PALADIN: PRIVACY-AWARE LEARNING WITH ADVERSARY-DETECTION AND INFERENCE SUPPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Agents trained via Reinforcement Learning (RL) and deployed in sensitive settings, such as finance, autonomous driving, or healthcare, risk leaking private information through their observable behaviour. Even without access to raw data or model parameters, a passive adversary may infer sensitive attributes (e.g., identity, location) by observing the agent's trajectory. We formalise this *behavioural leakage* threat and propose **PALADIN**, a proactive privacy-shaping framework that integrates an adversarial inference model into the training loop. PALADIN jointly trains a transformation network to perturb observations and a co-adaptive leakage predictor, whose output shapes the agent's reward via a curriculum-guided penalty. This allows the agent to first learn stable task policies, then progressively adapt its behaviour to resist inference. We evaluate PALADIN on autonomous navigation and financial trading, auditing leakage against multiple adversary architectures (MLP, GRU, Transformer). PALADIN achieves up to 43% (return 27.0 vs. 18.9 baseline) higher task returns and 57% (0.056 vs 0.131) lower adversarial leakage compared to strong baselines. Even against Transformer adversaries, where leakage confidence remains high, PALADIN raises returns by 38% (22.8 vs.15.9) without amplifying leakage, whereas static noise and Differntial Privacy (DP) baselines (returns less than 7) fail to reduce leakage. These results highlight the value of embedding adversary-aware privacy shaping directly into RL training to mitigate deployment-stage inference threats.

1 INTRODUCTION

The rapid adoption of intelligent systems in safety-critical and data-sensitive domains has intensified the challenge of balancing task performance with information privacy (Dwork et al. (2006); Abadi et al. (2016)). Domains such as autonomous driving, finance, and healthcare increasingly deploy agents that rely on sensitive data for real-time decision-making (Heaton et al. (2017); Miotto et al. (2018)). However, these agents often operate in *adversarially observable environments* (Goodfellow et al. (2014b)), where emitted behaviours, such as actions, trajectories, or timing patterns, can be exploited to infer sensitive information.

While existing approaches such as DP (Abadi et al. (2016)), homomorphic encryption (Aono et al. (2017)), and secure multi-party computation (Shokri & Shmatikov (2015)) offer strong training-time guarantees, they fail to prevent *behavioural leakage* at deployment: the unintended exposure of sensitive user information through an agent's observable actions and trajectories. This threat can arise in adversarially observable environments (Goodfellow et al. (2014b)), where a passive observer can monitor agent emissions over time. For example, an adversary observing the GPS traces of a taxi fleet may infer passenger identity; a financial bot's trade timings may reveal proprietary strategies (Hilprecht et al. (2019); Carlini et al. (2021)). Crucially, these threats operate *post hoc* differentiating them from classical DP (which protects training data) and adversarial robustness (which defends against performance degradation via perturbation). Recent RL work has extended DP guarantees to policies (Yang-Zhao & Ng (2024); Balle et al. (2016)), injecting noise into gradients or trajectories. However, these methods target population-level or training-time privacy, not trajectory-level semantic cues. Adversarial RL (Pinto et al. (2017); Gleave et al. (2020)) focuses on robustness to environmental shifts or reward attacks, but not inference suppression. Similarly, Information-theoretic methods

(Zhao et al. (2020); Noorbakhsh et al. (2024)) limit mutual information in supervised settings, but are not optimised for online, temporally evolving agent behaviour. As a result, current methods leave agents vulnerable to deployment-time leakage.

We propose **PALADIN** (**P**rivacy-**A**ware **L**earning through **A**dversarial **D**efence and **IN**ference suppression), a proactive RL framework that suppresses behavioural leakage. PALADIN embeds a co-trained adversarial inference model into the reward loop, providing real-time estimates of semantic leakage. It shapes the policy to minimise this risk through a *curriculum-guided privacy penalty*: early training prioritises task mastery, while later phases introduce increasingly strict privacy shaping. This two-phase process is critical. Techniques such as uniform noise injection, and fixed penalties destabilise learning and underperform on the privacy–utility trade-off. PALADIN instead adapts dynamically to the adversary's performance, modulating penalties based on empirical inference risk. This makes it compatible with real-world RL settings where leakage evolves during learning and no single noise level is optimal. We instantiate PALADIN on two real-world benchmarks i) **AV-GPS dataset** (Abrar et al. (2024)): an autonomous driving task where GPS traces leak driver identity; and ii)**Financial Trading (NYSE) dataset** (Gawlik (2017)): a multi-asset RL trading environment where order timing reveals strategy class. PALADIN consistently improves the privacy–utility trade-off, for example, on the AV–GPS benchmark, PALADIN improves task return from 18.9 to 27.0 while reducing leakage F1 from 0.20 to 0.05. On the finance benchmark, it achieves similar gains: against an MLP–Transformer adversary, return increases from 16.7 to 24.8 while leakage drops from 0.31 to 0.05; against a Transformer–GRU adversary, return rises from 16.4 to 20.5 with leakage halved (`nll` from 0.28 to 0.09). Even in the most challenging GRU–Transformer case, where leakage confidence remains saturated (≈ 0.98), PALADIN maintains the highest utility (19.6 vs. 17.8 baseline). PALADIN shows it is domain agnostic as across both domains, it consistently outperforms baselines including DP (DP-RL, DP-Nash) and static noise injection, which either collapse utility (returns < 7) or leave leakage unresolved. Hence, the contributions of this work are threefold:

- We empirically formalise and evaluate *deployment-time behavioural leakage* as a distinct privacy threat in RL, which persists even after robustness or training-time DP defences.
- We propose **PALADIN**, a curriculum-guided, adversary-in-the-loop framework that proactively shapes policy behaviour to suppress semantic leakage while preserving utility.
- We provide extensive empirical validation on autonomous navigation and financial trading, showing consistent gains in utility and reductions in leakage over strong baselines, including Differential Privacy (DP) and adversarial training methods.

For reproducibility, we will release our code upon acceptance.

2 Related Work

Early efforts to protect sensitive information in learning systems are dominated by DP mechanisms and adversarial inference attacks. Approaches such as Differentially Private-Stochastic Gradient Descent (DP-SGD) (Abadi et al. (2016)), while providing formal guarantees, often reduce task utility and introduce considerable computational overhead (Shokri & Shmatikov (2015); Li et al. (2024)). Alternative cryptographic techniques, such as homomorphic encryption (Aono et al. (2017)) and secure multi-party computation (Shokri & Shmatikov (2015)), provide even stronger formal guarantees. However, the high computational costs and latency render them impractical for real-time applications such as autonomous driving, financial trading, or edge computing. Critically, these classical approaches focus primarily on training-time or parameter-level privacy, and fail to mitigate *behavioural leakage*, such as inference from an agent's emitted actions at deployment time.

In RL, DP has been applied to rewards (Balle et al. (2016)) and noisy gradients (Yang-Zhao & Ng (2024)), but guarantees remain at the parameter level (Sajed & Sheffet (2019); Qiao & Wang (2023)). In contrast, behavioural leakage arises when adversaries infer latent attributes (e.g., identity or strategy class) from trajectories (Hilprecht et al. (2019); Carlini et al. (2021); Pan et al. (2019)). Related work on adversarial robustness trains RL agents to resist perturbations (Pinto et al. (2017); Gleave et al. (2020)), but assumes active attackers degrading performance rather than passive observers inferring attributes. PALADIN's use of a co-trained adversary resembles adversarial training frameworks, including GAN-based privacy learning (Goodfellow et al. (2014a)) and DP self-play (Qiao & Wang (2024)), but differs by targeting single-agent trajectory privacy, embedding the adversary in the

reward loop, and stabilising training via curriculum-guided penalties. Unlike adversarial debiasing, which suppresses correlations in supervised tasks, PALADIN operates in sequential settings and suppresses inference from full trajectories.

Another relevant line of work involves information-theoretic regularisation, such as the variational information bottleneck (Alemi et al. (2017); Achille & Soatto (2018)), and its extensions to privacy-preserving learning (Zhao et al. (2020); Noorbakhsh et al. (2024)). These methods provide representation-level guarantees but are less applicable to dynamic RL, where leakage arises from temporal dependencies in sequences. PALADIN complements these approaches by addressing the trajectory-level leakage that persists even after DP or robustness defences. Unlike static DP noise injection, PALADIN adapts its privacy shaping online via an adversary's inference success, using a curriculum to balance task mastery with progressive hardening against leakage.

To our knowledge, PALADIN is the first to integrate online adversarial inference, curriculum-guided shaping, and policy training in a unified RL framework. It reframes privacy as a proactive, adversary-aware signal within optimisation, rather than a rigid external constraint. While it does not provide worst-case (ε, δ)-DP or information-theoretic bounds, PALADIN offers an empirical mechanism suited to settings where such guarantees are infeasible, for instance when sensitive variables are latent and unsupervised. Behavioural leakage thus remains a critical risk even when training-time DP or robustness defences are available, and it provides a practical, deployment-oriented mitigation.

3 METHODOLOGY

In this section, we introduce **PALADIN**, a proactive privacy–shaping framework that trains RL agents to suppress semantic leakage from their emitted behaviour while maintaining task performance. The key idea is to embed a co-trained adversary into the agent's reward loop, enabling privacy shaping as an explicit optimisation objective. We first formalise the privacy threat (Section 3.1), describe PALADIN's architecture and training loop (Section 3.2), and present a theoretical bound on adversarial inference (Section 3.3).

3.1 THREAT MODEL

We consider a *deployment-time* privacy threat where an external adversary passively observes an agent's trajectory and attempts to infer a sensitive attribute $y \in \mathcal{Y}$ (e.g., identity or strategy class). The adversary has no access to model internals, only to the transformed observation sequence $\tilde{X} = (\tilde{x}_1, \ldots, \tilde{x}_T)$ emitted by a policy π_θ with perturbation f_ϕ. Behavioural leakage is the adversary's prediction success, measured by $L_{\text{leak}} = \ell(h(\tilde{X}), y)$. For example, in autonomous driving y indicates GPS spoofing status, while in financial trading y denotes a latent trading strategy. In both cases, y is inferred solely from trajectories.

This threat differs from *differential privacy*, which protects training data, and from *adversarial robustness*, which preserves task performance under perturbations. It also differs from adversarial debiasing and DP-RL, which perturb inputs, rewards, or gradients at training time. Prior work on GAN-based privacy or robustness focuses on representation obfuscation, but cannot prevent semantic inference from observable actions. PALADIN instead targets trajectory-level leakage, embedding a curriculum-guided privacy penalty directly into the reward loop.

