity without modifying malicious events. This attack requires no perturbation budget, leaves the malicious activity unchanged, and can therefore evade defenses designed around perturbation-based threat models. We characterize the phenomenon formally: for any detector with a nonzero benign false-positive rate, precision decays as $\Theta(1/m)$ with injected benign volume $m$, while asymptotic precision invariance holds if and only if the detector is benign-null, i.e., its flagged set has zero probability under the benign generating distribution. We further distinguish this property from *score locality under benign augmentation*, which governs the invariance of ranking metrics and is generally violated by message-passing graph architectures.

We propose *feasibility-gated detection*, which composes a learned scorer with a deterministic hard mask derived from structural feasibility constraints. This construction guarantees benign-nullity independently of the learned model parameters, while allowing the learned component to distinguish malicious from benign violations within the feasible constraint-violation set. We evaluate the approach on DARPA Transparent Computing Engagement 3 provenance graphs, comparing ungated GraphSAGE, invariant-feature controls, rule-only detectors, and the feasibility-gated model. We measure dilution exponents with confidence intervals, evaluate the recall–robustness trade-off, audit the realism of benign noise, test cross-scenario transfer, and stress-test the gate against an adaptive adversary restricted to feasibility-preserving actions. We additionally reproduce the dilution phenomenon on the Elliptic Bitcoin transaction graph. The results demonstrate that ranking-metric stability can coexist with substantial precision collapse, motivating dilution robustness as a complementary evaluation dimension for graph anomaly detection.

## 1 INTRODUCTION

A deployed anomaly detector is judged, almost universally, by a class-conditional ranking metric: ROC-AUC, or occasionally average precision. These metrics ask how well a detector separates the score distribution of the positive class from that of the negative class, holding the class proportions fixed. That question is the wrong one to ask about a detector that will be run against a population whose composition an adversary can influence.

Consider a graph anomaly detector deployed over a system-provenance graph, a financial transaction graph, or any other causal or relational log in which malicious events are a small minority of all events. An adversary who wants to blunt the detector's operational usefulness does not need to disguise their own malicious edges at all. It is enough to increase the volume of ordinary, functionally irrelevant, indisputably benign activity elsewhere in the graph: spawn more legitimate processes, read more legitimate files, make more legitimate connections, or, in a financial graph, route more legitimate transactions through the monitored population. If the detector has *any* nonzero false-positive rate on benign activity, every one of these harmless events is a lottery ticket for a false



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

alarm. As the benign population grows, false positives accumulate linearly while true positives do not move, and precision is driven toward zero — not because the detector got worse at distinguishing malicious from benign, but because the population it must distinguish them from grew.

We call this attack *base-rate dilution*. It has three properties that place it outside the scope of the adversarial-robustness literature as usually formulated. First, it carries no perturbation budget: the attacker's own malicious edges are never touched, so there is nothing to bound in an $\ell_p$ ball or any other distortion metric. Second, it produces no artifact to detect: every injected edge is, by construction, drawn from (or indistinguishable from) the true benign generating distribution, so there is no statistical test that flags the injected population as synthetic. Third, it preserves attack functionality exactly, because it does not touch the attack at all — it is an attack on the *detector's operating environment*, not on the attacker's own footprint. Standard robustness defenses, which harden a classifier against bounded perturbation of the input it must classify, have nothing to say about an attacker who never perturbs that input.

The reason ROC-AUC misses this attack, and the reason the attack is worth a name at all, is a base-rate argument that goes back to Axelsson (2000): at a fixed false-positive rate, precision (equivalently, the Bayesian posterior $P(\text{malicious} \mid \text{flagged})$) is governed by the ratio of the true base rate to the false-positive rate, and degrades sharply as the negative class grows relative to the positive class. Axelsson's observation was about a *static* population; we show that in a graph setting the population is a variable the adversary controls at inference time, which turns a fact about evaluation methodology into an active attack surface. Saito and Rehmsmeier (2015) make a closely related point about ROC curves being deceptively stable under class imbalance; we sharpen this into an exact characterization for the dilution setting and identify precisely which architectural property is responsible.

This paper makes four contributions.

1. **A threat model** for base-rate dilution that is explicit about what distinguishes it from perturbation-based evasion (Section 2).
2. **A tight characterization** (Section 4): precision decays as $\Theta(1/m)$ in the injected volume $m$ whenever the detector's benign false-positive rate is nonzero, and is asymptotically dilution-invariant if and only if the detector's flagged set has zero probability mass under the benign generating distribution — a property we call *benign-null*. We further show that this property is logically independent of *score locality*, the separate condition that governs whether ranking metrics such as ROC-AUC are themselves dilution-invariant, and that message-passing graph encoders violate score locality by construction because injected background edges perturb the neighborhood aggregation of unrelated existing edges.
3. **Feasibility-gated detection** (Section 5): a scorer $s(e) = \mathrm{mask}(e) \cdot f(e)$ that composes a learned component $f$ with a non-learned, deterministic mask derived from structural feasibility constraints that are empirically zero-violation on benign activity. Because $\mathrm{mask}$ does not depend on $f$'s parameters, the benign-null property is an architectural guarantee that holds for every parameter value the learned component could take, not a property that training merely encourages.
4. **An empirical evaluation** (Sections 6–7) on DARPA Transparent Computing provenance graphs and, for generality, the Elliptic Bitcoin transaction graph, comparing the gate against an ungated baseline and a critical control (invariant indicators supplied as *features* rather than an architectural mask), measuring the dilution exponent with bootstrap confidence intervals, auditing the realism of the injected benign noise, testing cross-scenario transfer, and stress-testing the gate against a gray-box adaptive adversary restricted to feasibility-preserving actions.

## 2 Threat Model

We consider a detector operating over a causal or relational graph $G = (V, E)$ constructed from an audit trail (system-call provenance, transaction logs, or similar). A subset $E^\star \subset E$ of edges is malicious; the remainder is benign. We assume a gray-box adversary who knows the general detection paradigm (e.g., that the deployed system is a rule-based checker, a message-passing graph neural network, or some combination) but not its learned parameters or exact decision threshold.

**Attacker capability.** The attacker can generate additional, causally valid, environment-feasible benign activity anywhere in the monitored population: legitimate process spawns, file accesses, and network connections that a real user or a legitimate automated system could equally well have produced. We denote by $m$ the number of such injected benign edges.

