

USENIX

THE ADVANCED COMPUTING
SYSTEMS ASSOCIATION

RAIL: Risk-Aware Human-in-the-Loop Framework with Adaptive Intrusion Response for Autonomous Vehicles

Dawood Wasif, *Virginia Tech;* Terrence J. Moore, *US DEVCOM Army Research Laboratory;* Seunghyun Yoon and Hyuk Lim, *Korea Institute of Energy Technology* (*KENTECH*)*;* Dan Dongseong Kim, *The University of Queensland;* Frederica F. Nelson, *US DEVCOM Army Research Laboratory;* Jin-Hee Cho, *Virginia Tech*

https://www.usenix.org/conference/vehiclesec26/presentation/wasif-rail

This paper is included in the Proceedings of the
4th USENIX Symposium on Vehicle Security and Privacy.

August 10–11, 2026 • Baltimore, MD, USA

ISBN 978-1-939133-56-4

Open access to the Proceedings of the
4th USENIX Symposium on Vehicle Security and Privacy
is sponsored by USENIX.

ARTIFACT EVALUATED USENIX AVAILABLE
ARTIFACT EVALUATED USENIX FUNCTIONAL

RAIL: Risk-Aware Human-in-the-Loop Framework with Adaptive Intrusion Response for Autonomous Vehicles

Dawood Wasif[1] *Terrence J. Moore*[2] *Seunghyun Yoon*[3] *Hyuk Lim*[3]
Dan Dongseong Kim[4] *Frederica F. Nelson*[2] *Jin-Hee Cho*[1]

[1]*Virginia Tech*
[2]*US DEVCOM Army Research Laboratory*
[3]*Korea Institute of Energy Technology (KENTECH)*
[4]*The University of Queensland*

Abstract

Autonomous vehicles must remain safe and effective when encountering rare long-tailed scenarios or cyber–physical intrusions during driving. We propose a risk-aware human-in-the-loop framework, RAIL, that closes the detect–respond–learn loop within a unified runtime architecture. RAIL introduces an interpretable Intrusion Risk Index (IRI) that fuses heterogeneous runtime cues, including curvature actuation integrity, time-to-collision proximity, and observation-shift consistency, through a weighted Noisy-OR to enable probabilistic, source-decomposed risk quantification at control rate. When risk rises, a contextual bandit selects cue-specific shields and a graded authority blending mechanism interpolates between nominal and safeguarded actions, preserving efficiency under low risk while enabling proactive containment under threat. RAIL further integrates dual reward shaping and risk-prioritized replay so that takeovers and near misses guide learning. Across MetaDrive and CARLA, RAIL matches expert-level success while reducing safety violations and operator interventions relative to RL, safe RL, offline and imitation baselines. Under Controller Area Network (CAN) injection and LiDAR spoofing, RAIL reduces attack success rates by up to 48% and lowers disengagement under attack by more than 50%, showing that explicit multi-source risk modeling yields strong robustness gains in safety-critical autonomy.

1 Introduction

From Pilot Systems to Open-World Autonomy. Autonomous vehicles (AVs) are transitioning from controlled pilots to open-world deployment, where reliability must extend beyond nominal driving to adversarial and safety-critical conditions. Advances in large-scale simulation, high-capacity function approximation, and model-based and model-free control have produced competent lane keeping, merging, and urban navigation, often approaching human-level performance under benign settings [30]. However, deployment exposes a systems-level gap: detection, control, and learning remain loosely coupled. Real-world operation requires more than task competence. An AV must *detect* anomalous conditions, whether rare environmental events or deliberate cyber–physical intrusions, *translate* those detections into calibrated control adaptations at actuation rate, and *integrate* the outcome into learning so similar failures become less likely. No existing subsystem guarantees this closed-loop detect–respond–learn capability under adversarial conditions. Consider, for example, an autonomous vehicle in urban traffic when a compromised electronic control unit injects a bounded steering bias onto the Controller Area Network bus. The trajectory drifts within a few control cycles, yet the learned policy receives no structured signal attributing the anomaly to its source, and the human safety operator has not yet perceived the deviation. Scenarios of this form expose the gap between cyber-physical detection and calibrated control adaptation at the actuation rate that motivates this work.

Limits of Implicit Safety. A dominant class of approaches encodes safety *implicitly* within the learning objective: RL may violate constraints during exploration, Safe RL relies on carefully tuned scalar penalties that degrade under sparse cost signals [2, 40], and imitation or offline RL inherit distributional bias [6, 18, 24]. These methods treat safety as an external scalar constraint rather than an internal state variable, so the policy lacks an explicit representation of *what* failed or *which channel* is compromised. Safety boundaries are learned indirectly, offering neither decomposed attribution nor graded response at test time.

Limits of Purely Reactive Human Oversight. Human-in-the-loop (HITL) frameworks incorporate demonstrations, interventions, or shared control to improve safety [19, 23, 25, 36], with recent methods such as HACO [25] and HAIM-DRL [19] achieving strong nominal performance. However, when human correction is the dominant containment mechanism, the policy cannot act preemptively before the operator perceives the threat, and takeover is encoded as a binary override rather than a structured signal distinguishing perception, planning, and actuation hazards [23, 26]. The limitation lies in purely reactive use of the human, not in human oversight itself: calibrated automated mitigation executing before takeover is

required, with the operator retained for exception handling.

The Architectural Gap: From Implicit Penalties to Explicit Risk. The limits of scalar safety costs and reactive human intervention expose a structural gap in autonomy stacks. Existing methods treat safety as an outcome to penalize after violation rather than as a calibrated runtime variable that governs control. Implicit approaches infer safety boundaries from rewards or takeovers but do not expose a decomposable representation of risk. As a result, the system cannot explain *why* intervention occurred, *which subsystem* triggered concern, or provide an auditable trail aligned with emerging cybersecurity requirements such as UN R155 and ISO/SAE 21434 [20, 29]. Explicit risk modeling elevates risk to a first-class variable: a calibrated, multi-source estimate computed at control rate and directly coupled to shield selection and graded authority blending. Because risk is decomposed by source, mitigation becomes targeted rather than binary, and learning receives structured feedback. This shift to explicit, decomposed risk estimation is essential for adversarially robust and certifiable autonomous systems, as demonstrated by RAIL.

RAIL: A Runtime Risk-Aware Control and Learning Architecture. We introduce Risk-Aware Human-in-the-Loop with Adaptive Intrusion Response (**RAIL**), a runtime architecture that operationalizes explicit risk modeling and closes the detect–respond–learn loop within a single system. RAIL elevates risk to a first-class control variable through a probabilistic Intrusion Risk Index (IRI) that fuses three runtime cues, a curvature-consistency cue for actuation integrity, a time-to-collision cue for planning hazards, and an observation-shift cue for perception anomalies, via weighted Noisy-OR aggregation, converting anomaly detection into calibrated, source-aware control adaptation.

When risk is low, the nominal policy executes without interference. As risk increases, a lightweight contextual bandit selects a cue-specific shield, and the executor blends nominal and safeguarded actions with authority that grows monotonically with risk. This enables proactive, graded containment rather than binary takeover. Human override remains available, but intervention shifts from primary containment to rare exception handling. The learning loop integrates penalties for safety events with risk-aware shaping and prioritized replay, ensuring that takeovers and near misses provide structured feedback rather than ad hoc correction. RAIL tightly couples detection, mitigation, and learning to deliver interpretable, composable, and adversarially robust autonomy.

Figure 1 illustrates the overall runtime architecture. At each control step, the AV agent proposes an action a_t based on observations o_t. Before execution, this action is evaluated by a runtime safety layer that performs risk assessment and, when necessary, mitigates elevated risk by adjusting the action. The executed control u_t is then applied to the environment. The environment returns updated observations, while a human operator may provide situational cues, manual override, or corrective feedback. These signals are incorporated into

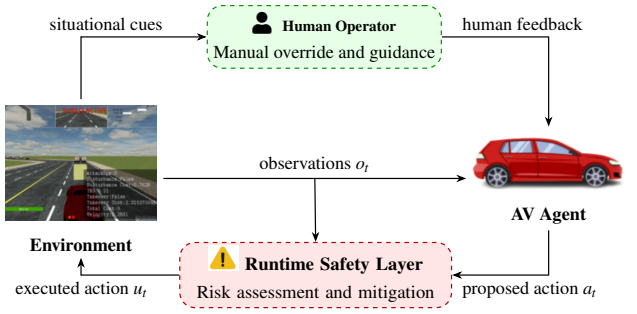


Figure 1: Risk-Aware Runtime Architecture for Human-in-the-Loop Autonomous Driving.

both the policy update and the runtime risk arbitration mechanism, forming a closed detect–assess–mitigate–learn loop that tightly couples anomaly detection, control adaptation, and experience-driven improvement.

Key Contributions. Our contributions are fourfold. First, we present **RAIL as a runtime risk-aware control and learning architecture** that closes the detect–respond–learn loop within a single system. Unlike prior HITL approaches that treat detection, mitigation, and learning as loosely coupled components [19, 23, 25, 26], RAIL tightly integrates probabilistic risk assessment, graded cue-specific shielding, human escalation, and experience replay at a control rate. This integration enables proactive containment of cyber-physical threats before human intervention becomes necessary.

Second, we introduce the **Intrusion Risk Index (IRI)**, an interpretable and multi-source risk quantification mechanism. IRI fuses complementary runtime cues, including curvature uncertainty, time-to-collision proximity, and LiDAR observation-shift, through a weighted Noisy-OR formulation. In contrast to implicit scalar safety costs, IRI decomposes risk by source, supports contextual shield selection, and provides an auditable trail at each control step.

Third, we develop a **safety-aware learning framework** that leverages structured operator feedback. By combining explicit penalties for violations and takeovers with risk-based shaping and prioritized replay, RAIL focuses learning on high-risk transitions while preserving nominal behavior, accelerating safety alignment in an off-policy actor–critic backbone.