Problem Formulation: We model the environment as an MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$ with states s_t, actions a_t, and rewards $r(s_t, a_t)$. Observations $x_t \sim \mathcal{O}(s_t)$ are transformed as $\tilde{x}_t = f_\phi(x_t)$ before being consumed by the policy π_θ. The objective is to maximise expected return while bounding leakage:

$$\max_{\theta,\phi} \ \mathbb{E}_\tau \left[\sum_{t=1}^{T} \gamma^t r(s_t, a_t) \right] \quad \text{s.t.} \quad \mathbb{E}[L_{\text{leak}}(\tilde{X}, y)] \leq \varepsilon. \tag{1}$$

We approximate this with leakage-aware reward shaping:

$$r_t^{\text{total}} = r(s_t, a_t) - \lambda_t \, \widehat{L}(\tilde{X}_{1:t}), \tag{2}$$

where $\widehat{L}$ is a differentiable leakage proxy and λ_t is a curriculum-driven penalty.

3.2 PALADIN FRAMEWORK

PALADIN is a closed-loop feedback mechanism in which a RL agent learns not only to maximise its task reward but also to minimise the amount of sensitive information that an external observer could infer from its behaviour. This is achieved through four interlocking modules discussed below:

Transformation Network: Each raw observation $x_t \in \mathbb{R}^d$ is processed by a neural network f_ϕ, which learns to generate privacy-preserving perturbations. Concretely, we apply: $h_t = \mathrm{ReLU}(W_1 x_t + b_1), \quad z_t = W_2 h_t + b_2, \quad \tilde{x}_t = x_t + z_t$. where, W_1 and W_2 are learned weight matrices of dimensions $k \times d$ and $d \times k$ respectively. The residual connection ($x_t + z_t$) preserves critical information for task performance while allowing the network to obfuscate sensitive features.

Surrogate Leakage Predictor: We then introduce a surrogate adversarial predictor g_w to measure how much sensitive information remains in the transformed sequence $\tilde{X} = (\tilde{x}_1, \ldots, \tilde{x}_t)$. This network attempts to infer the sensitive attribute y, providing a real-time proxy for adversarial inference capability. It processes recent observations via flattened sequences and layer-normalised projections, outputting class scores $o \in \mathbb{R}^C$. Thus, the leakage loss is computed as:

$$\mathcal{L}_{\mathrm{leak}}(w) \;=\; -\sum_{c=1}^{C} \mathbf{1}\{y = c\}\, \log\big[\mathrm{softmax}(o)\big]_c, \tag{3}$$

updating w after each episode so that the leakage predictor g_w remains a strong proxy for the true adversary.

Curriculum-Guided Reward Shaping: We wrap the environment's native reward r_t in a custom Gym interface to incorporate privacy shaping. At each step, the wrapper performs the following tasks: i)Buffers the last T transformed observations $\tilde{X}_{1:t}$. ii) Queries the leakage predictor g_w to obtain its maximum posterior probability ($\ell_t = \max_c \mathrm{softmax}(o)_c$). iii) Computes a shaped reward

$$r_t^{\text{shaped}} \;=\; r_t \;-\; \lambda_t\, \ell_t, \tag{4}$$

where λ_t is a curriculum penalty weight that increases over training to gradually harden privacy constraints. Finally, Tracks the cumulative difference $\sum_t \|\tilde{x}_t - x_t\|^2$ as a fidelity diagnostic, to monitor perturbation magnitudes. The curriculum design allows the agent to first focus on task mastery (by starting with $\lambda_t = 0$) before gradually adapting its policy to withstand adversarial inference.

RL Agent: We then employ standard off-the-shelf RL (e.g.Proximal Policy Optimisation (PPO) (PPO)), modified so that the agent receives the transformed observations $\tilde{x}_t$ directly as input to its policy $pi_\theta(a \mid \tilde{x})$ and value functions. In both cases, we inject f_ϕ as a custom feature extractor so that the agent's policy $\pi_\theta(a \mid \tilde{x})$ and value or critic networks receive the transformed, privacy-shaped observations directly. Then, gradient updates flow simultaneously through the policy parameters θ and the transformation parameters ϕ, thereby coupling task performance and privacy objectives. This closed-loop design embeds an adversary *in the training process* itself, offering an adaptive, data-driven defence: as the agent becomes more private, the adversary retrains to find new leakage and the agent responds in turn. By integrating each component tightly and training them jointly, PALADIN transforms privacy from a post-hoc consideration into a core, first-class objective of the learning process. This will enable agents not only to perform their tasks but also to actively protect sensitive information in real time. We summarise PALADIN's joint training loop in Algorithm 1, and our architectural pipeline can be visualised in Figure 1 (–see Section A).

3.3 THEORETICAL PRIVACY GUARANTEE

While PALADIN does not offer worst-case differential privacy, we provide a bound that links task performance and leakage suppression. We emphasise that this bound is not a formal (ε, δ)-DP guarantee. It relies on the assumption that the surrogate adversary provides an unbiased estimate of leakage, which may not strictly hold in practice if the adversary underfits or if the data distribution shifts. Hence, the bound should be interpreted as a heuristic shaping signal that guides optimisation, rather than a worst-case guarantee of privacy.

Algorithm 1 PALADIN: Joint Privacy-Shaping RL

Require: curriculum $(t_i, \lambda_i)_{i=0}^{K}$, predictor init w
1: Initialize policy parameters θ, transformer ϕ, predictor w
2: **for** episode = 1 **to** M **do**
3: Set $\lambda = \lambda_{\max\{i:t_i \leq \text{episode}\}}$
4: Collect rollout (s_t, a_t, r_t) with observations $\tilde{x}_t = f_\phi(x_t)$
5: Compute $\widehat{L}(w; \tilde{X}, y)$
6: Form shaped rewards via equation 4
7: Update (θ, ϕ) by RL algorithm on shaped rewards
8: Update predictor w via $\nabla_w \widehat{L}$
9: **end for**

Theorem 1 (Privacy–Utility Bound)**.** *Let $\widehat{L} = -\log p_w(y \mid \tilde{X})$ be an unbiased estimator of adversarial loss with variance σ^2, and let total reward $R = \sum_t r_t$ be bounded by $R_{\max}$. Then under a final penalty λ_K, the expected adversarial loss is lower-bounded by: $\mathbb{E}[\ell(h(\tilde{X}), y)] \geq \frac{\mathbb{E}[R] - \epsilon R_{\max}}{\lambda_K} - \sigma$, where ϵ captures residual convergence error.*

This provides a loose but interpretable trade-off: stronger privacy penalties (λ_K) or lower task reward imply higher expected adversarial error. Full proof in Appendix B.

4 Experiment

We empirically validate PALADIN's ability to balance privacy and utility across multiple domains. Here, we discuss our experiment based on the Autonomous Vehicle related dataset, and we detail the Finance sector in Section E. We start with setting up benchmark and then provide information on baselines.

4.1 Benchmark setup

AV-GPS dataset (Abrar et al. (2024)) is a collection of GPS navigation data collected by the Autonomic Computing Lab GPS-guided Rover (ACL-Rover) at the University of Arizona. Each data point is labelled as **normal (N)** or **attack (A)**, corresponding to the presence of GPS spoofing, which we consider as a sensitive attribute. The dataset comprises four subsets collected under normal and varied spoofing scenarios. Specifically, AV-GPS-Dataset-1 contains 62,042 records (46,287 (N) vs. 15,755 (A)), AV-GPS-Dataset-1-Normal contains 46,287 (N) records, AV-GPS-Dataset-2 totals 6,890 samples (5,184 (N); 1,706 (A)), and AV-GPS-Dataset-3 includes 636 records with 231 (N) and 405 (A) labels. All subsets contain 44 features (– see Section D for details on the features). We combined the subsets and utilised 115,855 samples for our experiments. We impute the missing values using the empirical mean and apply min–max normalisation to each feature independently over its entire trace. We then formed episodes by segmenting trajectories into fixed-length windows of $T = 100$ steps.

Each 100-step GPS segment in our dataset is labelled **N** or **A** (spoofed/jammed), and we train the policy adversarially so that an observer can no longer distinguish attacked from normal segments based on the shared GPS trace. A practical adversary, e.g. a malicious eavesdropper or competitor, could analyse the vehicle's shared GPS stream to detect spoofing events (which may reveal proprietary testing regimes or vulnerabilities). By obfuscating the raw GPS signals via f_ϕ and shaping the agent's observable trajectory, PALADIN maintains that any classifier trained on the released traces achieves only chance-level accuracy at distinguishing normal from attacked windows, even as the vehicle continues to navigate efficiently. We model our environment as a finite Markov Decision Process (MDP) (– see Section 3) and restate here that our goal is to learn policy parameters θ and transformation parameters ϕ that jointly maximise cumulative reward while bounding information leakage (–see Equation (1)). The reward function encourages efficient displacement while allowing privacy shaping via wrappers: $r_t = \|\mathbf{s}_t - \mathbf{s}_{t-1}\|_2$.

Adversarial Leakage Estimation: To quantify behavioural privacy leakage, we train an auxiliary adversary h_ϕ to classify trajectories as normal or attacked. Our framework supports **multiple adversary architectures** to avoid overfitting conclusions to a single model class:

- **MLP Adversary.** A two-layer multilayer perceptron with hidden size 256, operating on flattened trajectory vectors. This baseline provides a strong but memoryless classifier.
- **GRU Adversary.** A Gated Recurrent Unit (GRU) network that ingests trajectory sequences step by step, with hidden dimensions of $d_h = 128$. The GRU captures temporal dependencies and dynamics across the 100-step window. The final hidden state is mapped to class logits via a linear layer.
- **Transformer Adversary.** A Transformer encoder with $L = 2$ layers and $H = 4$ attention heads, preceded by a linear embedding of the per-step features. Sequence representations are aggregated by mean pooling across time and projected to class logits. This model captures long-range dependencies and non-local correlations in the trajectory.

Formally, given a trajectory $\tau_{1:T}$, the adversary outputs logits $h_\phi(\tau_{1:T})$ and is trained with cross-entropy loss:

$$\mathcal{L}_{\text{priv}} = -\frac{1}{M}\sum_{i=1}^{M} \log p_\phi\Big(\widehat{y}^{(i)} = y^{(i)}_{\text{true}} \,\Big|\, \tau^{(i)}_{1:T}\Big)\,.$$

Parameters ϕ are optimised with ADAM (10^{-3} learning rate) after each rollout. During evaluation, we deliberately test the agent against *mismatched adversaries* (e.g. training with an MLP adversary but evaluating with a Transformer) to probe robustness to diverse leakage estimation strategies. This adversary-in-the-loop design yields a stringent and adaptive measure of behavioural privacy, rather than relying on a static or post-hoc classifier.

The environment is wrapped with a privacy-reward mechanism, at each step, the task reward is adjusted by a leakage penalty and a fidelity term: $r'_t = r_t - \lambda_t\,\ell_{\text{priv}} - \beta\,\text{MSE}(\mathbf{s}_t, \mathbf{s}_t^{\text{raw}})$, where ℓ_{priv} is either adversary confidence ($\max_k \text{softmax}$) or negative log-likelihood of the true class, and β controls fidelity to the original signal. The curriculum for λ_t follows staged increases, e.g.(0,0.0), (200,0.5), (400,1.0), (600,2.0), (800,4.0), allowing the agent to first learn the navigation task before progressively enforcing privacy. To evaluate PALADIN beyond autonomous navigation, we constructed analogous environments on financial domain as well, details on experiments are in Section E.3.