**Attacker restriction.** The attacker does *not* modify, delete, delay, or reorder $E^\star$, the malicious edges themselves, and does not modify node or edge attributes of any existing event. This is what separates dilution from the tampering threat model studied elsewhere (edge deletion, insertion, reordering, and dependency forgery targeting the attacker's own footprint): dilution is an attack on the operating environment's composition, not on the attacker's own evidentiary trail, and the two are complementary rather than substitutes for one another.

**Attacker goal.** The attacker's objective is not to avoid detection of any specific malicious edge — recall is not the attacker's target — but to drive the detector's *precision* toward zero, so that the stream of alerts an analyst must triage is dominated by false positives, degrading the detector's operational value even if every malicious edge remains, in principle, flagged.

**Trust boundary.** We assume the audit-collection layer (kernel, hypervisor, or equivalent), the parser, and the detector itself are trusted and cannot be tampered with; only the resulting graph and its downstream consumption are in scope. We assume the attacker cannot violate invariants that are enforced by the trusted layer itself (for instance, a process cannot be observed acting before its own spawn event is recorded) — this assumption is what makes a feasibility gate meaningful at all, and Appendix A.4 evaluates what an adversary confined to this feasible region can still accomplish.

## 3 Related Work

**Base rates and intrusion detection.** Axelsson (2000) showed that even a detector with excellent per-class accuracy yields a poor Bayesian detection rate once the true base rate of intrusions is small relative to the false-positive rate, because the false-positive population dominates the alert stream. That argument treats the base rate as a fixed property of the deployment environment. We show it is instead a variable an adversary can manipulate at inference time in a graph setting, which converts a static evaluation caveat into an active, budget-free attack.

**Ranking metrics under imbalance.** Saito and Rehmsmeier (2015) demonstrate that ROC curves can indicate strong performance under severe class imbalance even when precision-recall performance is poor, because the false-positive *rate* that ROC-AUC conditions on does not reflect how that rate translates into precision once the negative population is large. Our score-locality analysis (Section 4) identifies the specific architectural property — whether a detector's per-edge score is a function of local, invariant edge structure or of graph-wide neighborhood context — that determines whether this blind spot is itself stable under an adversarially growing population, as opposed to merely present at a fixed operating point.

**Adversarial robustness.** The dominant formulation of adversarial robustness bounds a perturbation applied to the very input the classifier must decide on (Goodfellow et al., 2015; Madry et al., 2018), including graph-specific instantiations that perturb edges incident to the target node (Zügner et al., 2018). Base-rate dilution has no such perturbation: the attacker's own malicious edges are never touched, and the injected edges are not adversarial perturbations of anything but ordinary, independently generated benign activity. This places dilution outside the scope of certified or empirical defenses built around a perturbation budget, and is why we treat it as a distinct threat model rather than an instance of graph adversarial attack.

**Provenance-based intrusion detection.** HOLMES (Milajerdi et al., 2019) and UNICORN (Han et al., 2020) correlate or summarize provenance graphs to detect advanced persistent threats; ThreaTrace (Wang et al., 2022) and Kairos (Cheng et al., 2024) use graph representation learning — message-passing encoders of exactly the kind our score-locality result concerns — to score provenance events. None of these systems report a dilution-robustness measurement alongside their detection-rate or ROC-AUC numbers, and our theory predicts that their reported ranking-metric performance would be stable under exactly the attack their precision is not.

**Financial graph anomaly detection.** The Elliptic dataset (Weber et al., 2019) is a standard benchmark for GraphSAGE-family (Hamilton et al., 2017) illicit-transaction detection. We use it in Sec-

tion 7 to show that the dilution phenomenon itself, as opposed to our specific defense, is not an artifact of the provenance domain.

## 4 A Characterization of Base-Rate Dilution

Let a detector flag a subset of edges according to a score $s : E \to \mathbb{R}$ and a threshold. Let $k = |E^\star|$ be the number of malicious edges and let $p$ denote the detector's recall on $E^\star$, which by the threat model of Section 2 does not change as the attacker injects benign volume, since $E^\star$ is never touched. Let $n$ be the number of benign edges present before dilution and let the detector's false-positive rate on benign edges be $q$, assumed stable under the injection of $m$ further, similarly-distributed benign edges. Then:

$$\mathrm{TP}(m) = pk, \qquad \mathrm{FP}(m) = q(n+m), \qquad \mathrm{Precision}(m) = \frac{pk}{pk + q(n+m)}.$$

**Definition 1** (Benign-null detector)**.** *A detector is* benign-null *if its flagged set has zero probability mass under the benign generating distribution, i.e.* $q = 0$.

**Proposition 1** (Dilution rate)**.** *If* $q > 0$, $\mathrm{Precision}(m) = \Theta(1/m)$ *as* $m \to \infty$*; specifically* $\mathrm{Precision}(m) \sim \frac{pk}{q} \cdot \frac{1}{m}$. *If* $q = 0$, $\mathrm{Precision}(m) = 1$ *for every* $m$. *Consequently,* $\lim_{m\to\infty} \mathrm{Precision}(m) > 0$ *if and only if the detector is benign-null.*

*Proof.* Immediate from the closed form above: for fixed $p, k, q, n$ with $q > 0$, $\mathrm{Precision}(m) = pk/(pk + qn + qm)$ is a decreasing function of $m$ with $\mathrm{Precision}(m) \cdot m \to pk/q$ as $m \to \infty$, giving the stated rate. If $q = 0$ the denominator equals $pk$ for every $m$. □

Proposition 1 is elementary, but it has a consequence that is easy to miss in practice: *any* nonzero benign false-positive rate, however small, is eventually fatal to precision under dilution, because the false-positive count scales with the size of the population the attacker controls, while the true-positive count does not. A detector that looks excellent at a fixed, small $m$ can be arbitrarily bad at the $m$ an adversary is free to choose. This is why we build a detector (Section 5) around an architectural guarantee of $q = 0$ rather than a training objective that merely minimizes $q$.

Precision invariance is not the only thing one might want from a detector under dilution; a natural second candidate is invariance of *ranking* metrics such as ROC-AUC. We show these are governed by a logically separate property.