Finally, we provide **comprehensive empirical validation** under both nominal and adversarial conditions. Across MetaDrive and CARLA, and under Controller Area Network (CAN) injection and LiDAR spoofing attacks, RAIL demonstrates improved safety, robustness, and reduced operator burden relative to RL, safe RL, imitation, offline RL, and prior HITL baselines, showing that explicit risk modeling yields calibrated control rather than binary fail-safe behavior.

2 Related Work

In this section, we position RAIL within prior work on learning-based control, runtime safety mechanisms, human-

in-the-loop autonomy, and automotive intrusion detection, highlighting architectural limitations that motivate explicit runtime risk modeling.

2.1 Learning-Based and HITL Autonomy

Learning-based control for autonomous driving is largely built on deep reinforcement learning methods that provide strong baselines for continuous decision making. Algorithms such as Soft Actor-Critic (SAC) and Proximal Policy Optimization (PPO) enable data-efficient optimization through entropy regularization and clipped surrogate objectives [16, 37]. Safe RL extends these methods with cost constraints, including constrained policy optimization (CPO) [2] and Lagrangian variants that tune safety multipliers [15, 40]. Although these approaches reduce violations relative to unconstrained RL, safety remains encoded as scalar penalties in the objective. Risk is not represented as an explicit runtime variable, and mitigation is learned indirectly rather than executed through calibrated control adaptation at the actuation rate.

Complementary lines of work seek to reduce unsafe exploration through imitation and offline learning. Behavioral cloning [6] and adversarial approaches such as Generative Adversarial Imitation Learning (GAIL) [18] bootstrap policies from expert demonstrations, while interactive variants including Human-Gated DAgger (HG-DAgger) [23] and intervention-weighted regression [26] incorporate corrective labels during execution. Offline reinforcement learning methods such as Conservative Q-Learning (CQL) [24] rely on static datasets to limit exploration risk. Although these techniques improve data efficiency and reduce training-time hazards, safety knowledge remains embedded in data distributions rather than operationalized as structured runtime risk. As a result, they provide neither decomposed risk attribution nor graded authority modulation under adversarial perturbations.

More recently, human-in-the-loop frameworks explicitly integrate operator feedback into policy optimization. Human–AI Copilot Optimization (HACO) [25] penalizes divergence from inferred human preferences, and Human-as-AI-Mentor DRL (HAIM-DRL) [19] incorporates mentoring costs during exploration. These methods reduce intervention frequency and achieve strong nominal performance. However, intervention remains fundamentally reactive: takeover events are treated as scalar penalties rather than structured, multi-source indicators of risk. The policy cannot communicate which subsystem triggered concern, nor can it calibrate mitigation strength according to risk severity. In adversarial settings, this limits preemptive containment and forensic traceability.

In contrast, RAIL introduces **explicit, decomposed runtime risk modeling** that directly governs shield selection and graded control blending. By elevating risk to a first-class control variable, RAIL bridges the gap among detection, mitigation, and learning under cyber-physical threats.

Table 1: Comparison of runtime safety filtering architectures.

Capability	**CBF** [5]	**HJ** [12]	**Simplex** [38]	**Shield** [4]	**RAIL (Ours)**
Model-based guarantee	✔	✔	✔	▲	▲
Multi-source cue fusion	✘	✘	✘	✘	✔
Online mitigation adaption	✘	✘	✘	✘	✔
Operator awareness	✘	✘	✘	✘	✔
Graded authority blending	✘	✘	✘	▲	✔
Risk decomposition	✘	✘	✘	▲	✔
Model-free compatibility	✘	✘	▲	▲	✔

✔ = supported, ▲ = partially supported, ✘ = not supported. CBF = Control Barrier Function; HJ = Hamilton–Jacobi reachability; Shield = reactive shield synthesis.

2.2 Risk-Sensitive Control and Runtime Safety

A substantial body of work studies risk-sensitive reinforcement learning as a means to reduce catastrophic outcomes. These approaches model the tails of return distributions rather than optimizing expected reward alone. Techniques based on coherent risk measures and distributional critics, such as Implicit Quantile Networks (IQN) and Quantile Regression Deep Q-Networks (QR-DQN), replace the standard objective with Conditional Value-at-Risk (CVaR) or entropic risk formulations [14, 43]. Subsequent advances combine model uncertainty with distributional value estimation to improve robustness in safety-critical robotics [34], and extend proximal policy gradients with safety or risk constraints [44]. While these methods attenuate high-loss outcomes during training, risk is defined over abstract reward distributions rather than grounded in explicit, multi-source runtime signals. They leave open how to operationalize heterogeneous cues such as perception anomalies or vehicle-to-everything (V2X) inconsistencies, and how to translate elevated concern into interpretable, graded control adaptation at actuation rate [21, 33].

Complementary lines of research enforce safety through runtime filtering rather than reward shaping. Control Barrier Functions (CBFs) guarantee forward invariance of a safe set by solving a quadratic program at each control step under control-affine dynamics assumptions [5, 10]. Hamilton–Jacobi reachability analysis computes backward reachable tubes to certify safety over finite horizons, accommodating bounded disturbances but requiring offline grid-based value computation [12]. The Simplex architecture [38] switches control to a verified baseline when a monitor is violated, and shield synthesis [4] constructs reactive strategies from temporal-logic specifications. These frameworks provide formal guarantees when accurate dynamics models and constraint specifications are available. However, they typically assume single-constraint formulations, provide binary intervention decisions, and do not adapt mitigation strategies online from operator feedback. Nor do they natively fuse heterogeneous cyber-physical cues into a calibrated risk estimate. RAIL is instead positioned as a **runtime risk orchestration**

layer that can incorporate formally verified controllers as shield arms while additionally providing graded authority blending, multi-source cue fusion, and structured feedback to learning, thereby extending coverage to model-free settings where formal approaches alone are insufficient.

In Table 1, *model-free compatibility* denotes operation without a hand-specified dynamics model or invariant set, *operator awareness* denotes incorporation of human-intervention signals into shield selection or learning updates, and *risk decomposition* denotes attribution of overall risk to an identifiable subsystem source. Existing filtering approaches rely on model-based guarantees with fixed constraints or provide binary containment, and none natively fuse heterogeneous cyber–physical cues, adapt mitigation from operator feedback, or support graded authority blending. RAIL addresses these limitations while remaining compatible with formal controllers that serve as shield components.

2.3 Autonomous Vehicle Intrusion Defense

Automotive cybersecurity research has long established that in-vehicle networks and electronic control units (ECUs) are vulnerable to remote compromise, motivating systematic intrusion detection mechanisms for connected and autonomous vehicles [8, 27]. Subsequent work introduced specification-based monitoring and learning-driven anomaly detection on the Controller Area Network (CAN) bus, leveraging signal timing, message frequency, and fingerprinting techniques to identify malicious deviations [9, 11, 28]. More recent approaches incorporate reinforcement learning and distributional modeling to characterize deviations under uncertainty [22, 42]. In parallel, runtime monitors extend beyond bus-level anomalies to cross-check vehicle dynamics, actuator consistency, and state-estimation residuals, detecting control-relevant inconsistencies during operation [41].

Sensor-layer vulnerabilities further expand the attack surface. LiDAR spoofing, adversarial camera perturbations, and GPS manipulation have been demonstrated in both laboratory and field settings, showing that perception pipelines can be deceived without compromising internal software [3, 7, 35, 39]. Collectively, these studies underscore that autonomous vehicle security spans heterogeneous attack channels across perception, planning, and actuation.

More recent efforts move beyond detection toward runtime response. REACT [17] formulates intrusion response as an optimization problem, selecting countermeasures via linear programming under cost–benefit trade-offs. AVMON [1] combines invariant learning with residual prediction to enable anomaly detection and partial recovery during execution. Despite this progress, detection and control remain architecturally separated in most systems: anomaly scores are generated upstream, while mitigation actions are pre-defined or weakly integrated with human supervisory intent. Human intervention typically functions as an external override rather than as a structured component of containment.

Within the HITL autonomy literature, risk is commonly encoded implicitly as scalar penalties. In HACO [25], the human preference signal is modeled through a proxy advantage $\hat{A}^h$ that penalizes divergence from an inferred value function; the resulting cost is aggregated without attribution to specific risk sources. Similarly, HAIM-DRL [19] integrates takeover events as scalar negative rewards that shape exploration but do not expose the subsystem responsible for intervention. In both cases, human guidance influences learning, yet the representation of risk remains undifferentiated and opaque.

These limitations motivate a more tightly integrated formulation in which detection, containment, and learning are co-designed rather than sequentially layered.

RAIL models **risk as an explicit, decomposable control variable** instead of an implicit scalar penalty. The Intrusion Risk Index (IRI) aggregates cue-level contributions and identifies the dominant source via the inline selection rule $i^\star = \arg\max_i w_i r_i$, where each r_i corresponds to a structured cue aligned with a specific kill-chain layer such as perception, planning, or actuation. A Noisy-OR aggregation provides a probabilistic interpretation as the likelihood that at least one cue signals danger. This structure enables targeted shield selection rather than uniform penalty injection, allowing mitigation to be context-sensitive and subsystem-aware.

This design establishes **detection–mitigation coupling within a unified control loop**. The Intrusion Risk Index (IRI) serves as both detection output and control input, eliminating the interface gap between anomaly assessment and policy execution. A contextual bandit selects mitigation actions conditioned on the cue vector and operator feedback, enabling the response policy to evolve with operational experience. Explicit cue attribution supports auditability through logged risk decomposition, composability by adding new cue terms and corresponding shields without redesigning the architecture, and regulatory traceability through interpretable cause-effect mappings. In contrast to scalar-cost human-in-the-loop approaches, RAIL operationalizes risk as a structured signal that directly informs detection, containment, and learning.