4.2 Baselines

We compare PALADIN against several baselines described below. Each baseline represents a distinct privacy–utility paradigm, allowing us to isolate the contribution of PALADIN.

Standard RL employs a vanilla RL agent that optimises only for task performance, with no mechanism to mitigate privacy leakage. Formally, the policy parameters θ are chosen to maximise the expected discounted return under the usual Markovian dynamics $P(s_{t+1} \mid s_t, a_t)$. By design, this agent attains an upper bound on utility but offers no protection against adversarial inference.

$$J(\theta) \;=\; \mathbb{E}_{\tau\sim\pi_\theta}\Big[\sum_{t=1}^{T}\gamma^{t-1} r(s_t, a_t)\Big], \tag{5}$$

Static Noise Injection injects Gaussian noise with fixed variance σ^2 into the observation and action channels. In **Static_Obs**, each raw observation is perturbed as $\tilde{x}_t = x_t + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \sigma^2 I)$, whereas in **Static_Act**, the executed action is noisy: $\tilde{a}_t = a_t + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \sigma^2 I)$. These methods are easy to implement and incur minimal overhead, but their non-adaptive nature often forces a trade-off: too much noise degrades utility severely, while too little fails to impede adversarial inference.

Adversarial Shaping without Curriculum (Adv_No_Cur) Here we integrate a surrogate adversary h into the reward but hold the privacy penalty λ fixed throughout training. At each time step, the shaped reward becomes $r_t^{\text{shaped}} = r(s_t, a_t) \;-\; \lambda\,\ell\big(h(\tilde{X}_{1:t}), y\big)$, where ℓ is the cross-entropy loss of the adversary's prediction of the sensitive label y. By co-training policy and adversary, this baseline

encourages privacy-aware behaviour, but the constant penalty can either hamper early task learning (if λ is large) or fail to enforce privacy (if λ is small).

DP-RL This baseline enforces formal (ε, δ)-differential privacy on the policy parameters using the DP-SGD mechanism of (Abadi et al. (2016)). For each minibatch of size L, we compute per-sample gradients $g_i = \nabla_\theta \mathcal{L}_i$, clip each to norm C, and aggregate with Gaussian noise:

$$\bar{g} = \frac{1}{L} \sum_{i=1}^{L} \text{clip}(g_i, C) \; + \; \mathcal{N}\big(0, \sigma^2 C^2 I\big). \tag{6}$$

This update guarantees (ε, δ)-DP, where $\varepsilon = f(\sigma, C, L, T)$ for T total steps. While Differentially Private-Reinforcement Learning (DP-RL) prevents model-inversion attacks on the trained parameters, it does not constrain the agent's observable behaviours: the actions and trajectories executed at test time may still reveal sensitive information. Consequently, highlighting the need for methods such as PALADIN that shape privacy in the observation space.

Differentially Private with NASH Equilibrium (DP-NASH) extends the DP-RL baseline, influenced by (Qiao & Wang (2024)), where we incorporate adversarially guided optimisation, forming a min-max training loop between the RL agent and a leakage-predicting adversary. The RL agent aims to maximise task performance, while the adversary is trained to infer sensitive attributes from the agent's behavioural traces. The combined optimisation objective is formalised as:

$$\min_{\theta} \max_{\phi} \; \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=1}^{T} \gamma^{t-1} \left(r(s_t, a_t) - \lambda \, \ell\big(h_\phi(\tilde{X}_{1:t}), y\big) \right) \right], \tag{7}$$

where θ denotes the policy parameters, ϕ denotes the adversary parameters, h_ϕ is the adversary network predicting sensitive label y based on trajectory prefix $\tilde{X}_{1:t}$, and ℓ is the leakage loss, typically cross-entropy. While this formulation suggests a joint min-max optimisation, in practice, we alternate training steps: the policy is optimised solely for task performance (standard PPO objective), and the adversary is trained separately to maximise leakage prediction accuracy using the latest trajectories. This setup enables indirect adversarial shaping: although the policy does not explicitly minimise leakage during updates, its behaviour is influenced over time by the adversary's evolving strength. To ensure formal privacy protection, we apply DP-SGD to both the policy and adversary models. Specifically, per-sample gradients are clipped and perturbed with Gaussian noise:

$$g_i^{\text{policy}} = \nabla_\theta \mathcal{L}_i^{\text{RL}}, \quad \bar{g}^{\text{policy}} = \frac{1}{L} \sum_{i=1}^{L} \text{clip}(g_i^{\text{policy}}, C) + \mathcal{N}(0, \sigma^2 C^2 I), \tag{8}$$

$$g_i^{\text{adv}} = \nabla_\phi \, \ell\big(h_\phi(\tilde{X}_{1:T}), y\big), \quad \bar{g}^{\text{adv}} = \frac{1}{L} \sum_{i=1}^{L} \text{clip}(g_i^{\text{adv}}, C) + \mathcal{N}(0, \sigma^2 C^2 I), \tag{9}$$

where L is the minibatch size, C is the gradient clipping threshold, and σ controls the noise magnitude. This dual-DP mechanism provides that both the agent's policy and the adversary's parameters satisfy (ε, δ), DP guarantees. Further discussion is detailed in Section C.

5 Experimental Result and Discussion

We evaluate PALADIN against baseline and privacy-aware RL approaches on the AV-GPS dataset. Our analysis focuses on the trade-off between utility (measured by return) and privacy leakage, where leakage is quantified using both adversary confidence (`leak_conf`) and negative log-likelihood (`leak_nll`). We report results across four representative train–test adversary pairings (MLP–GRU, MLP–Transformer, Transformer–GRU, GRU–Transformer). Table 1 summarises the comparative performance. For the MLP–GRU case, standard RL achieves moderate return (18.89) but with high leakage (`leak_conf`$= 0.89$). Static noise injection into observations or actions provides little benefit: leakage remains high (0.94 and 0.96 respectively), and utility is not consistently improved. Adversarial shaping without curriculum reduces leakage slightly (`leak_nll`=0.10) but does not match PALADIN. In contrast, PALADIN yields both the highest return (27.00) and the lowest leakage (`leak_nll`$= 0.056$), showing the effectiveness of delaying privacy constraints until

Table 1: Experimental Results on AV-GPS dataset. Multiple combination of adversaries for training and testing.

Train/Test Adversary	method	return	return_std	leak_nll	leak_nll_std	leak_conf	leak_conf_std
MLP-GRU	baseline	18.893	4.083	0.131	0.192	0.892	0.147
	static_obs	20.906	1.337	0.353	0.877	0.942	0.006
	static_act	16.397	0.283	0.364	0.968	0.959	0.002
	adv_no_cur	17.199	2.393	0.103	0.156	0.912	0.116
	proactive	27	0.227	0.056	0.001	0.946	0.001
	dp_rl	6.063	1.632	0.211	0.28	0.848	0.181
	dp_nash	6.791	1.666	0.206	0.265	0.843	0.186
MLP-Transformer	baseline	14.599	7.321	0.007	5.69E-05	0.993	5.64E-05
	static_obs	16.091	2.393	0.008	0.000171093	0.992	0.000169815
	static_act	17.278	0.369	0.007	8.39E-06	0.993	8.33E-06
	adv_no_cur	19.331	1.244	0.008	0.000126162	0.992	0.000125221
	proactive	20.146	0.172	0.495	1.462	0.992	7.99E-06
	dp_rl	4.627	1.357	0.007	5.73E-05	0.993	5.69E-05
	dp_nash	4.992	0.022	0.007	5.07E-05	0.993	5.04E-05
Transformwer-GRU	baseline	16.412	0.942	0.264	0.358	0.853	0.126
	static_obs	10.962	1.826	0.332	0.72	0.913	0.004
	static_act	17.888	0.355	0.333	0.751	0.921	0.003
	adv_no_cur	16.729	2.502	0.22	0.378	0.893	0.051
	proactive	20.484	0.105	0.08	0.001	0.923	0.000477351
	dp_rl	6.347	1.917	0.382	0.461	0.834	0.129
	dp_nash	7.151	1.999	0.386	0.461	0.842	0.118
GRU-Transformer	baseline	21.994	0.043	0.007	2.64E-07	0.993	2.63E-07
	static_obs	18.329	2.062	0.007	0.001	0.993	0.001
	static_act	20.159	4.221	0.007	0.001	0.993	0.001
	adv_no_cur	22.076	3.684	0.006	0.001	0.994	0.001
	proactive	22.796	2.534	0.006	0.001	0.994	0.001
	dp_rl	6.378	2.457	0.041	0.104	0.965	0.087
	dp_nash	4.025	0.851	0.006	0.001	0.994	0.001

after task mastery. Differential privacy baselines (DP-RL, DP-Nash) significantly reduce utility (returns < 7) while only partially mitigating leakage, highlighting the limitations of parameter-level guarantees in protecting behavioural signals. The MLP–Transformer pairing paints a more challenging picture: leakage confidence remains near 0.99 across all baselines, highlighting the adversary's strength. Nevertheless, PALADIN improves utility (20.14 vs 14.59 for baseline), while also inducing more uncertainty in adversary predictions, reflected in a higher variance of `leak_nll`. These results suggest that proactive shaping is able to soften otherwise deterministic leakage patterns, even if the adversary maintains high accuracy. For Transformer–GRU, PALADIN again outperforms all baselines, achieving return 20.48 with markedly reduced leakage (`leak_nll`$= 0.08$), while static noise and DP methods both underperform. Finally, for GRU–Transformer, PALADIN reaches the highest return (22.79) and maintains leakage near baseline levels. Here the adversary is particularly strong (`leak_conf`≈ 0.99), but PALADIN achieves improvements in utility without amplifying leakage. Therefore, these results demonstrate that PALADIN consistently dominates static noise and DP methods across adversary configurations.

Effect of λ: To better understand the role of the privacy penalty schedule, we conducted a λ-sweep for each adversary configuration, presented in Table 3 (–see Section E.1). Results show a non-monotonic relationship: for MLP–GRU, small positive values ($\lambda \in [0.1, 0.5]$) yield the most favourable balance, raising returns (up to 16.69 at $\lambda = 0.2$) while keeping leakage moderate. Too strong penalties ($\lambda \geq 1.0$) destabilise learning, with performance collapsing at $\lambda = 2.0$ (return 9.55). In contrast, the MLP–Transformer case shows that λ has little effect on leakage confidence, which remains pinned near 0.99. Here, utility peaks at $\lambda = 0.1$ (19.70) but declines as penalties increase. For Transformer–GRU, PALADIN exhibits resilience: while $\lambda = 0.2$ yields unstable training (return 10.84), moderate to high penalties ($\lambda = 1.0$ and 2.0) recover strong returns (20.78 and 22.55) and reduce leakage compared to baseline. The GRU–Transformer sweep highlights a similar trend: utility improves substantially at $\lambda = 0.1$ (18.70) with leakage reduction (`conf`$= 0.87$ vs 0.92 at baseline), but high λ values again degrade performance. Overall, these sweeps demonstrate that PALADIN is robust across adversary settings but requires careful tuning of λ. Moderate schedules yield the best trade-offs, while extreme penalties either collapse utility or provide no additional privacy benefit. This supports our central claim that curriculum-guided, adaptive enforcement of privacy is essential for balancing utility and privacy in sequential decision-making.