**Definition 2** (Score locality)**.** *Let* $S_0(e)$ *be an edge's score before dilution and* $S_m(e)$ *its score after* $m$ *benign edges have been injected elsewhere in the graph. A detector has* score locality under benign augmentation *if* $S_m(e) = S_0(e)$ *for every pre-existing edge* $e$ *and every* $m$ *(equivalently, if the rank correlation* $\rho(S_0, S_m) = 1$*).*

A deterministic rule engine that evaluates each edge against a fixed, locally-checkable predicate (e.g., "does this edge's timestamp precede its source node's spawn timestamp") has score locality by construction: its output for edge $e$ is a function of $e$'s own local structure and does not reference the rest of the graph, so injecting edges elsewhere cannot change it. A message-passing graph neural network does not: its score for $e$ is a function of aggregated features over $e$'s $k$-hop neighborhood, and both the neighborhood's composition and any global normalization statistics shift as the graph grows, even for edges arbitrarily far (in graph distance) from the injected activity, and even though nothing about $e$ or the ground truth has changed.

**Observation 1.** *Benign-nullity and score locality are logically independent, and neither implies the other. A benign-null detector can still fail score locality if its underlying scorer is context-dependent even though its* decision *is masked to zero on the benign-null set (Section 7 shows this is not the case for our gate specifically, but it is not implied by benign-nullity in general). Conversely, a detector can have perfect score locality — because it evaluates a fixed, edge-local rule — and still have* $q > 0$*, in which case its ROC-AUC and score ranking are stable under dilution while its precision is not asymptotically invariant, only asymptotically* constant at a value below one*. This second case is realized empirically in Section 7.1 by the 15-rule engine, whose precision is flat but not equal to one, and separates the two properties directly.*

The practical upshot is that reporting ROC-AUC stability under a dilution stress test is not evidence of precision stability: a detector can pass the former while failing the latter arbitrarily badly, and the architectures most commonly used for graph anomaly detection are exactly the ones for which this gap is largest, because message passing is what breaks score locality in the first place.

## 5 FEASIBILITY-GATED DETECTION

Proposition 1 motivates a detector whose flagged set is provably null under the benign distribution. The natural first attempt — supplying a learned model with features that indicate whether known structural invariants are violated, and letting training discover that these features should suppress false positives — does not work, and we treat this as a critical experimental control rather than an assumption (Section 7.1): an unconstrained scorer can always assign nonzero probability to a benign-null region, because nothing in its parameterization forbids it, and a false-positive rate that is small but nonzero is, by Proposition 1, asymptotically fatal.

We instead compose a learned scorer with a non-learned, deterministic mask:

$$s(e) = \mathrm{mask}(e) \cdot f(e), \qquad \mathrm{mask}(e) \in \{0, 1\},$$

where $\mathrm{mask}(e) = 1$ if and only if $e$ violates at least one of a set of *HARD* structural invariants, and $f$ is a learned scorer (a GraphSAGE encoder (Hamilton et al., 2017) in our implementation) whose gradient and output are only ever exposed at edges where $\mathrm{mask}(e) = 1$. Because mask does not depend on $f$'s parameters, $s(e) = 0$ for every benign-compliant edge *regardless of what the learned component's weights are*: the benign-null property is a fact about the architecture, true at initialization, after any gradient step, and under adversarial choice of parameters, not a property that training is merely encouraged to find. We verify this directly with unit tests over randomly initialized, adversarially chosen, and trained parameters (Section 7.1).

This construction has an immediate consequence for recall: $\mathrm{Recall} \leq \mathrm{Recall}_{\mathrm{HARD}}$, the fraction of true malicious edges that happen to violate a HARD invariant. The learned component cannot recover recall the mask forecloses; its role is narrower and, we argue, still necessary: *within* the masked-in (structurally infeasible) set, not every constraint violation is malicious — the same invariant can in principle be tripped by a benign edge case as by an attacker — and separating the two is exactly the discrimination task $f$ is trained for. A fixed rule cannot express this distinction because a rule either flags a violation category or does not; a learned scorer conditioned on the same structural context can.

**Constructing the HARD set.** We do not assume which invariants are benign-null; we measure it. Starting from 15 candidate structural invariants over the provenance graph schema (edge-type consistency, self-loop freedom, parent/child and process/activity temporal ordering, sequence-index continuity, and related checks), we evaluate each rule's violation rate on held-out benign corpora across two tiers — a synthetic, idealized benign generator and a more realistic one including pre-audit daemons, timestamp collisions, and duplicate audit records — for a total of 40 evaluations. We retain a rule in the HARD set only if it has *zero* violations across every trial in both tiers; rules with any nonzero benign violation rate are SOFT and excluded from the mask. In our provenance setting, 11 of 15 candidate invariants are HARD; the 4 SOFT rules include one, a sequence-monotonicity check invalidated by a global (rather than per-process) edge-numbering scheme, whose benign violation rate is catastrophic (approaching 90%) and would alone make a rule-based detector using it nearly useless independent of any attack.

## 6 EXPERIMENTAL SETUP

**Data.** Our primary evaluation uses DARPA Transparent Computing Engagement 3 provenance graphs (DARPA I2O, 2018) across four scenarios spanning three orders of magnitude in size (scenario `1r`: 408K nodes / 9.3M edges; `3`: 20K nodes / 298K edges; `5m`: 35K nodes / 465K edges; `6r`: 1.1M nodes / 18.2M edges). For domain generality we additionally evaluate on the Elliptic Bitcoin transaction graph (Weber et al., 2019), injecting held-out licit/unlabeled transaction nodes as benign dilution volume.