3 Proposed Approach: RAIL

System overview. Figure 2 shows the RAIL pipeline, which turns runtime risk into graded, interpretable control. At each control step, the policy π_θ proposes a nominal action a_t from state s_t. Three cues covering actuation (r_{CURV}), planning (r_{TTC}), and perception (r_{OOD}) are fused via weighted Noisy-OR into $\mathrm{IRI}_t \in [0,1]$. If $\mathrm{IRI}_t \le \tau$, a_t executes unmodified; otherwise a contextual bandit selects a cue-specific shield $\mathcal{S}_{k_t}$ producing a safeguarded action $\tilde{a}_t$, and the executor outputs $u_t = (1-\alpha_t)a_t + \alpha_t \tilde{a}_t$ with α_t monotone in IRI. A human takeover, when issued, overrides u_t. Each annotated transition is stored in a prioritized replay buffer that drives off-policy SAC updates.

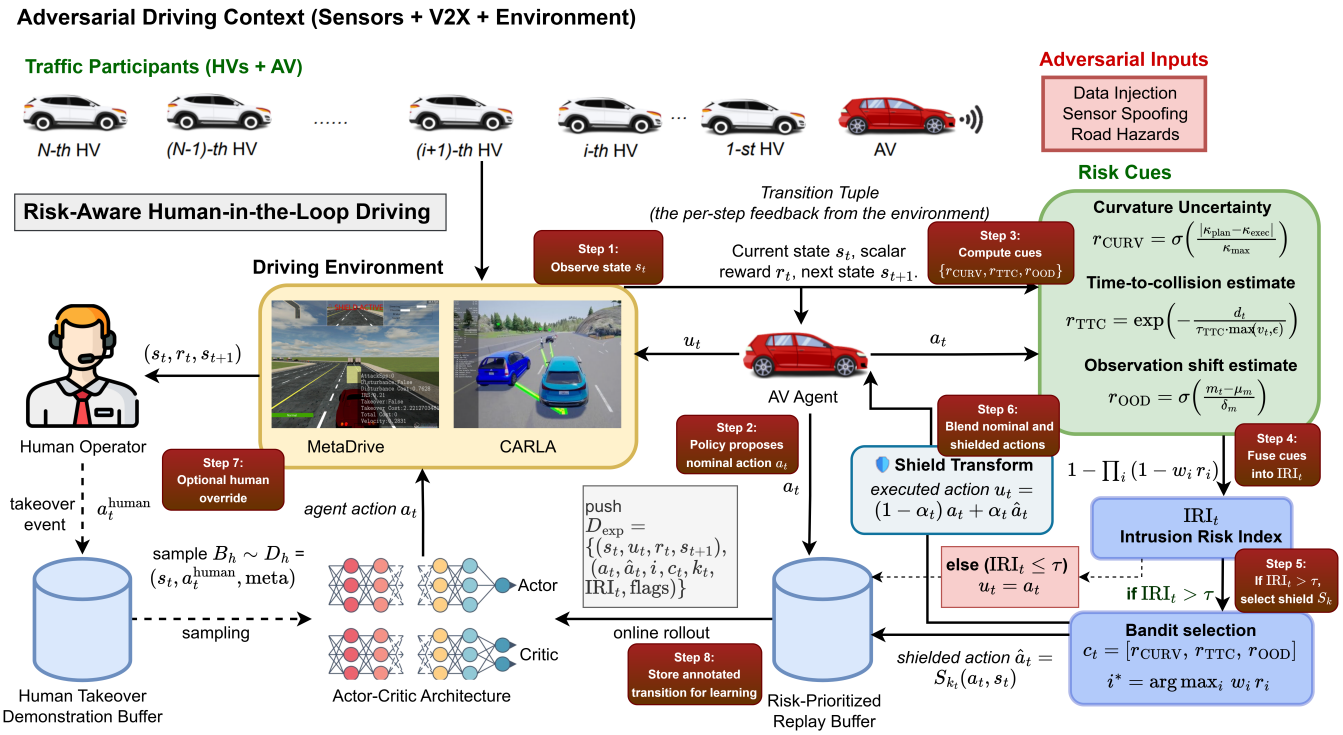


Figure 2: **RAIL architecture.** Steps 1–6 illustrate the runtime control path. The AV agent observes s_t, proposes a nominal action a_t, computes three risk cues $(r_{\text{CURV}}, r_{\text{TTC}}, r_{\text{OOD}})$, and fuses them via Noisy-OR into the Intrusion Risk Index IRI_t. When $\text{IRI}_t \le \tau$, the system follows the low-risk pass-through path and executes $u_t = a_t$. When $\text{IRI}_t > \tau$, a contextual bandit selects a cue-specific shield S_{k_t} to produce $\hat{a}_t = S_{k_t}(a_t, s_t)$, and the executor applies the blended control $u_t = (1-\alpha_t)a_t + \alpha_t \hat{a}_t$. Step 7 shows optional human override, and Step 8 shows that each annotated transition is logged to a risk-prioritized replay buffer for off-policy SAC updates.

Threat model (summary). RAIL targets two adversary classes representative of documented automotive attacks: actuation-channel injection modeling CAN-level ECU compromise [8, 27], and LiDAR spoofing modeling optical relay attacks [7, 39]. We assume black-box adversaries that operate in bounded bursts. The three IRI cues are chosen to cover the actuation, planning, and perception layers exercised by these attacks; full attack parameterizations and a discussion of stronger adversaries appear in Section 4.2.

3.1 Risk-Aware Control Architecture

We consider an ego vehicle operating in continuous state and action spaces. At time t, the vehicle state s_t captures ego kinematics, local map features, and perception outputs, while a stochastic policy $\pi_\theta(a_t \mid s_t)$ generates a continuous control action a_t consisting of steering and longitudinal commands. The environment evolves under the applied control u_t according to the transition model $p(s_{t+1} \mid s_t, u_t, \xi_t)$, where ξ_t represents exogenous disturbances.

RAIL augments this closed-loop interaction with an Intrusion Risk Index $\text{IRI}_t \in [0,1]$ computed from heterogeneous runtime cues spanning cyber and physical layers. When risk remains low, the nominal policy action is executed directly ($u_t = a_t$). As risk increases, a cue-specific shield produces a safeguarded proposal $\tilde{a}_t$, and an execution module blends $\tilde{a}_t$ with a_t according to risk-dependent authority. A human override mechanism remains available but transitions from primary containment to exception handling as automated mitigation becomes more reliable.

Shield selection and gain adaptation are governed by a lightweight contextual bandit conditioned on the current cue vector and delayed performance signals. This design enables mitigation policies to evolve online while preserving computational efficiency. Learning proceeds off-policy using the Soft Actor-Critic (SAC) algorithm [16]. The replay buffer stores the primary transition tuple (s_t, u_t, r_t, s_{t+1}), augmented with structured annotations $(a_t, \tilde{a}_t, \text{IRI}_t, c_t, i^\star, k_t, \text{flags})$. High-risk and human-corrected transitions are prioritized, ensuring that parameter updates concentrate on safety-critical operating regimes rather than nominal behavior. Table 2 summarizes the principal symbols used throughout the paper.

Table 2: Summary of symbols and definitions used in RAIL.

Symbol	Description
s_t	Ego state (kinematics, map, perception)
a_t	Nominal policy action [steer, accel]
u_t	Executed (blended) action
$\tilde{a}_t$	Shield output $\mathcal{S}_{k_t}(a_t, s_t)$
π_θ	Stochastic policy parameterized by θ
ξ_t	Environmental disturbances (including attacks)
IRI_t	Intrusion Risk Index $\in [0,1]$
r_i, w_i	Cue i risk value and fixed weight
$i^\star$	Dominant cue $\arg\max_i w_i r_i$
α_t	Shield authority $\in [0,1]$
c_t	Cue context vector $[r_{\mathrm{CURV}}, r_{\mathrm{TTC}}, r_{\mathrm{OOD}}]$
k_t	Selected shield arm
τ	IRI activation threshold
H_t	Human takeover flag $\in \{0,1\}$
R_t	Shaped reward signal

3.2 Intrusion Risk Index (IRI)

We define the Intrusion Risk Index (IRI) as a unified, interpretable risk metric that fuses heterogeneous cyber–physical cues into a single scalar value for real-time decision support.

Noisy-OR fusion. [32] The IRI aggregates multiple cyber-physical risk cues into a single interpretable value. At each time step t:

$$\mathrm{IRI}_t = 1 - \prod_{i=1}^{M} \big(1 - w_i\, r_i(s_t, a_t)\big), \tag{1}$$

where M is the number of monitored cues, $r_i(s_t,a_t) \in [0,1]$ is the normalized risk contribution of cue i, and $w_i \in [0,1]$ are fixed importance weights with $\sum_i w_i = 1$. The dominant cue $i^\star = \arg\max_i w_i r_i(s_t,a_t)$ is recorded for interpretability. The multiplicative Noisy-OR models the probability that at least one cue indicates danger: a single high-confidence anomaly is sufficient to elevate IRI_t, while multiple weak but consistent signals compound non-linearly.

Three cues suffice for the present study because they span the actuation, planning, and perception layers of the AV kill chain and are computable at control rate from signals already available in our simulator stack. The IRI is modular: additional cues such as GPS/GNSS spoofing indicators, radar–camera disagreement, and V2X consistency terms can be integrated by defining a normalized $r_i \in [0,1]$ and a matching shield arm, without retraining the base policy.

These cues are selected to satisfy four design principles [13, 31]: (i) **Observability**: each cue is computable from signals available at control rate without requiring additional sensing hardware. (ii) **Normalizability**: each raw measurement admits a monotone mapping to $[0,1]$, ensuring that Noisy-OR fusion remains well-defined.(iii) **Complementarity**: under common driving scenarios, the cues activate on distinct failure modes to maximize the information captured by the IRI, as verified empirically in Section 5.5. (iv) **Kill-chain coverage**: the cue set spans actuation, planning, and perception layers.

Table 3: Cue-to-threat mapping across the AV kill chain.