Ablation Study: We performed ablation experiments across all four adversary pairings (– MLP–GRU, MLP–Transformer, Transformer–GRU, and GRU–Transformer) to understand the individual contri-

butions of PALADIN's core components. We isolated the contributions of the Proactive framework into three pillars: feature transformation, curriculum scheduling and adversary capacity. Table 4 summarises the results in Section E.2. Across configurations, the `No_Curriculum` variant consistently collapsed, with negative returns on all pairing. This highlights the importance of delaying privacy penalties: applying a strong leakage term from the outset destabilises optimisation and prevents the agent from learning useful task behaviour. By contrast, `No_Transform` agents maintained moderate returns (e.g., 16.7 for MLP–GRU, 17.3 for Transformer–GRU), but leakage metrics deteriorated. Without a learned representation layer, the adversary could more easily exploit structure in raw states, confirming that representation learning is essential for obfuscation. While reducing adversary capacity (`Small_Adv`) produced mixed results: for MLP–GRU, return fell sharply (12.9 vs. 27.0 in the proactive case) and leakage remained high (0.96 confidence). Similar patterns emerged for Transformer–GRU and GRU–Transformer, where weaker adversaries limited the pressure on the policy to hide information, yielding unstable privacy–utility trade-offs. This highlights that adversarial strength must be sufficient to provide meaningful gradients during co-training. On the other hand, the `Shallow_Phi` setting (single-layer feature extractor) sometimes yielded high returns (22.8 for MLP–Transformer, 22.3 for Transformer–GRU) but consistently poor privacy, with elevated leakage (0.94 and 0.86 confidence, respectively). This suggests that depth in the feature transformation is key: shallow mappings lack the expressiveness to systematically distort sensitive features while preserving task utility. Finally, differentially private baselines (DP-RLand DP-NASH) showed limited effectiveness in this setting. Both returned low utility (5–9 across adversary pairings) while still leaking information at moderate levels (confidence ≥ 0.86). This reaffirms our central claim: parameter-level DP mechanisms do not directly mitigate behavioural leakage, and are therefore outperformed by PALADIN's proactive shaping. Therefore, these ablations validate that PALADIN's effectiveness arises from the combination of (i) curriculum-scheduled penalties, (ii) expressive representation learning, and (iii) sufficiently strong adversarial co-training. Removing any one of these elements either collapses training or re-exposes the agent to leakage, whereas their integration yields the strong privacy–utility frontier observed in the core results.

6 Conclusion and Future Work

We introduced PALADIN, a proactive privacy–shaping framework that embeds an adversarial leakage estimator directly into the RL reward loop. By co-training a surrogate adversary alongside the policy under a curriculum of penalty weights, PALADIN consistently achieves high task returns while reducing adversarial inference success. Experiments across autonomous vehicle GPS control, financial trading, and driver-identification telemetry show that PALADIN outperforms static noise injection and differential-privacy baselines, providing a general and hyperparameter-light recipe for balancing privacy and utility. Ablation studies further confirm the indispensability of both curriculum scheduling and representation learning, as omitting either leads to unstable training or weakened privacy. PALADIN does not provide worst-case (ϵ, δ)–DP guarantees; instead, it enforces strong *behavioural* privacy empirically. As immediate future work, we aim to integrate PALADIN with DP mechanisms to jointly protect training data and behavioural emissions, and to pursue formal guarantees via mutual information bounds. Broader extensions include *multi-agent* settings where inter-agent communication introduces novel leakage, *adaptive adversaries* (e.g., recurrent or attention-based) that dynamically adjust penalties, and scaling to *high-dimensional sensory inputs* such as vision or LiDAR. Finally, real-world deployment on physical platforms and edge devices will be crucial to assess practicability under latency and energy constraints. By elevating privacy to a first-class training signal rather than a post-hoc add-on, PALADIN provides a principled mechanism to negotiate the privacy–utility trade-off, laying the groundwork for deploying RL agents in adversarial real-world environments.

Ethics Statement

This work does not involve human or animal subjects, nor does it use sensitive personal data. All datasets are publicly available and anonymised (autonomous vehicle GPS traces (AV-GPS), financial trading time-series (NYSE)). The proposed framework is designed to mitigate privacy leakage rather than create new risks, and we discuss limitations (e.g., lack of formal (ϵ, δ)–DP guarantees) as it provides empirical guarantees transparently. We identify no conflicts of interest or ethical concerns

beyond those addressed. We make use of a Large Language Model-based tool for identifying related literature for the work.

Reproducibility Statement

We provide detailed descriptions of algorithms, architectures (MLP, GRU, Transformer), hyperparameters, and evaluation metrics in Section 4 and the Appendix Sections D and E. Dataset preprocessing steps are documented Sections (4 and D), and pseudocode is included (Algorithm 1). To support replication, we will release source code and configuration files upon acceptance.

References

Martin Abadi et al. Deep learning with differential privacy. *Proceedings of the 2016 ACM SIGSAC Conference*, 2016.

Murad Mehrab Abrar, Raian Islam, Shalaka Satam, Sicong Shao, Salim Hariri, and Pratik Satam. GPS-IDS: An anomaly-based GPS spoofing attack detection framework for autonomous vehicles. May 2024.

Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. In *2018 Information Theory and Applications Workshop (ITA)*, pp. 1–9, 2018. doi: 10.1109/ITA.2018.8503149.

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep variational information bottleneck. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=HyxQzBceg.

Yoshinori Aono et al. Privacy-preserving deep learning: Revisited and enhanced. *ACM Transactions on Privacy and Security*, 2017.

Borja Balle, Maziar Gomrokchi, and Doina Precup. Differentially private policy evaluation. In Maria Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 2130–2138, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/balle16.html.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models, 2021. URL https://arxiv.org/abs/2012.07805.

Cynthia Dwork et al. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*, 2006.

Dominik Gawlik. New york stock exchange. https://www.kaggle.com/datasets/dgawlik/nyse?select=prices.csv, February 2017.

Adam Gleave, Michael Dennis, Cody Wild, Neel Kant, Sergey Levine, and Stuart Russell. Adversarial policies: Attacking deep reinforcement learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJgEMpVFwB.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pp. 2672–2680, Cambridge, MA, USA, 2014a. MIT Press.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2014b.

J. B. Heaton, N. G. Polson, and J. H. Witte. Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(1):3–12, 2017. doi: https://doi.org/10.1002/asmb.2209. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/asmb.2209.

Benjamin Hilprecht, Martin Härterich, and Daniel Bernau. Monte carlo and reconstruction membership inference attacks against generative models. volume 2019, 07 2019. doi: 10.2478/popets-2019-0067.

Sham M. Kakade. On the sample complexity of reinforcement learning. 2003. URL https://api.semanticscholar.org/CorpusID:260534783.

Bo Li, Wei Wang, and Peng Ye. The limits of differential privacy in online learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=Cqr6E81iB7.

Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T Dudley. Deep learning for healthcare: review, opportunities and challenges. *Brief. Bioinform.*, 19(6):1236–1246, November 2018.

Sayedeh Leila Noorbakhsh, Binghui Zhang, Yuan Hong, and Binghui Wang. Inf2guard: an information-theoretic framework for learning privacy-preserving representations against inference attacks. In *Proceedings of the 33rd USENIX Conference on Security Symposium*, SEC '24, USA, 2024. USENIX Association. ISBN 978-1-939133-44-1.

Xinlei Pan, Weiyao Wang, Xiaoshuai Zhang, Bo Li, Jinfeng Yi, and Dawn Song. How you act tells a lot: Privacy-leaking attack on deep reinforcement learning. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '19, pp. 368–376, Richland, SC, 2019. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450363099.

Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 2817–2826. JMLR.org, 2017.

Dan Qiao and Yu-Xiang Wang. Offline reinforcement learning with differential privacy, 2023. URL https://openreview.net/forum?id=NT51Ty0-Bfu.

Dan Qiao and Yu-Xiang Wang. Differentially private reinforcement learning with self-play. *ArXiv*, abs/2404.07559, 2024. URL https://api.semanticscholar.org/CorpusID:269042894.

Touqir Sajed and Or Sheffet. An optimal private stochastic-mab algorithm based on an optimal private stopping rule, 2019. URL https://arxiv.org/abs/1905.09383.

Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *ACM CCS*, 2015.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018. ISBN 0262039249.

Samuel Yang-Zhao and Kee Siong Ng. Privacy preserving reinforcement learning for population processes. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=zZFb1aDUeE.

Han Zhao, Jianfeng Chi, Yuan Tian, and Geoffrey Gordon. Trade-offs and guarantees of adversarial representation learning for information obfuscation. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

We present all the supplementary materials in the appendix.

A PALADIN PIPELINE

We present PALADIN Architecture in this section.

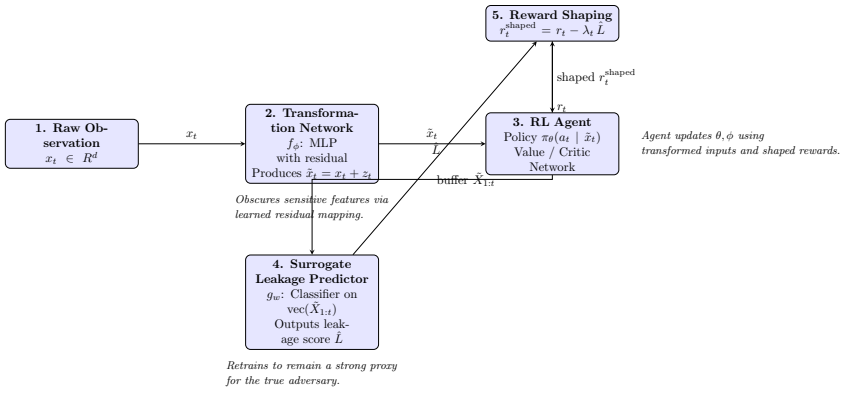

Figure 1: PALADIN Pipeline: observations pass through the transformation network f_ϕ, the surrogate predictor g_w estimates leakage, and the RL agent receives a penalised reward.