**Detectors compared.** (1) *Rule-only, HARD set*: the 11 provably benign-null invariants, no learned component. (2) *Rule-only, all 15 rules*: includes the 4 SOFT rules, isolating the cost of using

Table 1: Empirical Invariant Violation Rates Across Synthetic Mimicry and Resampled Benign Noise.

| Invariant Rule Name | Rule Class | Synthetic Noise (%) | Resampled Noise (%) | Violation Gap (%) |
|---|---|---|---|---|
| DeleteConsistencyRule | HARD | 2.60% | 0.00% | **+2.60%** |
| DuplicateEdgeRule | HARD | 0.00% | 0.00% | **+0.00%** |
| DuplicateEventRule | SOFT | 0.00% | 0.00% | **+0.00%** |
| ExecutionConsistencyRule | HARD | 2.83% | 0.00% | **+2.83%** |
| MissingNodeRule | HARD | 0.00% | 0.00% | **+0.00%** |
| NetworkConsistencyRule | HARD | 2.71% | 0.00% | **+2.71%** |
| ParentChildTemporalRule | HARD | 15.21% | 0.00% | **+15.21%** |
| ProcessActivityTemporalRule | HARD | 35.33% | 0.00% | **+35.33%** |
| ReadWriteConsistencyRule | SOFT | 5.23% | 0.00% | **+5.23%** |
| SelfLoopRule | HARD | 0.00% | 0.00% | **+0.00%** |
| SequenceGapRule | HARD | 0.00% | 0.00% | **+0.00%** |
| SequenceMonotonicityRule | SOFT | 77.37% | 0.00% | **+77.37%** |
| SpawnConsistencyRule | HARD | 7.77% | 0.00% | **+7.77%** |
| TimestampRule | HARD | 0.00% | 0.00% | **+0.00%** |
| UnspawnedProcessRule | SOFT | 0.00% | 1.30% | **-1.30%** |

invariants that are not benign-null. (3) *GraphSAGE, ungated*: trained end-to-end on the raw provenance graph with no invariant information. (4) *GraphSAGE + invariant features, ungated*: the same architecture given the 11 HARD-rule violation indicators as additional input features, with no multiplicative mask — the critical control described in Section 5. (5) *Feasibility-gated GraphSAGE (ours)*: $s(e) = \mathrm{mask}(e) \cdot f(e)$ using the same 11-rule mask and the same base encoder as (4).

**Protocol.** Train, validation, test, and benign-resampler-pool seeds are disjoint (train $\{0, 1, 2\}$, validation $\{3\}$, test $\{4\}$, resampler pool $\{5000, \ldots, 5009\}$), with zero edge overlap verified between splits. Operating thresholds are calibrated exactly once, on the validation split at $m = 0$, and then frozen across every seed and every dilution level; we do not re-calibrate thresholds as $m$ grows, since doing so would trivially mask precision collapse by moving the operating point rather than measuring the detector's stability at a fixed one.

**Benign noise realism.** An earlier mimicry-style noise generator used in preliminary experiments constructs background edges by drawing source, target, and timestamp independently and uniformly, which is convenient but not OS-plausible: independently drawn timestamps violate parent/child and process-activity temporal ordering at rates as high as 35% even though no attack is present, and unconstrained type pairing (e.g., a file node "spawning" a process) triggers every type-consistency rule. Table 1 audits this directly: 8 of 15 invariants show large positive violation-rate gaps under this generator relative to real benign traffic. Reporting a dilution stress test against such noise would attribute a generator artifact to the detector. We instead inject noise using a benign subgraph resampler that draws *complete, causally closed process trees* (every process together with its own spawn event and all its file/network activity) from a held-out benign pool, remaps node identifiers to avoid collision, and rebases timestamps into the target graph's time window while preserving exact relative time deltas within each sampled tree. This generator achieves a 0.00% violation gap across all 11 HARD rules (Table 1), i.e. it is indistinguishable from real benign activity with respect to every invariant our gate relies on, which is the standard we hold our own stress test to.

# 7 Results

## 7.1 Main Dilution Sweep and the Critical Control

Figure 1 shows precision as a function of injected benign volume $m \in [0, 100{,}000]$ under resampled benign noise, and Table 2 reports the corresponding fitted dilution exponents. The ungated GraphSAGE baseline decays with $\hat{\alpha} = 2.81$ (95% CI $[2.65, 3.00]$), consistent with Proposition 1's prediction of eventual $\Theta(1/m)$ decay once transient effects wash out. Both rule-only variants and the feasibility-gated detector are flat, exactly as predicted for $q = 0$.

The critical control — GraphSAGE given the same 11 invariant-violation indicators as ordinary input features, but with no architectural mask — is the load-bearing result of this section. Its precision starts at 1.000 at $m = 0$ and falls to $8 \times 10^{-5}$ by $m = 10^5$ (Figure 1, Table 2), a collapse visually

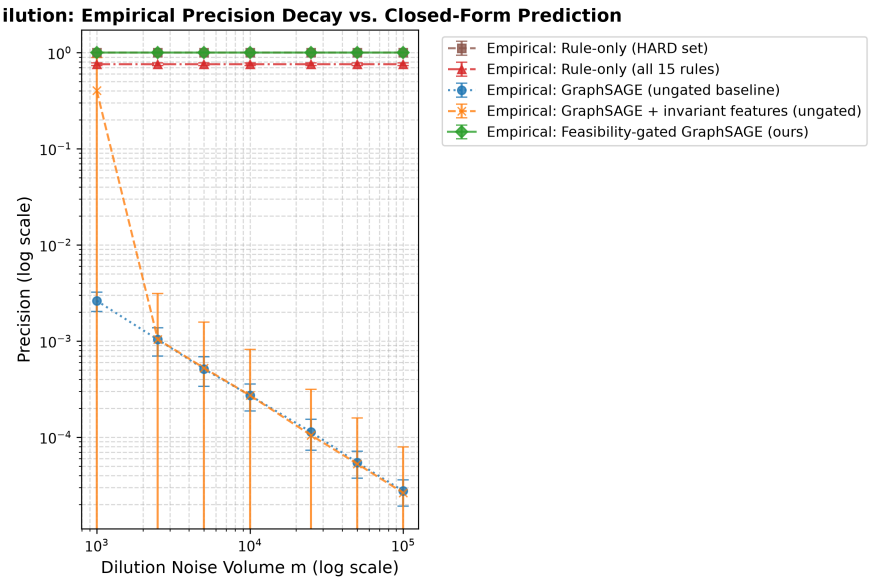


Figure 1: Empirical precision decay versus the closed-form prediction $\mathrm{Prec}(m) = pk/(pk + q(n + m))$, five detector configurations, resampled benign noise. The gated detector (green) and the HARD-rule engine (brown, overlapping) are flat at $\mathrm{Prec} = 1$; the ungated baseline (blue) and the invariant-features-as-input control (orange) both collapse.