Cue	Kill-chain	Attacks detected	Signal
r_{CURV}	Actuation	CAN injection, actuator fault	Yaw vs. plan
r_{TTC}	Planning	Phantom obstacle, brake disable	Range + vel.
r_{OOD}	Perception	LiDAR spoof / mask / blind	Beam stats

Curvature uncertainty cue. This cue measures control integrity by comparing planned and executed curvature:

$$r_{\mathrm{CURV}}(s_t,a_t) = \sigma\Big(\tfrac{|\kappa_{\mathrm{plan}}(s_t) - \kappa_{\mathrm{exec}}(s_t,a_t)|}{\kappa_{\max}}\Big), \tag{2}$$

where κ_{plan} is the planned path curvature from route geometry, κ_{exec} is the realized curvature from yaw rate over speed with a numerical floor, $\kappa_{\max}$ normalizes by feasible limits, and σ is a logistic squashing. Risk grows as the executed path diverges from the intended trajectory, flagging actuation faults or injected steering.

Time-to-collision cue. This cue converts available reaction time into risk:

$$r_{\mathrm{TTC}}(s_t,a_t) = \exp\Big(-\tfrac{d_t}{\tau_{\mathrm{TTC}}\cdot\max(v_t,\varepsilon)}\Big), \tag{3}$$

where d_t is the path-aligned distance to the most threatening object, v_t is the closing speed, $\varepsilon > 0$ avoids division by zero, and τ_{TTC} sets the time scale. The score approaches one when contact is imminent and decays as available time increases.

Observation shift cue. This cue flags sensor-level distribution shift in LiDAR returns:

$$r_{\mathrm{OOD}}(s_t) = \sigma\left(\frac{\sqrt{(z_t-\mu)^\top(\Sigma+\varepsilon I)^{-1}(z_t-\mu)} - \mu_m}{\delta_m}\right), \tag{4}$$

where $z_t \in \mathbb{R}^{72}$ is the LiDAR range vector, (μ,Σ) are the clean-data mean and covariance with Tikhonov regularization εI, and (μ_m,δ_m) center and scale the Mahalanobis distance. Risk increases as the current observation departs from the nominal sensor model, detecting spoofing, masking, or blinding.

3.3 Risk-Conditioned Shielding and Action Blending

This section describes how risk modulates control authority and how cue-dependent shields are selected and adapted.

Let a_t denote the nominal policy action and $\tilde{a}_t = \mathcal{S}_{k_t}(a_t,s_t)$ the output of shield k_t selected at time t. The executor computes the *executed control input* u_t, which is applied to the vehicle actuators, by blending the nominal and shielded actions using an authority coefficient $\alpha_t \in [0,1]$:

$$u_t = (1-\alpha_t)\,a_t + \alpha_t\,\tilde{a}_t, \tag{5}$$

where α_t is a monotonically non-decreasing function of IRI_t, smoothed via an exponential moving average and rate-limited across control cycles to prevent abrupt authority shifts.

This blending rule satisfies the properties below, consistent with Table 1.

1. **Passivity.** If $\mathrm{IRI}_t < \tau$, then $\alpha_t = 0$ and $u_t = a_t$, meaning the safety layer remains inactive and the nominal policy operates without modification.
2. **Monotonicity.** The authority coefficient α_t is monotonically non-decreasing in IRI_t, ensuring that increased risk cannot reduce the level of shielding applied.
3. **Bounded deviation.** Since $\|u_t - a_t\| = \alpha_t \|\tilde{a}_t - a_t\| \le \|\tilde{a}_t - a_t\|$, the executed action deviates from the nominal action by at most the shield correction magnitude, ensuring that blending never amplifies the intervention.

Shield selection is formulated as a contextual bandit over a small library of interpretable transformations. The context vector is $c_t = [r_{\mathrm{CURV}}, r_{\mathrm{TTC}}, r_{\mathrm{OOD}}]$. Each arm k is assigned a linear score $z_{k,t} = \theta_k^\top c_t$, yielding the selection probability

$$p_t(k) = \frac{\exp(\theta_k^\top c_t)}{\sum_j \exp(\theta_j^\top c_t)}. \tag{6}$$

After executing control input u_t (see Eq. (5)), delayed scalar feedback is computed as an indicator of success (one if no takeover occurs within horizon Δ) minus a small effort penalty proportional to $\|\tilde{a}_t - a_t\|$. The parameter vector θ_{k_t} is then updated in the direction of c_t scaled by this feedback, while other arms remain unchanged. This design preserves locality, since only the contextually relevant shield is adapted; auditability, since the active shield, α_t, and scores $\{z_{k,t}\}$ are logged at each step; and convergence, as temperature annealing drives concentration on the empirically best shield for each cue configuration.

3.4 Risk-Aware Reward Design

This subsection describes the reward formulation used to train RAIL, which integrates discrete safety penalties with continuous risk-aware shaping.

RAIL's training reward combines explicit penalties for safety-critical events with continuous shaping terms that discourage elevated risk before failures occur:

$$\begin{aligned} R_t = r_{\mathrm{task}} &- \lambda_1 \mathbb{1}[\text{coll.}] - \lambda_2 \mathbb{1}[\text{off-road}] - \lambda_3 H_t \\ &- \lambda_4 \mathrm{IRI}_t - \lambda_5 \|\tilde{a}_t - a_t\| \mathbb{1}[\mathrm{IRI}_t > \tau], \end{aligned} \tag{7}$$

where r_{task} rewards driving progress, lane centering, and speed tracking. The explicit component penalizes collisions (λ_1), off-road departures (λ_2), and human takeovers (λ_3). The shaping term penalizes elevated intrusion risk via $\lambda_4 \mathrm{IRI}_t$, encouraging low-risk behavior even without discrete failures. An additional penalty on shield effort, $\lambda_5 \|\tilde{a}_t - a_t\|$ when $\mathrm{IRI}_t > \tau$, discourages excessive reliance on shielding. We set $\lambda_1 = 1.0$, $\lambda_2 = 1.0$, $\lambda_3 = 0.5$, $\lambda_4 = 0.1$, and $\lambda_5 = 0.05$.

3.5 Risk-Prioritized Replay and Training

Unlike standard prioritized replay, which scales the sampling probability by the TD error alone, *risk-prioritized replay* additionally weights transitions by the runtime IRI and by whether a human takeover occurred, so that adaptation concentrates on cyber–physical hazards rather than generic high-error states. This subsection describes how RAIL integrates this mechanism with off-policy SAC learning.

Each transition stores the core tuple (s_t, u_t, R_t, s_{t+1}) together with structured annotations $(a_t, \tilde{a}_t, \mathrm{IRI}_t, c_t, i^\star, k_t, H_t)$. The sampling priority is defined as

$$p_t = |\delta_t|^\eta + \beta \mathrm{IRI}_t + \gamma H_t + \varepsilon_p, \tag{8}$$

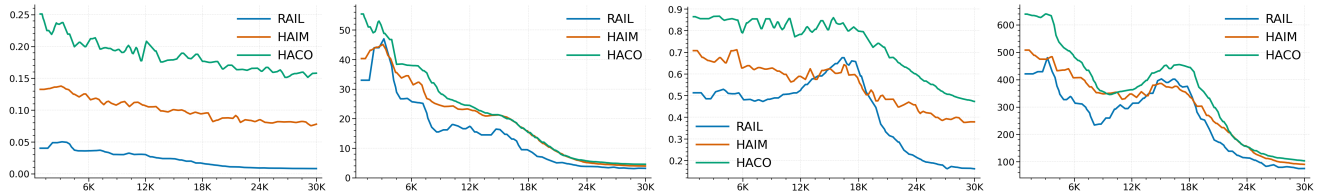
where δ_t is the temporal-difference (TD) error, η controls TD scaling, β weights the contribution of risk magnitude, γ emphasizes human takeovers, and $\varepsilon_p > 0$ prevents sample starvation. We use $\eta = 0.6$, $\beta = 0.3$, $\gamma = 1.0$, and $\varepsilon_p = 10^{-6}$. These choices follow standard prioritized-replay practice: $\eta < 1$ preserves partial uniform coverage, β is set so that risk-based priority contributes on the same order as typical early-training TD magnitudes, $\gamma \ge 1$ guarantees that takeover transitions dominate the priority budget, and ε_p prevents sample starvation. Moderate perturbations of these values did not materially change training dynamics. Transitions are sampled proportionally to p_t, and importance-sampling weights are applied to debias gradient updates.

Algorithm 1 summarizes the integrated runtime and training loop. Training alternates short online rollouts that refresh contextual bandit statistics and collect new experience with off-policy Soft Actor-Critic (SAC) updates from the prioritized buffer. Rollouts run with risk-aware blending and contextual shield selection enabled. Low-risk transitions are retained with small priority to preserve nominal coverage. When $\mathrm{IRI}_t > \tau$, the log records k_t and α_t for bandit credit assignment. Human takeover transitions receive the highest priority, followed by high-risk near-miss events, accelerating safety alignment and reducing future interventions.

4 Experimental Setup

4.1 Evaluation Environments

We evaluate in METADRIVE on procedurally generated road graphs, including straight segments, curves, T/X intersections, ramps, roundabouts, and forks, populated by heterogeneous stochastic traffic. Background vehicles follow Intelligent Driver Model (IDM) longitudinal dynamics with MOBIL lane changing, while static cones, barriers, and occasional slow vehicles induce long-tail interactions. Episodes terminate upon reaching the goal, collision, off-road departure, or timeout (1,000 steps). For cross-simulator generalization, we additionally test in CARLA using a top-down semantic raster



(a) Disturbance rate (b) Disturbance count (c) Takeover rate (d) Takeover count

Figure 3: Training curves on MetaDrive comparing RAIL, HAIM [19], and HACO [25]. RAIL reduces disturbance and operator takeovers fastest and reaches the lowest steady state across all four metrics.