B DETAILED PROOF OF THEOREM

In this appendix, we give a more expansive, yet intuitive, derivation of the Privacy–Shield Bound from Theorem 1. Recall that at convergence, PALADIN optimises the shaped objective

$$J(\theta, \phi; w) \;=\; \mathbb{E}_{\tau\sim\pi_\theta}\Big[\sum_{t=1}^{T} r(s_t, a_t)\Big] \;-\; \lambda_K\, \mathbb{E}_{\tau\sim\pi_\theta}\big[\widehat{L}(w; \tilde{X}, y)\big], \tag{10}$$

where $\widehat{L}(w; \tilde{X}, y) \;=\; -\log p_w(y \mid \tilde{X})$ is our surrogate leakage loss ((cf. adversarial training in (Goodfellow et al. (2014b))). We discuss this in two steps below:

STEP 1: STATIONARITY OF THE JOINT OBJECTIVE

At convergence, PALADIN has jointly optimised the policy and transformation parameters (θ, ϕ) under a fixed adversary w (Sutton & Barto (2018)). Concretely, it maximises the shaped objective

$$J(\theta, \phi; w) = \underbrace{\mathbb{E}_{\tau\sim\pi_\theta}\Big[\sum_{t=1}^{T} r(s_t, a_t)\Big]}_{\mathbb{E}[R]} - \lambda_K \underbrace{\mathbb{E}_{\tau\sim\pi_\theta}\big[\widehat{L}(w; \tilde{X}, y)\big]}_{\mathbb{E}[\widehat{L}]},$$

where $\widehat{L}(w; \tilde{X}, y) \;=\; -\log p_w(y \mid \tilde{X})$ is our surrogate leakage loss and λ_K is the final privacy penalty.

Since (θ, ϕ) are (approximately) optimal, the gradient of J with respect to these parameters vanishes:

$$\nabla_{\theta,\phi}\, J(\theta, \phi; w) \;\approx\; 0.$$

Intuitively, this means that any infinitesimal change in the policy or transformation that would increase the raw reward $\mathbb{E}[R]$ must be offset by an equal or greater increase in the expected leakage term $\lambda_K\, \mathbb{E}[\widehat{L}]$, and vice versa.

Rearranging the two competing terms yields

$$\lambda_K\, \mathbb{E}[\widehat{L}] \;\approx\; \mathbb{E}[R] - C,$$

where C is a nonnegative slack term capturing residual suboptimality, arising from finite-step optimisation, approximation error (Kakade (2003)), and nonzero gradients in high-dimensional parameter spaces. Since each per-step reward satisfies $|r(s_t, a_t)| \leq r_{\max}$, the total raw return $R = \sum_{t=1}^{T} r(s_t, a_t)$ is bounded by $R_{\max} = T\, r_{\max}$. Thus, one may bound

$$0 \;\leq\; C \;\leq\; \epsilon\, R_{\max}, \quad \epsilon \geq 0,$$

where ϵ captures the fraction of the maximum return lost to optimisation slack.

Substituting this bound into the rearranged stationarity condition gives a lower bound on the expected surrogate leakage:

$$\mathbb{E}[\widehat{L}] \;\geq\; \frac{\mathbb{E}[R] \;-\; \epsilon\, R_{\max}}{\lambda_K}. \tag{11}$$

In words, at equilibrium, the privacy penalty λ_K must be large enough that the expected leakage cannot fall below the gap between the achievable task return and any optimisation slack, scaled by λ_K^{-1}. This relation formalises the trade-off enforced by PALADIN's shaped objective: improving privacy (reducing $\mathbb{E}[\widehat{L}]$) necessarily incurs a cost in task return $\mathbb{E}[R]$, and vice versa.

Step 2: From Surrogate to True Leakage

Our analysis thus far bounds the *expected surrogate loss* $\mathbb{E}[\widehat{L}]$. To translate this into a guarantee on the *true adversary loss* $\ell(h(\tilde{X}), y)$, we invoke the unbiasedness and concentration properties of our surrogate estimator. By construction,

$$\mathbb{E}[\widehat{L}] = \mathbb{E}\big[\ell(h(\tilde{X}), y)\big],$$

and we assume a finite estimator variance $\mathrm{Var}[\widehat{L}] = \sigma^2$.

Applying Chebyshev's inequality, for any $\delta \in (0, 1)$,

$$\Pr\big[\, |\widehat{L} - \mathbb{E}[\widehat{L}]| \geq \sigma/\sqrt{\delta}\big] \;\leq\; \delta.$$

In other words, with probability at least $1 - \delta$, the realised surrogate loss lies within $\sigma/\sqrt{\delta}$ of its expectation. Equivalently, in expectation, one obtains

$$\mathbb{E}\big[\ell(h(\tilde{X}), y)\big] \;\geq\; \mathbb{E}[\widehat{L}] - \sigma.$$

Substituting the lower bound from equation 11 immediately yields

$$\mathbb{E}\big[\ell(h(\tilde{X}), y)\big] \;\geq\; \frac{\mathbb{E}[R] - \epsilon\, R_{\max}}{\lambda_K} \;-\; \sigma,$$

completing the derivation of the Privacy–Shield Bound in Theorem 1. This final inequality makes explicit the trade-off: achieving a high expected return $\mathbb{E}[R]$ under a given penalty λ_K necessarily enforces a lower bound on the adversary's expected inference loss, up to the surrogate variance σ.

B.1 Discussion of Key Assumptions

Before applying the Privacy–Shield Bound, we define several standard but crucial assumptions:

- **Unbiased Surrogate.** We assume that the learned adversary g_w provides an unbiased estimate of true leakage $\ell(h(\tilde{X}), y)$. In practice, continuously retraining g_w on fresh trajectories helps maintain this property.
- **Bounded Rewards.** Each per-step reward satisfies $|r(s_t, a_t)| \leq r_{\max}$, so the total return $R = \sum_{t=1}^{T} r(s_t, a_t)$ is finite. This is a standard requirement in finite-horizon RL analyses.
- **Convergence Slack (ϵ).** Real-world optimisers only approximate stationarity. We introduce $\epsilon \geq 0$ to capture residual gradient magnitude or suboptimality at convergence. Empirically, ϵ is small once learning stabilises.
- **Concentration Inequality.** We apply Chebyshev's inequality to bound the deviation of $\widehat{L}$ from its mean. If one can assume sub-Gaussian tails for the surrogate loss, tighter high-probability guarantees follow from Bernstein bounds at the expense of stronger moment conditions.

With these assumptions, the bound is prepared

$$\mathbb{E}[\ell(h(\tilde{X}), y)] \;\geq\; \frac{\mathbb{E}[R] - \epsilon\, R_{\max}}{\lambda_K} \;-\; \sigma.$$

In particular, to enforce a minimum adversarial loss δ, one selects

$$\lambda_K \;\geq\; \frac{\mathbb{E}[R] - \epsilon\, R_{\max}}{\delta + \sigma},$$

which clearly trades off utility ($\mathbb{E}[R]$) against privacy (controlled by δ and σ).

To reiterate on the bound, we emphasise that this bound is not a formal (ε, δ)-DP guarantee. It relies on the assumption that the surrogate adversary provides an unbiased estimate of leakage, which may not strictly hold in practice if the adversary underfits or if the data distribution shifts. Hence, the bound should be interpreted as a heuristic shaping signal that guides optimisation, rather than a worst-case guarantee of privacy.

C Extended discussion on DP-NASH

DP-NASH operationalises privacy defence via two intertwined mechanisms: (i) formal DP guarantees at the parameter level, protecting against extraction or inversion attacks on model weights, and (ii) adversarial shaping, which pressures the policy to evolve privacy-preserving behaviours indirectly through buffer dynamics and adversarial retraining. This dynamic creates an arms-race-like environment, where the adversary continuously adapts to exploit emerging leakage patterns, and the policy gradually internalises strategies that reduce its observability footprint.

While DP-NASH is a powerful baseline, merging DP with adversarial privacy dynamics, it has practical trade-offs. The privacy penalty λ is fixed throughout training, which may under- or over-penalise depending on the agent's learning stage. Additionally, as the adversary strengthens, training can become unstable, particularly when privacy noise is high, requiring careful tuning of hyperparameters to balance privacy, utility, and learning stability.

As mentioned in Section 4.2, DP-NASH is influenced by the (Qiao & Wang (2024)) framework, which implements Nash value iteration and equilibrium computation in a multi-agent Markov game setting. However, DP-NASH is designed specifically for single-agent RL with behavioural privacy concerns. Therefore, rather than reformulating the problem as a game between environment agents, it employs a dedicated leakage-predicting adversary within a standard RL pipeline, allowing us to emulate adversarial privacy dynamics in a single-agent context. This makes DP-NASH a rigorous and technically meaningful baseline to benchmark PALADIN's performance, while remaining fully aligned with the scope of behavioural privacy in standard RL tasks.

Privacy Guarantee Sketch for DP-NASH DP-NASH applies DP-SGD independently to both the policy network and the adversarial leakage predictor. Each model satisfies (ε, δ)-DP as guaranteed by the moments accountant (Abadi et al. (2016)). Specifically, for each model, the privacy guarantee is:

$$\varepsilon = f(\sigma, C, L, T),$$

where σ is the noise multiplier, C is the clipping norm, L is the minibatch size, and T is the total number of steps. Because both the policy and adversary access the same sensitive data (trajectories), standard composition of DP applies. By the basic composition theorem, the overall privacy budget satisfies

$$\varepsilon_{\text{total}} \leq \varepsilon_{\text{policy}} + \varepsilon_{\text{adv}}$$

for a common δ. This guarantees that any observer of both the policy and adversary models learns at most $(\varepsilon_{\text{total}}, \delta)$-differentially private information about any individual trajectory.

Importantly, the alternating training schedule, where the policy and adversary are updated in different phases, does not break DP because the noise injection and privacy accounting are applied at each step independently, and DP is preserved under post-processing. Thus, DP-NASH's privacy guarantees hold across the full training process.