Table 2: Dilution Exponent ($\hat{\alpha}$) and Precision Stability Across Detector Configurations.

| **Detector Configuration** | **Noise Model** | **Fitted $\hat{\alpha}$** | **95% Bootstrap CI** | $R^2$ |
|---|---|---|---|---|
| Rule-only (HARD set) | Resampled | **0.0000** | [0.0000, 0.0000] | 1.0000 |
| Rule-only (all 15 rules) | Resampled | **0.0000** | [0.0000, 0.0000] | 0.0000 |
| GraphSAGE (ungated baseline) | Resampled | **2.8129** | [2.6532, 3.0044] | 0.8151 |
| GraphSAGE + invariant features (ungated) | Resampled | **0.0000** | [0.0000, 0.0000] | 1.0000 |
| Feasibility-gated GraphSAGE (ours) | Resampled | **0.0000** | [0.0000, 0.0000] | 1.0000 |

indistinguishable from the fully ungated baseline. Feeding a network the information it would need to be benign-null is not sufficient for it to become benign-null: nothing in the training objective forces the false-positive rate to be *exactly* zero rather than merely small, and Proposition 1 shows that the difference between exactly zero and merely small is the entire story asymptotically. This is why our construction places the invariant information in a non-learned multiplicative mask rather than an input feature.

The gate's recall is bounded by the coverage of the HARD invariant set on the true positive class: in our evaluation graphs, the 11 HARD rules catch $10.7\%$ of malicious edges on average, and the gated detector achieves exactly this recall at precision $1.000$, while the ungated baseline achieves higher initial recall ($44\%$ at $m = 0$) that becomes worthless as precision collapses. A supplementary sweep that moves rules from the SOFT set into the mask one at a time, trading initial recall for dilution robustness, shows the dilution exponent falling from $\hat{\alpha} \approx 1.07$ at zero HARD rules (fully ungated) to $\hat{\alpha} \approx 0$ once enough HARD-only rules are active — the frontier is monotonic in coverage, as Proposition 1 predicts, since adding a rule to the mask can only remove flagged benign mass, never add it.

### 7.2 Ranking Metrics and Score Locality

Table 3 reports ROC-AUC (standard, and optimistic/pessimistic bounds under alternative tie-breaking, since a rule-based detector produces a large tied block of identically-scored edges) to-

Table 3: ROC-AUC and score locality at $m = 100{,}000$, resampled noise. The 15-rule engine illustrates Observation 1: perfect score locality and a flat ROC-AUC coexist with precision that is stable but not equal to one, because $q > 0$ for that configuration.

| **Detector** | **AUC (std.)** | **AUC (opt./pess.)** | $\rho(S_0, S_m)$ | **Precision** |
|---|---|---|---|---|
| Rule-only (HARD) | 0.553 | 1.000 / 0.107 | 1.000 | 1.000 |
| Rule-only (all 15) | 0.853 | 1.000 / 0.707 | 1.000 | 0.768 |
| GraphSAGE (ungated) | 0.607 | 0.609 / 0.605 | 0.658 | 0.0009 |
| Gated GraphSAGE (ours) | 0.553 | 1.000 / 0.107 | 1.000 | 1.000 |

gether with the score-locality statistic $\rho(S_0, S_m)$ at $m = 10^5$, computed over the full edge set rather than a restricted candidate set.

Both rule-based detectors hold $\rho = 1.000$ exactly at every dilution level, because a fixed, edge-local predicate cannot be perturbed by edges elsewhere in the graph; the gated detector inherits this property because its nonzero scores are confined to the same structurally-fixed HARD-violating set. The ungated baseline's $\rho$ falls from $1.000$ at $m = 0$ to $0.658$ at $m = 10^5$: message passing lets injected background edges reshape the neighborhoods, and hence the learned scores, of edges that are not themselves adversarial and have not changed. Consistent with Observation 1, the 15-rule engine's ROC-AUC is essentially flat ($0.851 \rightarrow 0.853$ across the sweep) even though its precision is not 1: score locality stabilizes the *ranking*, not the *operating-point* precision, and the two must be reported separately.

### 7.3 Cross-Scenario Transfer

We next test whether the benign-null guarantee is scenario-specific or transfers zero-shot. We evaluate dilution in both directions between DARPA scenario pairs $\{\texttt{1r}, \texttt{3}\}$ and $\{\texttt{5m}, \texttt{6r}\}$, i.e. the mask is never re-fit to the target scenario's benign traffic. All 11 HARD invariants show exactly $0.0000\%$ violation on every target scenario's *un-diluted* benign traffic (44 of 44 rule-scenario pairs; full table in Appendix A.1) — the benign-null property itself transfers cleanly across scenario shift.

The full 44-rule/scenario audit is provided in Appendix A.1; all 44 target rule-scenario pairs have zero violations on undiluted benign traffic.

The gated detector's precision nonetheless does decay under cross-scenario dilution, with $\hat{\alpha} \approx 1$ in both directions, in contrast to its exact flatness within a single scenario (Section 7.1). We traced this directly: the resampled noise subgraphs are drawn from a different scenario's clock and rebased into the target scenario's time window, and this rebasing introduces a small but nonzero residual violation rate on the *injected* edges specifically ($q_{\text{true}} = 0.0129\%$ at $m = 100{,}000$, i.e. 164 of 1,267,039 injected edges), concentrated entirely in one temporal-ordering rule. The gated detector's false positives at this dilution level number exactly 164 — a 1-to-1 match with the directly measured rule violations on the injected population, confirming that every false positive traces to a genuine (if rare) invariant violation in the noise rather than to any leakage from the learned component onto benign-compliant edges. This is the guarantee behaving exactly as specified: benign-nullity is a statement about the *benign generating distribution* the mask was calibrated against, and a distribution shift that itself introduces a small rate of genuine constraint violation will produce a correspondingly small, fully attributable degradation — far milder than the ungated baseline, whose precision under the same cross-scenario shift is either identically zero (one direction has zero true positives at every dilution level) or decays with $\hat{\alpha} = 0.56$ from an already-degraded starting point.

### 7.4 A Second Domain: Financial Graph Anomaly Detection

To test whether the dilution phenomenon extends beyond provenance graphs, we additionally evaluate an ungated GraphSAGE detector on the Elliptic Bitcoin transaction graph; the full protocol, table, and figure are moved to Appendix A.3 to preserve the main-text budget. The experiment reproduces the same dissociation between precision collapse and comparatively stable ROC-AUC.