Algorithm 1: RAIL Runtime and Learning Loop

Init: actor π_θ, critics $Q_{\psi_{1,2}}$, targets, replay $\mathcal{D}$; cue weights $\{w_i\}$; bandit params $\{\theta_k\}$; threshold τ; horizon Δ
for each environment step t **do**
  Sample $a_t \sim \pi_\theta(\cdot \mid s_t)$
  Compute $c_t = [r_{\mathrm{CURV}}, r_{\mathrm{TTC}}, r_{\mathrm{OOD}}]$; $\mathrm{IRI}_t \leftarrow 1 - \prod_i (1 - w_i r_i)$
  if $\mathrm{IRI}_t > \tau$ **then** ▷ Shield activation
    $k_t \sim \mathrm{Softmax}(\{\theta_k^\top c_t\})$; $\tilde{a}_t \leftarrow \mathcal{S}_{k_t}(s_t, a_t)$
    $u_t \leftarrow (1-\alpha_t)\, a_t + \alpha_t\, \tilde{a}_t$ ▷ Eq. (5)
  else $u_t \leftarrow a_t$
  end if
  if human takeover **then** $u_t \leftarrow a_t^{\mathrm{human}}$; $H_t = 1$
  else $H_t = 0$
  end if
  Execute u_t; observe s_{t+1}; compute R_t ▷ Eq. (7)
  Push to $\mathcal{D}$ with priority p_t ▷ Eq. (8)
  Sample minibatch from $\mathcal{D}$; update $Q_{\psi_{1,2}}$, π_θ, temperature
  if $\mathrm{IRI}_t > \tau$ and no takeover in next Δ steps **then**
    $\theta_{k_t} \leftarrow \theta_{k_t} + \eta_b\, y_t\, c_t$ ▷ Bandit update
  end if
end for

input with identical control heads, confirming that RAIL's performance gains transfer across observation modalities.

4.2 Threat Model

We evaluate RAIL under representative cyber and physical attack scenarios targeting both control and perception channels. In both cases, the adversary operates under a **black-box** threat model: it has no knowledge of RAIL's internal policy parameters, IRI state, or shield selection, and cannot observe or adapt to the defense in real time. The attacker controls injection timing and magnitude but not the victim's software stack. Attacks are **non-persistent**, occurring in fixed-duration bursts with inter-burst recovery periods, consistent with transient exploitation scenarios documented in the automotive security literature [8, 27].

Stronger adversaries lie outside the current scope. A white-box attacker with knowledge of IRI weights, τ, and shield parameters could in principle craft low-magnitude perturbations that stay below per-cue activation bounds, or exploit shield-selection dynamics by alternating across cue sources. A persistent adversary could maintain steady-state bias across recovery windows, eroding the assumption that inter-burst intervals restore nominal behavior. The present evaluation therefore characterizes RAIL under a widely reported exploitation regime consistent with documented automotive attacks, rather than the worst case. Adaptive thresholding, online cue-weight adaptation, and adversarial training against white-box and persistent attackers are identified as future work in Section 7.

Cyber attack: Action-channel injection. An adversary injects bounded perturbations into steering or longitudinal acceleration after the policy but before the safety layer, modeled as $u_t = \mathrm{sat}(a_t + \eta_t)$. Piecewise-constant biases of up to 50% full scale affect either control channel for 5 s every 30 s, with random start time, sign, and channel selection. This emulates electronic control unit (ECU) or Controller Area Network (CAN) faults that introduce unintended steering or throttle/brake bias. The 50% magnitude reflects the upper range of demonstrated CAN frame injection amplitudes [27], while the 5 s burst and 30 s interval model intermittent exploit windows observed in ECU fuzzing campaigns [8]. These injections are not inherently stealthy: large biases produce observable trajectory deviations that RAIL's curvature cue can detect, though smaller or ramping injections could evade magnitude-based detection and are left to future work.

Physical attack: LiDAR spoofing. A phantom obstacle is created by overwriting a contiguous azimuth sector in the 72-beam LiDAR with short-range returns corresponding to a rigid object 4 m ahead, with ± 0.5 m lateral jitter. Attack bursts last 5 s and recur every 30 s. This setup follows prior demonstrations of relay-based phantom injection and optical-channel LiDAR exploits, where spoofed returns at 3–5 m range were sufficient to trigger emergency braking in commercial systems [7, 39]. The spoofed sector is detectable by RAIL's observation-shift cue as an out-of-distribution perturbation; however, spatially distributed or gradually intensifying spoofs that better approximate natural obstacles may reduce detectability and represent an important direction for future evaluation. An attack is considered successful if a collision or off-road event occurs within 3 s of a burst.

4.3 Baseline Methods

We benchmark RAIL against 12 representative baselines grouped into five categories.

(1) RL with reward shaping. These methods add dense safety penalties to the reward but do not enforce constraints.

1. **Soft Actor-Critic with Reward Shaping (SAC-RS)** [16]. SAC with entropy regularization and dense penalties on collisions, off-road events, and excessive actuation.
2. **Proximal Policy Optimization with Reward Shaping (PPO-RS)** [37]. PPO with a clipped surrogate objective and identical shaping terms to stabilize safety updates.

(2) Safe RL with explicit cost constraints. These approaches separate reward and safety cost signals and enforce constraints during optimization.

3. **Soft Actor-Critic with Lagrangian Constraints (SAC-Lag)** [15]. SAC with a learned dual variable enforcing episodic safety budgets via Lagrangian relaxation.
4. **Proximal Policy Optimization with Lagrangian Constraints (PPO-Lag)** [40]. PPO with responsive dual updates to achieve faster constraint satisfaction.
5. **Constrained Policy Optimization (CPO)** [2]. Trust-region updates projected onto a linearized cost-feasible set using a safety critic.

(3) Offline RL and imitation learning. These methods learn from fixed datasets or expert demonstrations rather than fully online exploration.

6. **Conservative Q-Learning (CQL)** [24]. Offline RL with conservative Q-regularization to penalize out-of-distribution actions.
7. **Behavior Cloning (BC)** [6]. Supervised regression on expert state-action pairs, prone to compounding error.
8. **Generative Adversarial Imitation Learning (GAIL)** [18]. Adversarial occupancy matching to replicate expert visitation distributions.

(4) Conventional human-in-the-loop methods. These approaches incorporate human corrections through data aggregation or weighted regression.

9. **Human-Gated Dataset Aggregation (HG-DAgger)** [23]. Selective human corrections aggregated into the training buffer to address distributional shift.
10. **Intervention-Weighted Regression (IWR)** [26]. Higher regression weights assigned to human takeover transitions.

(5) Modern human-in-the-loop methods. These methods integrate intervention signals into reinforcement learning.

11. **Human-AI Copilot Optimization (HACO)** [25]. Proxy advantage penalizing divergence between agent and human actions during online correction.
12. **Human-AI Mentor Deep Reinforcement Learning (HAIM-DRL)** [19]. Binary takeover-based mentoring cost shaping updates to reduce intervention frequency.

All baselines use the same observation encoder, action bounds, control rate (10 Hz), random seeds, and training budget as RAIL, with hyperparameters from the original works.

4.4 Metrics

We evaluate performance, safety, efficiency, and robustness using episode-aggregated metrics. Let N denote the number of evaluation episodes, T_i the length of episode i, and $\mathcal{A} \subseteq \{1,\dots,N\}$ the subset subjected to adversarial attack.

(1) Test Success Rate (TSR).

$$\mathrm{TSR} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[\mathrm{goal}_i = 1 \wedge \mathrm{coll}_i = 0 \wedge \mathrm{offroad}_i = 0]\,. \quad (9)$$

TSR measures the fraction of episodes that reach the goal without safety violations; higher values indicate reliable task completion.

(2) Test Return (TR).

$$\mathrm{TR} = \frac{1}{N}\sum_{i=1}^{N} G_i, \quad G_i = \sum_{t=0}^{T_i} r_t^{(i)}, \quad (10)$$

where $r_t^{(i)}$ follows (7). TR reflects cumulative task performance and driving quality.

(3) Test Safety Violations (TSV).

$$\mathrm{TSV} = \frac{1}{N}\sum_{i=1}^{N}\left(n_{\mathrm{coll}}^{(i)} + n_{\mathrm{offroad}}^{(i)}\right). \quad (11)$$

TSV averages collision and off-road counts per episode; lower values indicate safer behavior.

(4) Disturbance Rate (DR).

$$\mathrm{DR} = \frac{1}{\sum_{i=1}^{N} T_i}\sum_{i=1}^{N}\sum_{t=0}^{T_i} \mathbb{1}\left[|a_t^{\mathrm{acc},(i)}| > a_{\max}\right]. \quad (12)$$

DR quantifies excessive longitudinal acceleration, capturing ride comfort and traffic smoothness.

(5) Training Safety Violations ($\mathrm{TSV}_{\mathrm{train}}$).

$$\mathrm{TSV}_{\mathrm{train}} = \sum_{k=1}^{K}\left(n_{\mathrm{coll}}^{(k)} + n_{\mathrm{offroad}}^{(k)}\right). \quad (13)$$

This aggregates all safety events during training, reflecting exploration cost.

(6) Training Data Usage (TDU).

$$\mathrm{TDU} = \sum_{k=1}^{K} T_k. \quad (14)$$

TDU measures total interaction steps and thus sample efficiency.

(7) Disengagement Rate under Attack (DRA).

$$\mathrm{DRA} = \frac{1}{|\mathcal{A}|} \sum_{i \in \mathcal{A}} \mathbb{1}[\mathrm{takeover}_i = 1]. \tag{15}$$

DRA captures reliance on human takeover under adversarial conditions; lower is better.

(8) Attack Success Rate (ASR).

$$\mathrm{ASR} = \frac{1}{|\mathcal{A}|} \sum_{i \in \mathcal{A}} \mathbb{1}[\mathrm{coll}_i = 1 \vee \mathrm{offroad}_i = 1]. \tag{16}$$

ASR measures the fraction of attacked episodes resulting in safety failure and serves as the primary robustness metric.