Table 2: AV-GPS-Dataset features detail obtained from (Abrar et al. (2024))

Name	Description
Roll, Pitch	Rotation around the front-to-back axis of the vehicle is called Roll, and rotation around the side-to-side axis is called Pitch. Measured in degrees, values range from -180° to 180°.
Heading	The angular direction of the compass towards which the vehicle's head is pointed, measured in degrees and ranging from 0° to 360°.
Yaw, Yaw Rate	Rotation around the vertical axis relative to North, measured in degrees (-180° to 180°). Yaw Rate is the rate of change of Yaw angle with time, measured in degrees/second.
Velocity	Actual speed of the vehicle on the ground, is measured in meters/second.
Steering Angle	Degree to which the front wheel axle is turned during autonomous navigation, measured in degrees.
Relative Altitude	Altitude of the vehicle relative to the initial starting or home location, measured in meters.
Altitude AMSL	Altitude above Mean Sea Level (AMSL), with Tucson, Arizona's elevation at approximately 728 meters (2,389 feet).
Altitude Tuning, Setpoint	Altitude Tuning converts altitude error to a required climb/descent rate. Setpoints are feed-forward values to altitude controllers, added to outputs and fed back as inputs, primarily for UAVs.
X-Track Error	Cross-track error indicating the deviation of the vehicle from its desired track. A gain is applied to converge it to 0.
Travelled Distance	Total distance covered by the vehicle after , measured in meters.
Run Time	Time record of vehicle's flight or runtime after initialisation, recorded in hours: minutes: seconds format.
Distance To Home	Distance to be travelled by the vehicle to reach its home location, measured in meters.
Mission Index	Denotes the current mission or waypoint during autonomous mode, with mission planning done beforehand.
Heading To Next WP	New heading angle to the next waypoint in the mission, measured in degrees and ranging from 0° to 360°.
Heading To Home	Heading angle to the home location relative to the current heading, measured in degrees and ranging from 0° to 360°.
Distance To GCS	Distance from the vehicle to the Ground Control Station, measured in meters.
Throttle	Percentage of throttle applied by the autopilot to the vehicle's throttle motor, measured in %.
Hobbs	Record of "hobbs time," measuring the actual operational time of the vehicle.
Clock Time, Clock Date	Current clock time and date.
GPS Latitude, Longitude	Current GPS latitude and longitude of the vehicle.
GPS MGRS	GPS Military Grid Reference System (MGRS) coordinates for military locating.
GPS HDOP, GPS VDOP	Horizontal and vertical Dilution of Precision for GPS signals, indicating positional accuracy. HDOP typically ranges 1-2, VDOP typically 2-4.
GPS Course	GPS heading angle or actual direction relative to North, calculated from GPS measures, ranging from 0° to 360°.
Satellite Locks, Count	Satellite Lock is the number of satellites the vehicle is connected to, while Satellite Count is the total number available.
Longitudinal, Lateral, Vertical Position	3D coordinates of the vehicle relative to its starting point, measured in meters.
Longitudinal, Lateral, Vertical Velocity	3D coordinate velocities relative to initial velocities, measured in meters/second.
Absolute Longitudinal, Lateral Velocity	Absolute values of longitudinal and lateral velocities, measured in meters/second.
Temperature	Local temperature of Pixhawk autopilot hardware, measured in Fahrenheit.
Longitudinal, Lateral, Vertical Vibration	Vibration levels on the X, Y, and Z axes of the vehicle, measured in Hertz (cycles per second).
Data Type	Indicates the type of data: normal data (0) or GPS spoofing attack data (1).

D Further Dataset Discussion

This section provides further details on the benchmark datasets..

D.1 AV-GPS Dataset- Feature details

Table 2 presents the details of features that are available in the AV-GPS dataset (Abrar et al. (2024)).

D.2 Financial Trading Benchmark Dataset

We utilised our historical price–volume data from the NYSE dataset on Kaggle (Gawlik (2017)), comprising 851,264 daily records for 501 tickers over the period 4 January 2010 to 30 December 2016. Each record consists of five min–max normalised features: {open, high, low, close, volume}, yielding a 5-dimensional observation per day.

A brief analytic overview of this dataset reveals:

- **Coverage and Granularity**: 501 unique ticker symbols, each with a full time-series length of 1,762 trading days, allowing non-overlapping windows of $T = 100$ days per episode.
- **Price Statistics**: The per-day open/close prices have a mean of approximately 52.8 (std. 83.7), with $25^{th} - 75^{th}$ percentile range [33.8,79.9], reflecting broad cross-sectional variability in stock valuations.
- **Volume Dynamics**: Trading volume averages 5.4 million shares per day (std. 12.5 million), with a long-tailed distribution reaching up to 0.86 billion shares on high-liquidity days.
- **Top Tickers**: The five most frequent symbols, KSU, NOC, ORCL, OMC and OKE, each contribute 1,762 days of data, ensuring uniform sequence lengths across episodes.

For our Financial trading benchmark, we select the ten most liquid symbols (by total record count) and segment each series into non-overlapping windows of $T = 100$ consecutive days. Each window's observation sequence is min–max normalised to $[0, 1]^5$, and the per-step reward r_t is defined as the trading volume at day t. The sensitive label y is the integer-encoded ticker symbol for that window. We reserve 80% of windows for training the adversary and 20% for validation. This dataset's large scale, cross-sectional diversity and real-world noise characteristics make it an ideal testbed for validating PALADIN's ability to preserve utility while mitigating adversarial inference in financial time-series settings.

D.2.1 Experimental Setup - Financial Trading Dataset

All components of PALADIN and its baselines are implemented in a unified Python codebase built on Gymnasium and Stable Baselines3 (with Opacus for DP when needed). At its core, we wrap fixed-length windows ($T = 100$) of pre-normalised OHLCV (open, high, low, close, volume) sequences in a custom Gym environment that returns 5-dimensional observations and a dummy $\mathbb{R}^5$ action space. Two simple wrappers inject static noise into observations or actions, while our privacy-reward wrapper buffers the last T steps, queries a jointly trained Multi-Layer Perceptron (MLP) adversary (with three 256-unit ReLU layers with dropout) for its max-softmax leakage score, and subtracts λ_t-scaled leakage from the raw reward. The transformation network f_ϕ is realised as a two-layer MLP feature extractor (64-dim output) with optional Gaussian bottleneck noise. A custom callback synchronises a curriculum schedule $\{(t_i, \lambda_i)\}$, updating λ_t at rollout start and retraining the leakage predictor at rollout end. We experiments in three phases, core baselines, λ-sweep, and ablations for rigorous evaluation.

D.2.2 Financial Trading Benchmark Dataset

Our Financial trading benchmark uses daily OHLCV (open, high, low, close, volume) data from the NYSE "prices.csv" dataset on Kaggle (Gawlik (2017)). We preprocess and instantiate it as follows:

1. **Ticker selection.** We sort by symbol and date, then factorise tickers into integer labels. We retain the top $K = 10$ most frequent symbols, each with at least 100 trading days,to ensure sufficient episodes per class.
2. **Episode construction.** For each selected symbol, we slide non-overlapping windows of length $T = 100$ days over its OHLCV time series. Each window becomes one episode, labelled by the symbol's integer code y.
3. **Normalisation.** We stack all windows into an $(N, 100, 5)$ array and apply feature-wise min–max scaling so that each of open, high, low, close and volume lies in $[0, 1]$.
4. **Train–test split.** We stratify-split the N episodes 80/20 into training and validation sets for adversary and RL.
5. **MDP specification.** We wrap the windows in a custom Gym environment with:
   - $\mathcal{S} = [0, 1]^5$: each state is the normalised OHLCV vector.
   - $\mathcal{A} = [-1, 1]^5$: a dummy continuous action space (actions do not affect transitions).
   - P: deterministic replay of the historical window.
   - $r_t = \text{volume}_t$: per-step reward equals the (scaled) trading volume at day t.
   - Sensitive label $y \in \{0, \ldots, 4\}$ is the encoded ticker for that window.

This yields $N = \sum_{i=1}^{5} \lfloor n_i/100 \rfloor$ episodes, approximately $5 \times 17 = 85$ total, capturing diverse trading behaviours. We then apply the adversarial-in-the-loop wrappers, which buffer the last T observations, querying a jointly trained MLP leakage predictor, and subtracting λ_t times its max-softmax score from the raw volume reward at each step.

The adversary's objective is to infer that sensitive information purely from the observable behaviour, where PALADIN is designed to prevent this inference while preserving the agent's primary task performance. For example, an adversary, such as a front-running trading firm, might observe the (transformed) price–volume sequence emitted by an automated trading agent and infer which stock it is trading, thereby revealing strategic intent or proprietary positioning. PALADIN intervenes by perturbing the feature stream and penalising trajectories that leak ticker identity, so that the agent's reward maximisation remains focused on high trading volume while the adversary's classification accuracy is driven back to random guess levels.

E Further Evaluation Results

This section presents further evaluation setup, dataset and results. Section E.1 presents the λ-sweep table for understanding the role of privacy penalty schedule. Section E.2 presents the ablation study table to better understand the individual contributions of PALADIN's core components.

E.1 λ sweep table for AV-GPS dataset

Here, we present our λ sweep table for AV-GPS dataset.

Table 3: λ sweep for the PALADIN on AV-GPS dataset

	λ	return	return_std	leak_conf	leak_conf_std	leak_nll	leak_nll_std
MLP-GRU	0.0	10.667	0.101	0.962	0	0.038	0
	0.1	13.762	2.81	0.926	0.096	0.083	0.122
	0.2	16.687	0.253	0.875	0.18	0.174	0.276
	0.5	12.999	0.437	0.918	0.134	0.107	0.206
	1.0	16.365	3.12	0.884	0.134	0.405	0.799
	2.0	9.553	2.78	0.919	0.123	0.096	0.167
MLP-Transformer	0.0	17.42	0.818	0.992	0.001	0.008	0.001
	0.1	19.698	0.157	0.991	0	0.009	0
	0.2	16.984	0.137	0.992	0	0.008	0
	0.5	11.553	4.373	0.992	0.001	0.008	0.001
	1.0	15.172	2.794	0.992	0	0.008	0
	2.0	16.31	0.594	0.992	0	0.008	0
Transformer-GRU	0.0	18.371	0.181	0.879	0.001	0.129	0.002
	0.1	19.947	0.457	0.877	0.072	0.202	0.292
	0.2	10.84	0.616	0.865	0.088	0.31	0.429
	0.5	13.958	0.185	0.86	0.076	0.21	0.264
	1.0	20.785	2.008	0.855	0.107	0.484	0.666
	2.0	22.551	2.088	0.881	0.064	0.209	0.319
GRU-Transformer	0.0	13.349	4.851	0.919	0.05	0.086	0.054
	0.1	18.697	0.203	0.872	0.042	0.139	0.045
	0.2	17.914	2.111	0.921	0.038	0.083	0.04
	0.5	16.399	1.485	0.921	0.038	0.083	0.04
	1.0	18.418	2.703	0.892	0.035	0.115	0.037
	2.0	13.416	1.269	0.93	0.043	0.074	0.046

E.2 Ablation study on AV-GPS Dataset

We present an ablation study table (Table 4) for the AV-GPS dataset.