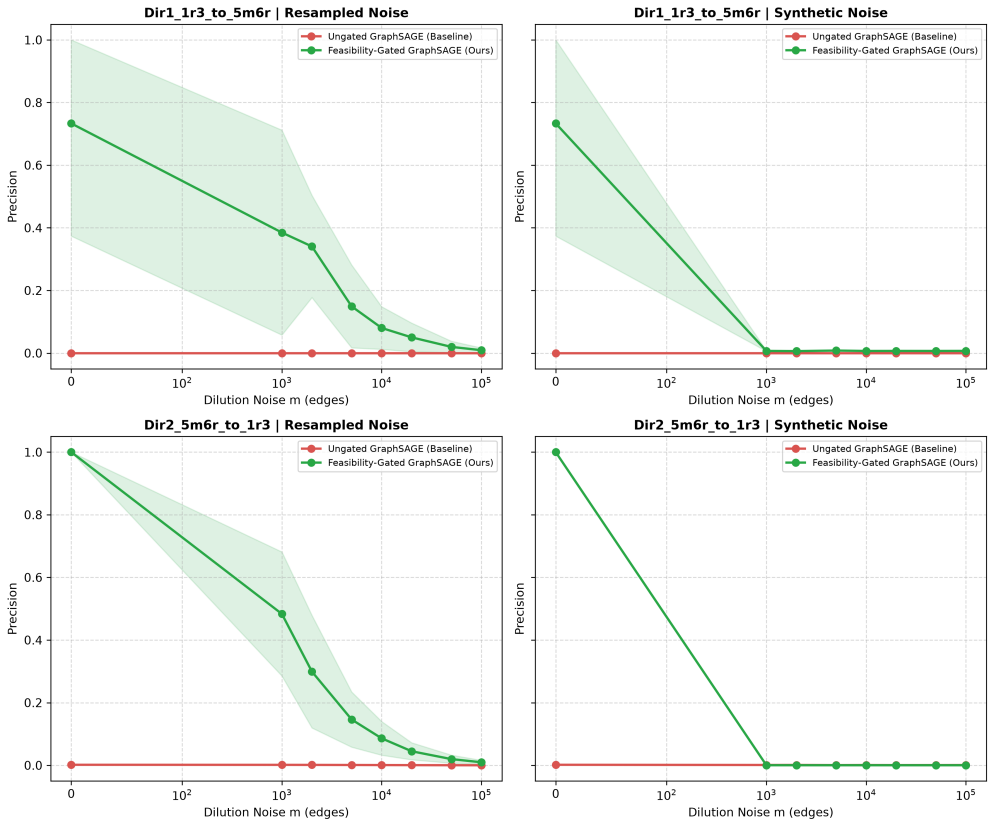


Figure 2: Precision under cross-scenario dilution in both transfer directions. The gated detector degrades only with the small residual invariant-violation rate introduced by scenario transfer.

## 8 Conclusion

Ranking metrics evaluate a detector's ability to separate two score distributions at a fixed class ratio; they say nothing about what happens when an adversary is free to change that ratio, and our results show the two questions can have arbitrarily different answers for the same detector. Base-rate dilution requires no perturbation of the attacker's own footprint and evades every defense built around bounding one. We gave an exact characterization of when precision survives it — the flagged set must be null under the benign distribution, not merely low-probability — and showed this is an architectural property that a feasibility gate can guarantee by construction, where feature-level access to the same information cannot. We also stress-test the feasibility gate against adaptive feasible actions; detailed results are provided in Appendix A.4. We hope the dilution exponent and the noise-realism audit it enables become standard companions to ranking metrics.

### Reproducibility Statement

All operating thresholds are calibrated once on a validation split at $m{=}0$ and frozen thereafter; train/validation/test/resampler-pool seeds are disjoint with zero verified edge overlap (Section 6). The closed-form precision model, the HARD/SOFT invariant partitioning procedure, and the benign subgraph resampler are fully specified in Sections 5–6. Five architectural unit tests verifying the mask's parameter-independence are described in Appendix A.2.

## AI USE STATEMENT

LLM-based tools assisted with literature mapping and organization of prior work. The authors independently evaluated the literature, selected citations, designed and conducted the experiments, analyzed results, and verified the manuscript.

## REFERENCES

Stefan Axelsson. The base-rate fallacy and the difficulty of intrusion detection. *ACM Transactions on Information and System Security (TISSEC)*, 3(3):186–205, 2000.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pages 1024–1034, 2017.

Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I. Weidele, Claudio Bellei, Tom Robinson, and Charles E. Leiserson. Anti-money laundering in Bitcoin: Experimenting with graph convolutional networks for financial forensics. In *KDD Workshop on Anomaly Detection in Finance*, 2019.

Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE*, 10(3):e0118432, 2015.

Xueyuan Han, Thomas Pasquier, Adam Bates, James Mickens, and Margo Seltzer. UNICORN: Runtime provenance-based detector for advanced persistent threats. In *Network and Distributed System Security Symposium (NDSS)*, 2020.

Sadegh M. Milajerdi, Rigel Gjomemo, Birhanu Eshete, R. Sekar, and V. N. Venkatakrishnan. HOLMES: Real-time APT detection through correlation of suspicious information flows. In *IEEE Symposium on Security and Privacy (S&P)*, pages 1137–1152, 2019.

Su Wang, Zhiliang Wang, Tao Zhou, Hongbin Sun, Xia Yin, Dongqi Han, Han Zhang, Xingang Shi, and Jiahai Yang. ThreaTrace: Detecting and tracing host-based threats in node level through provenance graph learning. *IEEE Transactions on Information Forensics and Security*, 17:3972–3987, 2022.

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 *IEEE Symposium on Security and Privacy (S&P)*, pages 3533–3551, 2024.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015.

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.

Daniel Zügner, Amir Akbarnejad, and Stephan G"unnemann. Adversarial attacks on neural networks for graph data. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, pages 2847–2856, 2018.

DARPA I2O. DARPA Transparent Computing Program: Engagement 3 Data Release, 2018. `https://github.com/darpa-i2o/Transparent-Computing`.

## A APPENDIX

### A.1 FULL CROSS-SCENARIO PARTITION TRANSFER TABLE

Table 4 reports the benign violation rate of every one of the 11 HARD invariants on the un-diluted benign traffic of every held-out target scenario referenced in Section 7.3.