4.5 Training and Implementation Details

All methods use Soft Actor-Critic (SAC) with shared hyperparameters selected via grid search: actor, critic, and entropy learning rates of 1×10^{-4}, batch size 1024, horizon 1,000, learning starts after 100 steps, control at 10 Hz, action clipping, and first-order rate limiting. Observations include ego kinematics, a 72-beam LiDAR with 60 m range, side detectors, lane features, and RAIL diagnostics. Actions $a_t = [a_t^{\mathrm{steer}}, a_t^{\mathrm{acc}}] \in [-1,1]^2$ map to ± 0.35 rad steering and $[-4.0, 2.5]\,\mathrm{m\,s^{-2}}$ acceleration. The Intrusion Risk Index uses weights $(0.3, 0.4, 0.3)$ with threshold $\tau = 0.3$ (sensitivity analysis in Section 5.5). The reward follows Eq. (7). All methods and results use five random seeds; the best checkpoint on the validation split is retained for evaluation. All experiments run on a single NVIDIA RTX 3090 GPU (24 GB VRAM); RAIL trains in approximately 2.5 hours for 30K steps.

Scripted human surrogate. Large-scale evaluation with real operators is infeasible, so we use a deterministic scripted surrogate in place of a human co-pilot, consistent with prior HITL evaluations [19, 25]. The surrogate monitors the same cue vector c_t as the policy and issues a takeover when IRI_t exceeds a higher threshold $\tau_H = 0.7$, well above the shield activation threshold $\tau = 0.3$, or when a collision or off-road cue persists for more than five consecutive control steps while shielding is active. On activation, it applies an IDM-style braking command with lane-centering steering until the triggering condition clears. The surrogate therefore acts as a safety backstop that fires only after proactive shielding has already engaged, not as a primary controller. Section 5.4 isolates the respective contributions of the shield and the surrogate.

5 Experimental Results and Analysis

This section presents a systematic evaluation of RAIL across nominal, adversarial, and ablation settings. We organize the empirical study around five guiding research questions that examine performance, robustness, component contributions, sensitivity, and practical overhead.

We structure the analysis around the following questions:

Q1: *How does RAIL perform relative to reinforcement learning, safe reinforcement learning, offline reinforcement learning, imitation learning, and human-in-the-loop baselines in terms of safety, task performance, and traffic smoothness?*

Q2: *To what extent does RAIL maintain robustness under representative cyber-physical intrusion scenarios?*

Q3: *Which architectural components of RAIL contribute most significantly to overall performance and safety gains?*

Q4: *How sensitive is RAIL to key design parameters, including the Intrusion Risk Index threshold, cue weights, and cue interactions?*

Q5: *What are the runtime overhead and human intervention requirements associated with RAIL?*

5.1 Comparative Performance

We first address Q1 by comparing RAIL against reinforcement learning, safe reinforcement learning, offline learning, imitation learning, and human-in-the-loop baselines under nominal conditions.

Reward-shaped RL vs. Safe RL. Table 4 shows that reward-shaped RL attains high return but weak safety. SAC-RS achieves 384.56 return but incurs 0.87 safety violations and 0.015 disturbance, while PPO-RS drops to 305.41 return with 4.12 violations. Safe RL reduces violations at the cost of performance: SAC-Lag lowers violations to 0.78 but achieves only 0.71 success rate, and CPO collapses to 0.21 success. These results suggest that implicit cost signals, whether reward penalties or Lagrangian multipliers, lack failure-mode decomposition, limiting targeted mitigation.

Offline and Imitation Learning. CQL, BC, and GAIL underperform across all metrics (SR ≤ 0.13), indicating that fixed datasets and supervised objectives fail to capture the diversity of procedurally generated MetaDrive scenarios.

Human-in-the-Loop Methods vs. RAIL. Figure 3 shows that RAIL reduces disturbance and takeovers more rapidly than HACO and HAIM-DRL during training. At convergence (Table 4), RAIL achieves 360.65 return, matches the top success rate (0.85), and records the lowest test safety violations (0.75) and training violations (29.07) using only 30K samples. Compared to HAIM-DRL, RAIL improves return by 1.8% while maintaining comparable disturbance and success. RAIL is the only method that simultaneously matches expert-level success, minimizes training safety violations, and preserves sample efficiency. Similar trends appear in CARLA (Table 5), where RAIL achieves 1609.70 return and 0.41 success rate with 8K steps, confirming cross-simulator generalization.

5.2 Robustness to Cyber-Physical Intrusions

We next address Q2 by evaluating RAIL under actuation-channel injection and perception spoofing attacks.

Table 4: Performance comparison on MetaDrive. Values are mean ± std over five seeds; best results in **bold**.

Category	Method	Total Training Safety Violation	Training Data Usage	Test Return	Test Safety Violation	Disturbance Rate	Test Success Rate
Expert	Human	–	–	$388.16_{\pm 45.00}$	$0.03_{\pm 0.00}$	0	1.00
RL	SAC-RS [16]	$2.78K_{\pm 0.97K}$	1M	$\mathbf{384.56}_{\pm \mathbf{37.5}}$	$0.87_{\pm 1.47}$	$0.015_{\pm 0.005}$	$0.83_{\pm 0.32}$
	PPO-RS [37]	$27.51K_{\pm 3.86K}$	1M	$305.41_{\pm 14.23}$	$4.12_{\pm 1.24}$	$0.021_{\pm 0.014}$	$0.67_{\pm 0.12}$
Safe RL	SAC-Lag [15]	$1.98K_{\pm 0.75K}$	1M	$352.46_{\pm 108.78}$	$0.78_{\pm 0.58}$	$0.019_{\pm 0.007}$	$0.71_{\pm 0.79}$
	PPO-Lag [40]	$15.46K_{\pm 5.13K}$	1M	$298.98_{\pm 50.99}$	$3.28_{\pm 0.38}$	$0.025_{\pm 0.016}$	$0.52_{\pm 0.27}$
	CPO [2]	$4.36K_{\pm 2.22K}$	1M	$194.06_{\pm 108.86}$	$1.71_{\pm 1.02}$	–	$0.21_{\pm 0.29}$
Offline RL	CQL [24]	–	49K	$116.45_{\pm 34.94}$	$3.68_{\pm 7.61}$	$0.007_{\pm 0.006}$	$0.13_{\pm 0.09}$
IL	BC [6]	–	49K	$36.13_{\pm 10.06}$	$1.05_{\pm 0.54}$	$0.012_{\pm 0.017}$	$0.01_{\pm 0.02}$
	GAIL [18]	$3.68K_{\pm 3.17K}$	49K	$108.36_{\pm 16.08}$	$4.18_{\pm 1.25}$	$0.001_{\pm 0.0009}$	$0.03_{\pm 0.01}$
Conventional HITL	HG-Dagger [23]	35.58	50K	106.21	2.63	0.108	0.04
	IWR [26]	69.74	50K	298.87	3.61	0.122	0.61
Human-AI Copilot	HACO [25]	$30.05_{\pm 10.89}$	30K	$350.01_{\pm 9.72}$	$0.78_{\pm 0.85}$	$0.038_{\pm 0.0083}$	$0.83_{\pm 0.07}$
Human as AI Mentor	HAIM-DRL [19]	$29.84_{\pm 10.25}$	30K	$354.34_{\pm 11.08}$	$0.76_{\pm 0.28}$	$\mathbf{0.0023}_{\pm \mathbf{0.00072}}$	$\mathbf{0.85}_{\pm \mathbf{0.03}}$
Risk-Aware HITL	**RAIL (Ours)**	$\mathbf{29.07}_{\pm \mathbf{11.36}}$	**30K**	$360.65_{\pm 23.06}$	$\mathbf{0.75}_{\pm \mathbf{0.03}}$	$0.0027_{\pm 0.00052}$	$\mathbf{0.85}_{\pm \mathbf{0.07}}$

Table 5: CARLA generalization with top-down semantic raster input.

Method	**Training Data**	**TSV↓**	**TR↑**	**TSR↑**
PPO [37]	500K	80.84	1591.00	0.35
HACO [25]	8K	12.14	1578.43	0.35
HAIM [19]	8K	**11.25**	1590.85	0.38
RAIL (Ours)	**8K**	12.38	**1609.70**	**0.41**

Table 6: Robustness on MetaDrive under cyber and physical intrusions.

Method	**CAN Injection**	**LiDAR Spoof**
	TSR↑ / DRA↓ / ASR↓	*TSR↑ / DRA↓ / ASR↓*
PPO [37]	0.23 / 0.85 / 0.62	0.49 / 0.21 / 0.23
HACO [25]	0.58 / 0.90 / 0.75	0.74 / 0.24 / 0.29
HAIM [19]	0.62 / 0.74 / 0.65	0.77 / 0.17 / 0.20
RAIL (Ours)	**0.68 / 0.37 / 0.34**	**0.80 / 0.03 / 0.11**

Robustness Results. Table 6 shows that RAIL is the most robust method under both intrusion types. Under Controller Area Network injection, RAIL achieves TSR 0.68 compared to PPO (0.23), HACO (0.58), and HAIM-DRL (0.62), while reducing DRA to 0.37 (HACO: 0.90, HAIM-DRL: 0.74) and ASR to 0.34 (HAIM-DRL: 0.65). Under LiDAR spoofing, RAIL attains TSR 0.80 with near-zero DRA (0.03) and ASR (0.11). The concurrent improvement in TSR and reduction in DRA and ASR indicates active containment rather than conservative avoidance: cue-driven shielding mitigates both actuation corruption and phantom obstacles without frequent human takeover or performance collapse.

Table 7: RAIL ablations on MetaDrive.

Variant	**SR↑**	**SV↓**	**DR↓**	**AIRI↓**	**AS↑**
w/o curvature cue	0.78	0.80	0.0140	0.17	22.7
w/o TTC cue	0.67	1.02	0.0220	**0.12**	22.5
w/o OOD cue	0.83	0.77	0.0049	0.15	**23.3**
w/o bandit (fixed)	0.79	1.10	0.0180	0.16	19.7
w/o weights (uniform)	0.81	0.79	0.0056	0.16	20.4
w/o shield penalty	0.74	**0.72**	0.0044	**0.12**	16.6
RAIL (full)	**0.85**	0.75	**0.0027**	0.15	22.8

SR = success rate; SV = safety violations; DR = disturbance rate; AIRI = avg. IRI per episode; AS = avg. speed (km/h).