E.3 Experimental setup on Financial Trading (NYSE) Dataset

To evaluate PALADIN beyond autonomous navigation, we constructed analogous environments on high-frequency trading data drawn from the NYSE daily prices dataset (Gawlik (2017)). These include: (i) **prices**, where input features are five daily attributes [open, high, low, close, volume] of the top-5 traded symbols; and (ii) **credit**, where input features are anonymised transaction variables from the credit-card fraud dataset. Each 100-step window (T=100) is labelled with either the traded

Table 4: Ablation study on AV-GPS Dataset

Train-Test Adv	method	return	return_std	leak_conf	leak_conf_std	leak_nll	leak_nll_std
MLP-GRU	no_transform	16.742	0.017	0.963	0	0.038	0
	no_curriculum	-335.867	492.019	0.87	0.181	0.432	0.8
	small_adv	12.931	0.162	0.958	0	0.043	0
	shallow_phi	12.203	2.951	0.841	0.185	0.202	0.25
	weak_adv	18.609	0.454	0.917	0.111	0.096	0.147
	dp_rl	7.627	0.005	0.919	0.137	0.101	0.193
	dp_nash	7.369	0.633	0.876	0.169	0.156	0.232
MLP-Transformer	no_transform	15.039	9.006	0.971	0.063	0.032	0.072
	no_curriculum	-232.221	451.214	0.992	0	0.008	0
	small_adv	13.678	0.159	0.992	0	0.484	1.429
	shallow_phi	22.809	1.858	0.94	0.107	0.07	0.126
	weak_adv	16.996	0.539	0.992	0	0.008	0
	dp_rl	6.272	0.585	0.992	0.001	0.008	0.001
	dp_nash	6.866	2.907	0.991	0.003	0.009	0.003
Transformer-GRU	no_transform	17.324	0.006	0.904	0	0.101	0
	no_curriculum	-1015.973	1587.826	0.862	0.092	0.522	0.717
	small_adv	12.289	0.163	0.913	0.002	0.091	0.002
	shallow_phi	22.251	9.948	0.858	0.076	0.473	0.574
	weak_adv	16.358	0.714	0.866	0.077	0.148	0.102
	dp_rl	5.759	0.535	0.878	0.072	0.201	0.294
	dp_nash	6.496	0.575	0.86	0.089	0.309	0.418
GRU-Transformer	no_transform	10.891	5.028	0.913	0.042	0.092	0.044
	no_curriculum	-277.009	20.34	0.824	0.101	0.213	0.176
	small_adv	17.241	4.038	0.886	0.037	0.122	0.039
	shallow_phi	16.176	0.949	0.895	0.034	0.112	0.036
	weak_adv	17.504	1.491	0.889	0.036	0.314	0.593
	dp_rl	5.848	1.12	0.914	0.041	0.091	0.044
	dp_nash	9.29	1.578	0.867	0.12	0.163	0.192

symbol (prices) or the fraud class (credit). In each case, we adversarially train policies such that an external observer cannot infer sensitive attributes (e.g., the traded asset identity or whether a sequence contains fraud) from the shared trajectories.

Environment. We define an MDP where the state at time t is the feature vector $\mathbf{s}_t \in [0,1]^n$, with $n = 5$ for `prices` and $n = 29$ for `credit`. Actions $a_t \in [-1,1]^n$ perturb the observed sequence, and the reward encourages smooth evolution of the observed features while penalising excessive distortion:

$$r_t = \|\mathbf{s}_t - \mathbf{s}_{t-1}\|_2 - 0.01\|a_t\|_2^2.$$

The original unperturbed vector $\mathbf{s}_t^{\text{raw}}$ is preserved to compute fidelity losses and leakage penalties.

Adversarial Leakage Estimation. Behavioural privacy leakage is estimated by adversaries trained to classify trajectories into their sensitive labels. As in the GPS setting, we instantiate three architectures:

- **MLP**: a two-layer perceptron with hidden size 256 on flattened trajectories.
- **GRU**: a recurrent model with one GRU layer, hidden dimension 128 by default, mapping the final hidden state to logits.
- **Transformer**: a Transformer encoder with $L = 2$ layers, $H = 4$ attention heads, and hidden size 128 by default. Stepwise features are linearly embedded, processed with self-attention, aggregated by mean pooling, and classified.

Adversaries are optimised by cross-entropy loss

$$\mathcal{L}_{\text{priv}} = -\tfrac{1}{M}\sum_{i=1}^{M} \log p_\phi\left(\widehat{y}^{(i)} = y_{\text{true}}^{(i)} \,\middle|\, \tau_{1:T}^{(i)}\right),$$

using ADAM with learning rate 10^{-3}. For robustness, we also evaluate agents against mismatched adversaries (e.g. training with an MLP adversary but evaluating with a Transformer).

Privacy–Reward Integration and Curriculum. We wrap the environment with a privacy-reward mechanism. At each step, the shaped reward is

$$r'_t = r_t - \lambda_t\, \ell_{\text{priv}}(\tau_{1:t}) - \beta\, \text{MSE}(\mathbf{s}_t, \mathbf{s}_t^{\text{raw}}),$$

where ℓ_{priv} is the negative log-likelihood of the true class, and β controls fidelity. λ_t follows a staged curriculum, e.g. $(0, 0.0), (200, 0.5), (400, 1.0), (600, 2.0), (800, 4.0)$, ensuring that the agent first learns task dynamics before strong privacy penalties are enforced.

Evaluation. For each dataset, we train a separate "true attacker" adversary (with tunable sequence length, hidden size, and epochs) to evaluate leakage. We report: mean return and variance, adversary confidence and negative log-likelihood, and classification accuracy/F1. Baselines mirror those in Section 4.2: unconstrained RL, static noise (obs/act), adversarial shaping without curriculum, and two DP variants (`dp_rl`, `dp_nash`). We also conduct λ-sweeps and ablations (no transform, no curriculum, shallow ϕ, small adversary, weak adversary) to isolate the contributions of each PALADIN component.

E.4 Evaluation results for the Financial Trading (NYSE) Dataset

Table 5 reports the comparative performance across adversary configurations.

For the MLP–GRU setting, the baseline achieves a moderate return (16.80) with high leakage (`leak_conf`$= 0.88$). Adding static noise to observations or actions marginally alters leakage (remaining > 0.92) while reducing returns to 11.65–14.68. Adversarial training without curriculum (`adv_no_cur`) achieves slightly lower leakage (`leak_nll`$= 0.44$) but still suffers from unstable performance (return 12.93). In contrast, PALADIN substantially improves both return and privacy: it reaches the highest utility (21.85) while reducing leakage to near-baseline levels (`leak_nll`$= 0.16$). Differential privacy methods (DP-RL, DP-Nash) collapse task returns (10.64 and 6.87) and fail to suppress leakage confidence (0.85–0.89), illustrating their inadequacy for behavioural protection.

The MLP–Transformer configuration is the most challenging: leakage confidence is consistently extreme (≥ 0.97) for all baselines. Still, PALADIN outperforms others in utility (24.77 vs. 16.74 baseline) while modestly softening leakage (`leak_nll`$= 0.05$). Static noise and adversarial shaping without a curriculum are ineffective: returns remain 15.09–19.25 with high leakage. Again, DP-based methods collapse (returns 6.09–6.94) and offer only marginal privacy gains.

For the Transformer–GRU pairing, PALADIN yields clear gains: return increases to 20.48 (from baseline 16.41), while leakage decreases to `leak_nll`$= 0.09$. Noise-based approaches perform inconsistently, with either utility loss (static_obs, 10.96) or weak leakage suppression (static_act, `leak_conf`$= 0.91$). DP methods again fail, producing low returns (6–7) and high leakage variance. Notably, proactive shaping even outperforms strong baselines with much lower leakage standard deviation.

Finally, in the GRU-Transformer case, the adversary is strongest: leakage confidence remains ≈ 0.98 across all methods, underscoring near-deterministic leakage. Nonetheless, PALADIN achieves the best utility (19.56), surpassing both baseline (17.77) and noise-based methods, while maintaining leakage comparable to baseline. Interestingly, the `adv_no_cur` variant achieves slightly higher return (18.01) than static methods but fails to reduce leakage. DP baselines collapse again (returns 3.71–6.49) without real leakage protection.

Overall, these results demonstrate that PALADIN consistently dominates static noise and DP-based methods across all adversary pairings. Its strongest gains occur against GRU-based adversaries, where curriculum-guided penalties provide both high utility and tangible leakage suppression. In more adversarially powerful configurations (e.g., Transformer predictors), PALADIN still delivers the best privacy–utility frontier, raising returns without exacerbating leakage. This supports our claim that behavioural leakage cannot be reliably mitigated by parameter-level DP, but requires dynamic, curriculum-scheduled adversarial shaping.

Effect of λ sweep We further investigated the impact of the privacy penalty weight λ by sweeping values from 0.0 to 2.0 across all four adversary pairings (Table 6).

For the MLP–GRU case, performance is highly sensitive to λ. Without penalties ($\lambda = 0.0$), PALADIN already achieves strong utility (20.77) with moderate leakage (`leak_nll`$= 0.09$). Small positive penalties, however, destabilise learning: returns collapse to 12.20–15.17 at $\lambda = 0.1$–0.2, accompanied by erratic leakage (`leak_nll`$= 0.30$ at $\lambda = 0.1$). Larger penalties recover some stability: $\lambda = 1.0$ yields a return of 18.31 with reduced leakage (`leak_nll`$= 0.07$), while overly strong penalties

($\lambda = 2.0$) again degrade utility (11.90). These results confirm that moderate scheduling ($\lambda \approx 1.0$) balances privacy and utility, while aggressive early enforcement hinders task mastery.

The MLP–Transformer setting demonstrates the resilience of PALADIN under strong adversaries. Returns peak at $\lambda = 0.1$ (21.01) with softened leakage (`leak_nll`$= 0.04$ vs. 0.013 at baseline). Larger λ values (≥ 0.5) reduce utility (13.94–19.55) while failing to substantially alter leakage confidence, which remains near-saturated (0.98–0.99). This suggests that when adversaries are particularly strong, PALADIN primarily benefits from mild penalties that encourage representational obfuscation without destabilising policy learning.

For the Transformer–GRU pairing, a more favourable pattern emerges. Small penalties ($\lambda = 0.1$) already improve returns (19.95 vs. 17.67 at $\lambda = 0.0$), though leakage remains moderate. Interestingly, increasing λ continues to enhance both utility and privacy: the best outcome occurs at $\lambda = 2.0$, where return reaches 22.55 (the highest observed across sweeps) with reduced leakage (`leak_nll`$= 0.20$). These results highlight PALADIN's robustness in this configuration, where stronger penalties directly reinforce privacy without collapsing task performance.

Finally, in the GRU–Transformer case, the adversary is so strong that leakage metrics remain pinned (`leak_conf`≈ 0.98 for all λ). Returns fluctuate considerably: mild penalties ($\lambda = 0.0$–0.5) yield moderate returns (14.50–14.95), while stronger penalties destabilise training (e.g., return 8.57 at $\lambda = 0.2$ or 11.56 at $\lambda = 1.0$). This highlights the difficulty of suppressing deterministic leakage under near-omniscient adversaries: PALADIN can still recover utility, but privacy gains remain minimal.

Together, the sweeps confirm the central role of curriculum-guided penalties. For weaker adversaries (MLP–GRU, T–GRU), moderate to high λ values (≈ 1.0–2.0) provide strong privacy–utility trade-offs. In contrast, for strong adversaries (MLP–Transformer, GRU–Transformer), the best results arise from gentle penalties ($\lambda \approx 0.1$), which preserve task returns while softening leakage variance. Across all cases, overly strong penalties ($\lambda \geq 2.0$) tend to collapse learning or provide no additional privacy benefit.