Table 4: Out-of-Domain Partition Transfer Audit Across Target DARPA Scenarios.

| Target Scenario | Invariant Rule Name | Violations | Benign Violation Rate (%) | Partition Status |
|---|---|---|---|---|
| 1r | DeleteConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 1r | DuplicateEdgeRule | 0 | 0.0000% | **TRANSFERRED** |
| 1r | ExecutionConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 1r | MissingNodeRule | 0 | 0.0000% | **TRANSFERRED** |
| 1r | NetworkConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 1r | ParentChildTemporalRule | 0 | 0.0000% | **TRANSFERRED** |
| 1r | ProcessActivityTemporalRule | 0 | 0.0000% | **TRANSFERRED** |
| 1r | SelfLoopRule | 0 | 0.0000% | **TRANSFERRED** |
| 1r | SequenceGapRule | 0 | 0.0000% | **TRANSFERRED** |
| 1r | SpawnConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 1r | TimestampRule | 0 | 0.0000% | **TRANSFERRED** |
| 3 | DeleteConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 3 | DuplicateEdgeRule | 0 | 0.0000% | **TRANSFERRED** |
| 3 | ExecutionConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 3 | MissingNodeRule | 0 | 0.0000% | **TRANSFERRED** |
| 3 | NetworkConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 3 | ParentChildTemporalRule | 0 | 0.0000% | **TRANSFERRED** |
| 3 | ProcessActivityTemporalRule | 0 | 0.0000% | **TRANSFERRED** |
| 3 | SelfLoopRule | 0 | 0.0000% | **TRANSFERRED** |
| 3 | SequenceGapRule | 0 | 0.0000% | **TRANSFERRED** |
| 3 | SpawnConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 3 | TimestampRule | 0 | 0.0000% | **TRANSFERRED** |
| 5m | DeleteConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 5m | DuplicateEdgeRule | 0 | 0.0000% | **TRANSFERRED** |
| 5m | ExecutionConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 5m | MissingNodeRule | 0 | 0.0000% | **TRANSFERRED** |
| 5m | NetworkConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 5m | ParentChildTemporalRule | 0 | 0.0000% | **TRANSFERRED** |
| 5m | ProcessActivityTemporalRule | 0 | 0.0000% | **TRANSFERRED** |
| 5m | SelfLoopRule | 0 | 0.0000% | **TRANSFERRED** |
| 5m | SequenceGapRule | 0 | 0.0000% | **TRANSFERRED** |
| 5m | SpawnConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 5m | TimestampRule | 0 | 0.0000% | **TRANSFERRED** |
| 6r | DeleteConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 6r | DuplicateEdgeRule | 0 | 0.0000% | **TRANSFERRED** |
| 6r | ExecutionConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 6r | MissingNodeRule | 0 | 0.0000% | **TRANSFERRED** |
| 6r | NetworkConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 6r | ParentChildTemporalRule | 0 | 0.0000% | **TRANSFERRED** |
| 6r | ProcessActivityTemporalRule | 0 | 0.0000% | **TRANSFERRED** |
| 6r | SelfLoopRule | 0 | 0.0000% | **TRANSFERRED** |
| 6r | SequenceGapRule | 0 | 0.0000% | **TRANSFERRED** |
| 6r | SpawnConsistencyRule | 0 | 0.0000% | **TRANSFERRED** |
| 6r | TimestampRule | 0 | 0.0000% | **TRANSFERRED** |

### A.2 Gate Guarantee Unit Tests

We verify the architectural claim of Section 5 — that $s(e) = 0$ for every benign-compliant edge independent of the learned component's parameters — with five unit tests against trained checkpoints: (1) a 100-random-initialization test confirming the mask's output is identical across randomly drawn parameter values; (2) an adversarial-parameter test that explicitly searches for a parameter setting that violates the guarantee and fails to find one; (3) a gradient-isolation test confirming that gradients never flow through the masked-out region; (4) a monotone-candidate-set test confirming the masked-in edge set can only shrink or stay fixed as invariants are added, never grow; and (5) a constant-scorer test confirming the guarantee holds even when the learned component is replaced by an arbitrary constant function. All five tests pass.

### A.3 Second-Domain Evaluation: Elliptic Bitcoin Graph

We evaluate an ungated GraphSAGE detector on the Elliptic Bitcoin transaction graph (Weber et al., 2019), injecting held-out licit/unlabeled nodes as dilution volume under a fixed operating threshold. This experiment tests the dilution phenomenon rather than the feasibility-gated defense: Elliptic has no OS-kernel-enforced causal invariant analogous to the HARD rule set. Precision falls from 0.878 to 0.070 ($\hat{\alpha} = 0.629$, 95% CI $[0.614, 0.641]$, $R^2 = 0.947$), while ROC-AUC changes from 0.975 to 0.891.

Table 5: Empirical Performance Metrics under Dilution on the Elliptic Bitcoin Graph Anomaly Benchmark.

| **Dilution $m$ (nodes)** | **Precision** | **Recall** | **F1 Score** | **ROC-AUC** | **Spearman $\rho$** |
|---|---|---|---|---|---|
| 0 | 0.8778 | 0.7725 | 0.8215 | 0.9746 | 0.0268 |
| 1000 | 0.8007 | 0.6932 | 0.7428 | 0.9498 | -0.0554 |
| 5000 | 0.5695 | 0.6930 | 0.6249 | 0.9278 | -0.0645 |
| 10000 | 0.4341 | 0.6947 | 0.5338 | 0.9151 | -0.0522 |
| 20000 | 0.2846 | 0.6943 | 0.4035 | 0.9035 | -0.0413 |
| 50000 | 0.1368 | 0.6963 | 0.2285 | 0.8923 | -0.0423 |
| 100000 | 0.0698 | 0.6972 | 0.1268 | 0.8910 | -0.0769 |