Attack Model Grounding. The CAN injection model, implemented as bounded additive bias on actuator channels with randomized timing, reflects real-world automotive exploits in which compromised electronic control units inject arbitrary frames onto the bus [8, 27]. The LiDAR spoofing setup follows documented relay-based phantom injection and optical-channel attacks [7, 39]. Both intrusions are scheduled periodically rather than adaptively; evaluating adversaries that respond to RAIL's shielding strategy remains future work.

5.3 Component Ablation Analysis

We next address Q3 by isolating the contribution of each RAIL component under nominal conditions.

Table 7 highlights distinct roles across cues and modules.

Table 8: IRI threshold (τ) sensitivity on MetaDrive.

τ	SR↑	SV↓	DR↓	AS↑	Shield %
0.1	0.79	0.61	0.0016	18.9	41.7
0.2	0.84	0.69	0.0021	21.0	24.8
0.3	**0.86**	**0.74**	**0.0028**	**22.5**	**13.6**
0.4	0.82	0.95	0.0076	23.3	7.4
0.5	0.75	1.21	0.0142	23.8	4.6

Shield % = fraction of control steps with $\text{IRI}_t > \tau$.

Removing the **time-to-collision (TTC)** cue causes the largest degradation, reducing SR to 0.67 and increasing SV to 1.02 and DR to 0.022, as braking shields fail to activate in time. Removing the **curvature** cue lowers SR to 0.78 and raises DR to 0.014 due to unchecked actuation faults. Dropping the **out-of-distribution (OOD)** cue preserves SR (0.83) but increases DR (0.0049) and AS (23.3), reflecting reduced caution under sensor anomalies. Replacing the adaptive **contextual bandit** with a fixed shield increases SV to 1.10 and reduces AS to 19.7, demonstrating that cue-aware arbitration is necessary to match shield type to risk context. Finally, removing the **shield-effort penalty** (λ_5 in (7)) causes persistent over-shielding, resulting in a 27% reduction in average speed (16.6 vs. 22.8 km/h) and the lowest SR (0.74). This confirms that the penalty term is essential to balance safety intervention and task efficiency.

Figure 4 shows qualitative examples where IRI spikes trigger dominant-cue selection and shielded control. For each cue type, RAIL responds with a targeted intervention that averts the incident while maintaining driving progress.

5.4 Shield-Only vs. Surrogate-Only Ablation

A natural concern with any human-in-the-loop design is whether reported robustness is driven by the proactive safety layer or by the fallback operator. We therefore decompose RAIL into its two intervention mechanisms and evaluate four configurations under nominal driving and both attack regimes from Section 4.2: *RAIL (full)* with both active, *Shield only* with the scripted surrogate disabled, *Surrogate only* with cue-conditioned shielding disabled, and *Neither*, which reduces RAIL to its SAC backbone. All four variants share the same policy weights, observations, and control rate.

Table 9 shows a consistent ordering across all three settings: *Shield only* sits much closer to *RAIL (full)* than *Surrogate only* does. Under nominal driving, *Shield only* recovers most of the full-system performance (TSR 0.82 vs. 0.85), while *Surrogate only* lags at 0.76 and the base policy sits at 0.71, consistent with SAC-RS in Table 4. The separation is starkest under CAN injection, where *Shield only* cuts ASR nearly in half relative to *Surrogate only* (0.39 vs. 0.53); by the time a takeover fires, the injected steering bias has already produced a safety-relevant deviation that the curvature cue would have counteracted within a few control cycles. Under LiDAR spoofing the same ordering holds with a smaller gap (ASR 0.15 vs. 0.24), because phantom-obstacle spoofs trigger slower trajectory responses that give a reactive surrogate more time to correct. In all three cases, the residual improvement from *Shield only* to *RAIL (full)* reflects the surrogate acting as a backstop for the small number of episodes where shielding alone is insufficient, supporting the central claim that the robustness gains in Section 5.2 come primarily from cue-conditioned shielding rather than from operator intervention.

5.5 Intrusion Risk Index Sensitivity

We address Q4 by analyzing sensitivity to the threshold, cue weights, and inter-cue correlations.

Threshold Sensitivity. Table 8 varies τ across five values. At $\tau = 0.1$, shielding activates on 42% of steps, reducing average speed (18.9 km/h) and SR (0.79) due to over-braking. At $\tau = 0.5$, activation drops to 5%, and SV rises to 1.21 as threats go unmitigated. The default $\tau = 0.3$ balances safety (SV=0.74) and performance (22.5 km/h), achieving the highest SR (0.86).

Weight Sensitivity. Table 10 compares four configurations to the default $(0.3, 0.4, 0.3)$. TTC-heavy weights $(0.2, 0.6, 0.2)$ lower SV to 0.70 but increase DR (0.0058) due to excessive braking. CURV-heavy weights $(0.5, 0.25, 0.25)$ raise SV to 0.93 because TTC is under-weighted. Uniform weights $(0.33, 0.33, 0.33)$ degrade modestly across metrics. The default achieves the highest SR (0.86) and lowest DR (0.0026), supporting greater emphasis on TTC.

Cue Correlation. Figure 5 reports pairwise Spearman correlations across test steps. All off-diagonal values are below 0.35; the highest is between r_{CURV} and r_{TTC} ($\rho = 0.34$), reflecting coupled trajectory deviation and reduced headway. Correlations involving r_{OOD} remain low ($\rho \leq 0.17$), indicating independence from control cues. These results support the conditional-independence assumption underlying Noisy-OR fusion and confirm cue complementarity.

5.6 Runtime Efficiency and Operator Burden

We finally address Q5 by quantifying computational overhead and human workload during deployment.

Runtime Overhead. Table 11 reports a micro-profile over 1,000 control steps. The full risk pipeline adds 0.215 ± 0.043 ms per step, or 0.22% of a 100 ms control cycle at 10 Hz. The dominant cost arises from the observation-shift cue (r_{OOD}) at 0.168 ± 0.037 ms due to a lightweight Mahalanobis distance on the 72-beam LiDAR vector. Noisy-OR fusion, bandit selection, and $\mathbb{R}^2$ blending each operate in the microsecond range. Overall, RAIL delivers interpretable risk estimation and adaptive shielding with negligible runtime impact, preserving real-time responsiveness.



(a) Curvature (b) TTC (c) OOD

Figure 4: **Qualitative IRI responses under dominant cues.** Each column shows two representative episodes in which curvature, TTC, or OOD dominates IRI_t. In each case, RAIL activates a cue-specific shield to avert the incident while maintaining progress.

Table 9: Shield-only vs. surrogate-only ablation under nominal and adversarial conditions on MetaDrive. Best result per metric in **bold**. Higher TSR is better; lower TSV, DRA, and ASR are better. DRA is undefined for variants without any human-style intervention and is reported as 0.00 for completeness.

	Nominal				CAN Injection				LiDAR Spoofing			
Condition	**TSR↑**	**TSV↓**	**DRA↓**	**ASR↓**	**TSR↑**	**TSV↓**	**DRA↓**	**ASR↓**	**TSR↑**	**TSV↓**	**DRA↓**	**ASR↓**
RAIL (full)	**0.85**	**0.75**	0.04	**0.08**	**0.68**	**1.02**	0.37	**0.34**	**0.80**	**0.62**	0.03	**0.11**
Shield only	0.82	0.81	**0.00**	0.12	0.64	1.18	**0.00**	0.39	0.77	0.71	**0.00**	0.15
Surrogate only	0.76	0.94	0.11	0.16	0.52	1.54	0.61	0.53	0.70	0.88	0.19	0.24
Neither (base)	0.71	1.08	**0.00**	0.21	0.41	1.87	**0.00**	0.60	0.58	1.19	**0.00**	0.31

Table 10: IRI weight configuration sensitivity on MetaDrive.

Weights (w_C, w_T, w_O)	**SR↑**	**SV↓**	**DR↓**	**AS↑**
Uniform $(0.33, 0.33, 0.33)$	0.82	0.81	0.0049	21.6
TTC-heavy $(0.2, 0.6, 0.2)$	0.84	0.70	0.0058	21.2
CURV-heavy $(0.5, 0.25, 0.25)$	0.79	0.93	0.0087	22.0
OOD-heavy $(0.25, 0.25, 0.5)$	0.81	0.83	0.0039	21.0
Default $(0.3, 0.4, 0.3)$	**0.86**	**0.74**	**0.0026**	**22.7**

Operator Burden. Table 12 summarizes workload indicators for human-in-the-loop baselines. RAIL achieves the largest mean time between takeovers (MTBT) and the lowest cumulative operator active time, indicating that risk-triggered shielding resolves many hazards before human intervention is required. This shifts the operator role from continuous supervision to rare exception handling. The false-positive rate, defined as shielded steps not followed by a safety event within 50 steps, is 14.2%, meaning 85.8% of activations align with impending hazards. This selectivity reduces workload and supports trust calibration. Formal subjective workload assessments are left to future work.

6 Insights and Practical Implications

This section distills the principal empirical insights and discusses the practical implications of the RAIL architecture. Broader extensions and future research directions are deferred to the concluding section.

Core Empirical Insights. First, explicit per-cue risk decomposition closes a gap left by implicit cost-based approaches.

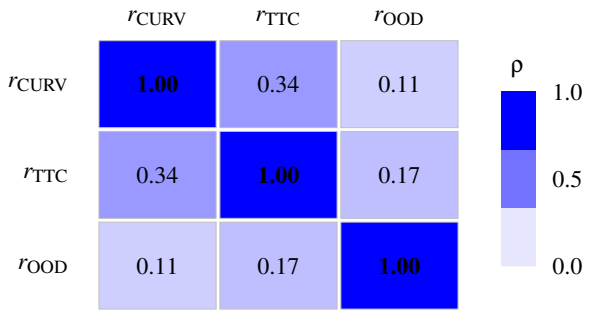


Figure 5: Pairwise Spearman correlations between IRI cues.

Table 11: Per-step computational overhead of RAIL components (mean ± std over 1,000 steps).