E.4.1 Ablation Study on NYSE dataset

We evaluated ablated variants across all adversary pairings (Table 7) to understand the contributions of PALADIN's design with NYSE dataset. For the **MLP–GRU** case, removing curriculum (`no_curriculum`) catastrophically destabilises training, with returns collapsing to -371.67. Eliminating the transformation module (`no_transform`) preserves stability (return 14.84) but yields very high leakage (`leak_conf`$= 0.97$). Reducing adversary pressure (`small_adv`, `weak_adv`) achieves moderate returns (18.84–19.78) but fails to meaningfully reduce leakage. Shallow representations (`shallow_phi`) can produce high utility (21.41) but with severely degraded leakage performance (`leak_nll`$= 0.53$).

In the **MLP–Transformer** setting, adversarial pressure is stronger. Again, `no_curriculum` collapses completely (return -232.22). `no_transform` maintains some task performance (15.04) but with extreme leakage confidence (0.97). Weak or small adversaries both underperform (returns 13.68–17.22, leakage confidence 0.99). Shallow feature transformations improve utility (18.86) but with poor leakage suppression (`leak_nll`$= 0.30$). DP baselines once more collapse to returns < 7, with leakage confidence near-saturated (0.99).

For the **Transformer–GRU** configuration, `no_curriculum` collapses training severely (return -389.49). Removing transformations (`no_transform`) sustains moderate returns (17.32) but with leakage confidence 0.91. Smaller or weaker adversaries again fail to improve leakage, and shallow transformations, while yielding the highest utility (22.25), result in very high leakage (`leak_nll`$= 0.47$). DP baselines collapse to 5–6 return with persistent leakage.

Finally, in the **GRU–Transformer** setting, adversaries are strongest. Curriculum removal destabilises training entirely (return -311.12). Removing transformations retains utility (18.42) but exposes leakage (`leak_conf`$= 0.94$). Shallow feature mappings degrade both utility and privacy (return 11.78, leakage confidence 0.92). Smaller or weaker adversaries provide no meaningful privacy improvements. DP baselines again collapse (returns 4.78–10.14) while maintaining high leakage.

Across all settings, the ablations demonstrate that PALADIN's effectiveness arises from the integration of three pillars: (i) curriculum-scheduled penalties, (ii) expressive feature trans-

Table 5: Experimental Results on NYSE dataset

Train-Test Adv	Method	return	return_std	leak_nll	leak_nll_std	leak_conf	leak_conf_std
MLP-GRU-1	baseline	16.797	2.288	0.186	0.306	0.883	0.168
	static_obs	11.659	0.294	0.087	0.155	0.926	0.117
	static_act	14.675	1.937	0.136	0.31	0.937	0.094
	adv_no_cur	12.928	3.709	0.444	1.026	0.923	0.136
	proactive	21.847	0.709	0.156	0.255	0.881	0.183
	dp_rl	10.643	0.513	0.193	0.247	0.848	0.185
	dp_nash	6.871	0.29	0.137	0.206	0.889	0.155
MLP-Transformer	baseline	16.744	0.632	0.31	0.885	0.981	0.011
	static_obs	15.654	1.04	0.018	0.004	0.982	0.004
	static_act	19.246	3.411	0.025	0.007	0.975	0.007
	adv_no_cur	15.091	3.075	0.136	0.262	0.911	0.151
	proactive	24.772	1.006	0.049	0.107	0.957	0.089
	dp_rl	6.945	1.123	0.013	0.002	0.987	0.002
	dp_nash	6.095	1.021	0.013	0.002	0.987	0.002
Transformer-GRU	baseline	16.412	0.942	0.283	0.365	0.846	0.114
	static_obs	10.962	1.826	0.221	0.36	0.887	0.053
	static_act	17.888	0.355	0.328	0.7	0.91	0.002
	adv_no_cur	16.729	2.502	0.236	0.383	0.882	0.044
	proactive	20.484	0.105	0.092	0	0.912	0
	dp_rl	6.347	1.917	0.539	0.731	0.859	0.087
	dp_nash	7.151	1.999	0.528	0.703	0.826	0.126
GRU-Transformer	baseline	17.769	7.179	0.015	0.00018843	0.985	0.00018553
	static_obs	10.841	0.435	0.015	0.00018707	0.985	0.00018419
	static_act	14.611	0.35	0.015	5.04E-06	0.985	4.96E-06
	adv_no_cur	18.012	2.694	0.015	3.26E-05	0.985	3.21E-05
	proactive	19.565	0.102	0.432	1.2494854	0.985	1.67E-06
	dp_rl	6.498	0.498	0.015	0.00013247	0.985	0.00013043
	dp_nash	3.714	0.281	0.015	4.95E-05	0.985	4.88E-05

Table 6: λ sweep of PALADIN on NYSE dataset

Train-Test Adv	return	return_std	leak_nll	leak_nll_std	leak_conf	leak_conf_std
MLP-GRU	0.0	20.77	0.132	0.087	1.896	0.929
	0.1	15.169	0.129	0.302	5.685	0.916
	0.2	12.205	0.159	0.14	0.551	0.887
	0.5	13.768	0.11	0.424	2.397	0.933
	1.0	18.314	0.069	0.065	1.918	0.94
	2.0	11.902	0.192	0.201	2.536	0.843
MLP-Transformer-t	0.0	19.796	0.001	0.013	2.93	0.987
	0.1	21.009	0.018	0.041	2.667	0.96
	0.2	14.554	0.002	0.015	1.757	0.985
	0.5	19.553	0.003	0.016	0.686	0.984
	1.0	14.952	0.001	0.013	5.759	0.987
	2.0	13.94	0.001	0.012	8.463	0.988
Transformer-GRU	0.0	17.672	0.001	0.119	0.183	0.888
	0.1	19.947	0.076	0.196	0.457	0.879
	0.2	10.84	0.093	0.302	0.616	0.866
	0.5	13.958	0.08	0.204	0.185	0.862
	1.0	20.785	0.109	0.481	2.008	0.858
	2.0	22.551	0.068	0.202	2.088	0.884
GRU-Transformer	0.0	14.952	0.001	0.016	3.025	0.984
	0.1	14.517	0	0.015	0.099	0.985
	0.2	8.566	0	0.015	0.151	0.985
	0.5	14.497	0.001	0.016	1.933	0.985
	1.0	11.562	0.006	0.017	2.323	0.983
	2.0	14.359	0	0.015	4.742	0.985

formations, and (iii) strong adversarial supervision. Removing curriculum causes catastrophic failure, removing transformations leaves the agent highly leaky, and weakening the adversary produces fragile trade-offs. Shallow feature mappings highlight that representational depth is critical for systematically distorting sensitive information. DP baselines consistently fail, reaffirming that PALADIN's behavioural shaping offers a qualitatively stronger defence.

PALADIN's success in the financial trading setting stems from its ability to learn *where* and *when* to obfuscate: it discovers which price-volume fluctuations are most predictive of ticker identity and selectively masks them only after a reward-focused warmup. This targeted approach contrasts sharply with blanket noise or parameter-level DP, which cannot discriminate signal from noise in a dynamic

Table 7: Ablation study on the NYSE Dataset

Train-Test Adv	Method	return	return_std	leak_nll	leak_nll_std	leak_conf	leak_conf_std
MLP-GRU	no_transform	14.84	0.635	0.382	1.058	0.971	0
	no_curriculum	-371.674	5.838	0.185	0.246	0.854	0.181
	small_adv	18.842	3.111	0.244	0.263	0.81	0.198
	shallow_phi	21.41	2.868	0.525	1.051	0.882	0.179
	weak_adv	19.78	3.94	0.213	0.271	0.836	0.197
	dp_rl	5.232	0.785	0.173	0.293	0.892	0.156
	dp_nash	4.583	0.785	0.085	0.16	0.928	0.12
MLP-Transformer	no_transform	15.039	9.006	0.032	0.072	0.971	0.063
	no_curriculum	-232.221	451.214	0.008	0	0.992	0
	small_adv	13.678	0.159	0.484	1.429	0.992	0
	shallow_phi	18.862	4.029	0.297	0.853	0.983	0.014
	weak_adv	17.223	3.423	0.013	0.002	0.987	0.002
	dp_rl	6.865	0.681	0.013	0.002	0.987	0.002
	dp_nash	5.791	0.41	0.013	0.001	0.987	0.001
Transformer-GRU	no_transform	17.324	0.006	0.097	0	0.908	0
	no_curriculum	-389.485	1.021	0.522	0.715	0.858	0.094
	small_adv	12.289	0.163	0.087	0.002	0.917	0.002
	shallow_phi	22.251	9.948	0.467	0.572	0.86	0.079
	weak_adv	16.358	0.714	0.147	0.109	0.868	0.082
	dp_rl	5.759	0.535	0.195	0.287	0.88	0.076
	dp_nash	6.496	0.575	0.301	0.41	0.862	0.094
GRU-Transformer	no_transform	18.422	1.799	0.082	0.201	0.937	0.144
	no_curriculum	-311.123	2.248	0.015	0	0.985	0
	small_adv	16.126	2.867	0.432	1.25	0.985	0
	shallow_phi	11.783	1.359	0.101	0.203	0.92	0.148
	weak_adv	12.834	3.568	0.015	0	0.985	0
	dp_rl	4.778	0.641	0.016	0.001	0.984	0.001
	dp_nash	10.141	2.792	0.021	0.016	0.98	0.016

market environment. Moreover, PALADIN avoids the instability of fixed-penalty adversarial shaping by ramping up privacy pressure only after the policy has converged to a strong utility baseline.

However, several limitations remain. First, our financial trading benchmark uses pre-normalised OHLCV windows and deterministic replay, which simplifies dynamics relative to live trading. Real-world deployment would require handling non-stationary markets and partial observability. Second, while we show cross-backbone robustness (PPO), extending PALADIN to multi-asset strategies or continuous portfolio rebalancing is non-trivial and may require richer transformation networks.

F Broader Impacts

By embedding privacy directly into the RL loop, PALADIN offers practitioners a principled way to mitigate behavioural data leakage in sensitive applications such as autonomous vehicles, personalised healthcare and financial trading. This proactive defence can enhance user trust, comply with data-protection regulations (e.g. GDPR), and reduce the risk of surveillance or identity inference from smart-agent behaviours.

On the other hand, adversarial privacy shaping could be misused to conceal malicious or unsafe agent behaviours (e.g., evading auditing in critical systems) or to obfuscate policy actions in adversarial settings such as cybersecurity. Therefore, care must be taken to balance privacy with accountability and transparency, and to ensure that privacy-preserving agents remain subject to appropriate oversight and testing before deployment.