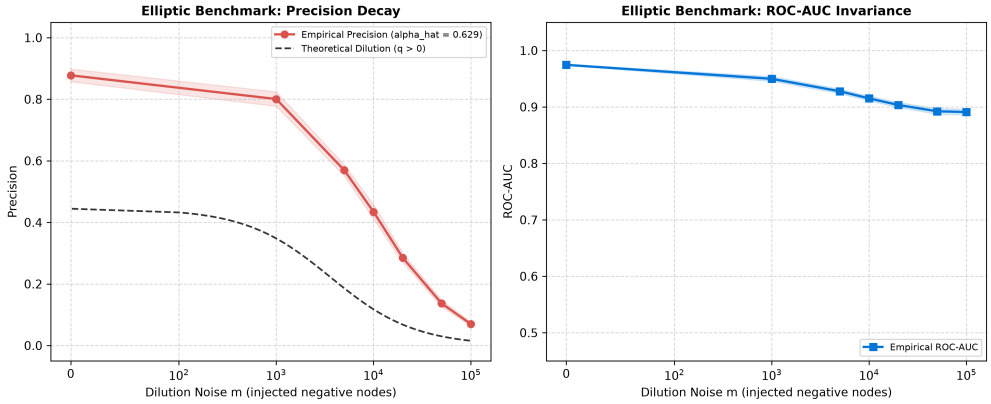


Figure 3: Elliptic Bitcoin benchmark under dilution: empirical precision decay against the closed-form $q > 0$ prediction and ROC-AUC. Precision collapses by more than an order of magnitude while ROC-AUC changes comparatively little.

### A.4 Adaptive Adversary Capability Certificate

A feasibility gate is defenseless against an attacker who accomplishes their objective entirely through HARD-invariant-compliant actions. We stress-test this directly with a gray-box adaptive adversary that knows the 11 HARD rules but not the learned parameters, attempting 7 attack objectives via search over feasible edge sequences, 50 trials per objective (5 seeds $\times$ 10 trials).

Four objectives (credential access, lateral movement, exfiltration, and log tampering) are achieved in every trial without violating a HARD invariant. Unspawned stealth execution requires inserting a genuine spawn edge, at $2\times$ edge overhead. Backdated timestamp tampering succeeds in 5 of 50 trials, exclusively for root processes lacking a parent spawn edge; self-loop execution is unachievable in all 50 trials. On every achieved feasible realization, the gated detector's score is exactly $0$. These results delimit the scope of the defense: it protects against base-rate dilution, not arbitrary invariant-compliant intrusion.

## B Discussion: Dilution Robustness as a Reportable Diagnostic

Two practical uses follow directly from Proposition 1. First, the fitted dilution exponent $\hat{\alpha}$ is itself a diagnostic for whether a reported robustness stress test used realistic benign noise: Section 6 shows that an implausible noise generator manufactures spurious invariant violations that masquerade as detector weakness, and comparing $\hat{\alpha}$ under a noise-realism audit (Table 1) exposes this before it contaminates a reported result. Second, because ROC-AUC and $\hat{\alpha}$ are governed by different, independent architectural properties (Observation 1), reporting one does not bound the other, and a benchmark that reports only ranking metrics gives no evidence either way about dilution robustness. Following Axelsson (2000), whose base-rate analysis of static intrusion-detection deployments mo-

Table 6: Adaptive Adversary Capability Certificate and Detection Cost under Feasible Search.

| Objective Name | Achieved? | Infeasible Cost | Feasible Cost | Effort Overhead | Gated Score | Certificate Verdict |
|---|---|---|---|---|---|---|
| credential_access | YES | 1.0 | 1.0 | 1.0x | 0.0000 | Feasible realization achieved with 1.0x edge overhead cost (range [1.0x, 1.0x]) |
| lateral_movement | YES | 2.0 | 2.0 | 1.0x | 0.0000 | Feasible realization achieved with 1.0x edge overhead cost (range [1.0x, 1.0x]) |
| exfiltration | YES | 2.0 | 2.0 | 1.0x | 0.0000 | Feasible realization achieved with 1.0x edge overhead cost (range [1.0x, 1.0x]) |
| log_tampering | YES | 1.0 | 1.0 | 1.0x | 0.0000 | Feasible realization achieved with 1.0x edge overhead cost (range [1.0x, 1.0x]) |
| unspawned_stealth_exec | YES | 1.0 | 2.0 | 2.0x | 0.0000 | Feasible realization achieved with 2.0x edge overhead cost (range [2.0x, 2.0x]) |
| backdated_timestamp_tampering | YES | 1.0 | 1.0 | 1.0x | 0.0000 | Feasible realization achieved with 1.0x edge overhead cost (range [1.0x, 1.0x]) |
| self_loop_execution | NO | 1.0 | 0.0 | 0.0x | 0.0000 | UNACHIEVABLE without violating rule(s): ExecutionConsistencyRule, SelfLoopRule |

tivated reporting the Bayesian detection rate rather than the true/false-positive rates alone, we argue graph anomaly detection benchmarks should report an explicit dilution-robustness measurement — at minimum, precision at several injected volumes, or the fitted exponent $\hat{\alpha}$ — alongside any ranking metric.

# C LIMITATIONS

The gate's recall ceiling is a real cost, not an incidental one: it is bounded by the fraction of true malicious edges that happen to violate a provably benign-null invariant, which was $10.7\%$ in our evaluation graphs, and a domain in which no deterministic invariant is both benign-null and broadly triggered by attacks will see a correspondingly tighter ceiling. The construction also presupposes that such invariants exist and can be identified empirically; our second-domain experiment (Section 7.3) evaluates only the dilution phenomenon, not the gated defense, precisely because financial transaction graphs lack an analogous kernel-enforced invariant structure, and extending feasibility gating to domains without such structure is open. Finally, Appendix A.4 shows the gate provides no protection against an adversary who accomplishes their objective without ever leaving invariant-compliant territory; feasibility gating is a defense against base-rate dilution specifically, and should be paired with, not substituted for, detection mechanisms aimed at stealthy but invariant-compliant intrusions.

## C.1 DATASET STATISTICS

Table 7: DARPA TC E3 scenario sizes.

| Scenario | Nodes | Edges | Avg. in-degree | Avg. total degree |
|---|---|---|---|---|
| 1r | 408,320 | 9,295,127 | 22.76 | 45.53 |
| 3 | 20,157 | 297,777 | 14.77 | 29.55 |
| 5m | 34,835 | 464,858 | 13.35 | 26.69 |
| 6r | 1,123,475 | 18,206,475 | 16.21 | 32.41 |