Component	Operation	Time (ms)
r_{CURV}	Subtract + sigmoid	0.012 ± 0.004
r_{TTC}	Divide + exp	0.010 ± 0.003
r_{OOD}	Mahalanobis ($\mathbb{R}^{72}$)	0.168 ± 0.037
IRI fusion	Noisy-OR (3 terms)	0.006 ± 0.002
Bandit selection	Softmax (3 arms)	0.014 ± 0.005
Action blending	Linear interp ($\mathbb{R}^2$)	0.005 ± 0.002
Total RAIL overhead		$\mathbf{0.215 \pm 0.043}$

While safe reinforcement learning methods reduce aggregate violations, they do not expose which failure mode triggered intervention. In contrast, RAIL attributes each shield activation to a dominant cue and threat layer, improving targeted adaptation and post-hoc auditability. Second, sensitivity analysis shows stable behavior across thresholds and weight configurations: success rate remains above 0.79 for all tested $\tau \in [0.1, 0.5]$ and weight vectors, with the default setting ($\tau = 0.3$, weights $0.3/0.4/0.3$) lying on the safety-efficiency Pareto frontier. Third, low pairwise cue correlations ($\rho \leq 0.34$) empirically support the conditional-independence assumption behind Noisy-OR fusion and validate the complementarity design principle.

Collectively, these findings demonstrate that structured, decomposed risk modeling enables simultaneous gains in safety, interpretability, and efficiency without sacrificing robustness.

Architectural and Deployment Implications. RAIL's decomposed Intrusion Risk Index provides a transparent forensic trace aligned with automotive cybersecurity documentation requirements, enabling traceable linkage between observed anomalies and mitigation actions. The cue interface is modular: new sensing or anomaly-detection modules can be integrated via monotone $[0, 1]$ mappings without redesigning the policy. The shield library is likewise composable, allowing certified low-level controllers to operate as shield arms. In this role, RAIL functions as a coordination layer over heterogeneous safety mechanisms rather than replacing them.

Table 12: Operator workload indicators on MetaDrive (test episodes).

Method	MTBT (steps)↑	Active time (s)↓	FP rate (%)↓
HACO [25]	142	8.7	–
HAIM [19]	186	6.2	–
RAIL (Ours)	**310**	**3.1**	14.2

MTBT = mean time between takeovers; Active time = cumulative operator engagement per episode; FP rate = shield activations with no safety event within 50 steps using.

Current Scope and Constraints. The present evaluation relies on a scripted human surrogate and two simulators, which capture diverse traffic and intrusion scenarios but do not encompass hardware-in-the-loop or field validation. Attack models are periodic rather than adaptive, and formal guarantees are limited to passivity, monotonicity, and bounded deviation.

Sim-to-real deployment considerations. Three deployment aspects remain open for vehicle hardware. First, real LiDAR exhibits multipath returns, weather-induced sparsity, and calibration drift that can shift the Mahalanobis baseline (μ, Σ), motivating periodic online recalibration of the perception cue. Second, the per-step overhead measured in Table 11 is small relative to a 100 ms control cycle, but real vehicle buses introduce additional CAN arbitration and sensor-fusion latencies that must be budgeted in a production timing analysis. Third, RAIL is intended to sit above existing low-level safety controllers such as automatic emergency braking and electronic stability control, not to replace them; shield arms can be instantiated with certified controllers where formal guarantees are required. These boundaries motivate the broader research agenda outlined in the conclusion.

7 Conclusions and Future Work

Building on the empirical insights and practical implications discussed above, we introduced RAIL, a risk-aware human-in-the-loop framework for autonomous vehicles operating under cyber-physical threats. RAIL fuses heterogeneous runtime cues into an Intrusion Risk Index using Noisy-OR aggregation, yielding a single yet decomposable measure of threat severity. When risk exceeds the activation threshold, a contextual bandit selects an interpretable shield, and a blending executor interpolates between nominal and safeguarded actions with authority proportional to risk. Training integrates dual reward shaping that penalizes both safety violations and unnecessary shielding, while a risk-prioritized replay buffer revisits rare but safety-critical transitions.

Across MetaDrive and CARLA and 12 baselines spanning reinforcement learning, constrained safe RL, offline learning, imitation learning, and prior human-in-the-loop methods, RAIL matches expert-level success rates (0.85 on MetaDrive and 0.41 on CARLA) while achieving the lowest test and

training safety violations using 30K interaction steps. Under Controller Area Network injection and LiDAR spoofing, RAIL reduces attack success rates by up to 48% and operator disengagements by up to 50% relative to the strongest human-in-the-loop baseline, while maintaining the highest task success rate under attack. These results show that explicit per-cue risk modeling converts threat awareness into graded, auditable control actions rather than binary fail-safes, supporting safer and more transparent human-AI collaboration.

Several directions remain for future investigation. Adaptive thresholding, online cue-weight learning, and adversarial training against white-box and persistent attackers could allow the operating point to adjust to changing traffic density, sensor conditions, and threat intensity, extending RAIL beyond the bounded-burst regime studied here. Extending the Intrusion Risk Index to multi-agent settings would enable cooperative risk sharing through compact vehicle-to-everything cue exchange. Hardware-in-the-loop validation on a drive-by-wire platform with real Controller Area Network traffic and physical sensors would further assess timing constraints and sim-to-real robustness. Finally, controlled human factors studies are needed to quantify how graded authority and selective shielding influence trust calibration and cognitive workload, complementing the log-based indicators reported here.

Ethical Considerations. RAIL is defensive in nature, aimed at improving the safety and cyber-physical resilience of autonomous vehicles. All experiments are conducted in simulation (MetaDrive and CARLA) with no real vehicles, communication buses, or human subjects. The attack models evaluated, including CAN bus injection and LiDAR spoofing, are grounded in prior automotive security research [7, 8, 27, 39] and do not introduce novel offensive techniques. While we describe attack parameterizations to support reproducibility, the underlying vulnerabilities are well documented. We believe the defensive contributions of this work, including interpretable risk decomposition, graded shielding, and reduced operator burden, outweigh any residual dual-use risk.

Acknowledgments

This work was supported by the Army Research Office under Grant No. W911NF-24-2-0241.

Use of Generative AI

Generative AI tools were used for editorial purposes in this manuscript, and all outputs were inspected by the authors to ensure accuracy and originality.

References

[1] ABDO, A., MALEK, S. M. B., ZHAO, X., AND ABU-GHAZALEH, N. Avmon: securing autonomous vehicles by learning control invariants and residual prediction. In *Symposium on Vehicle Security and Privacy (VehicleSec)* (2024).

[2] ACHIAM, J., HELD, D., TAMAR, A., AND ABBEEL, P. Constrained policy optimization. In *International conference on machine learning* (2017), PMLR, pp. 22–31.

[3] ALORAINI, F., JAVED, A., AND RANA, O. Adversarial attacks on intrusion detection systems in in-vehicle networks of connected and autonomous vehicles. *Sensors 24*, 12 (2024), 3848.

[4] ALSHIEKH, M., BLOEM, R., EHLERS, R., KÖNIGHOFER, B., NIEKUM, S., AND TOPCU, U. Safe reinforcement learning via shielding. In *Proceedings of the AAAI conference on artificial intelligence* (2018), vol. 32.

[5] AMES, A. D., XU, X., GRIZZLE, J. W., AND TABUADA, P. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control 62*, 8 (2016), 3861–3876.

[6] BAIN, M., AND SAMMUT, C. A framework for behavioural cloning. In *Machine intelligence 15* (1995), pp. 103–129.

[7] CAO, Y., XIAO, C., CYR, B., ZHOU, Y., PARK, W., RAMPAZZI, S., CHEN, Q. A., FU, K., AND MAO, Z. M. Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security* (2019), pp. 2267–2281.

[8] CHECKOWAY, S., MCCOY, D., KANTOR, B., ANDERSON, D., SHACHAM, H., SAVAGE, S., KOSCHER, K., CZESKIS, A., ROESNER, F., AND KOHNO, T. Comprehensive experimental analyses of automotive attack surfaces. In *20th USENIX security symposium (USENIX Security 11)* (2011).

[9] CHO, K.-T., AND SHIN, K. G. Fingerprinting electronic control units for vehicle intrusion detection. In *25th USENIX security symposium (USENIX Security 16)* (2016), pp. 911–927.

[10] CHOI, J., CASTAÑEDA, F., TOMLIN, C., AND SREENATH, K. Reinforcement learning for safety-critical control under model uncertainty, using control lyapunov functions and control barrier functions. *Robotics: Science and Systems XVI* (2020).

[11] CHOUDHARY, G., SHARMA, V., YOU, I., YIM, K., CHEN, R., AND CHO, J.-H. Intrusion detection systems for networked unmanned aerial vehicles: A survey. In *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)* (2018), IEEE, pp. 560–565.

[12] FISAC, J. F., AKAMETALU, A. K., ZEILINGER, M. N., KAYNAMA, S., GILLULA, J., AND TOMLIN, C. J. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control 64*, 7 (2018), 2737–2752.

[13] GETTMAN, D., AND HEAD, L. Surrogate safety measures from traffic simulation models. *Transportation research record 1840*, 1 (2003), 104–115.

[14] GREENBERG, I., CHOW, Y., GHAVAMZADEH, M., AND MANNOR, S. Efficient risk-averse reinforcement learning. *Advances in Neural Information Processing Systems 35* (2022), 32639–32652.

[15] HA, S., XU, P., TAN, Z., LEVINE, S., AND TAN, J. Learning to walk in the real world with minimal human effort. *arXiv preprint arXiv:2002.08550* (2020).

[16] HAARNOJA, T., ZHOU, A., ABBEEL, P., AND LEVINE, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning* (2018), Pmlr, pp. 1861–1870.

[17] HAMAD, M., FINKENZELLER, A., KÜHR, M., ROBERTS, A., MAENNEL, O., PREVELAKIS, V., AND STEINHORST, S. React: Autonomous intrusion response system for intelligent vehicles. *Computers & Security 145* (2024), 104008.

[18] HO, J., AND ERMON, S. Generative adversarial imitation learning. *Advances in neural information processing systems 29* (2016).

[19] HUANG, Z., SHENG, Z., MA, C., AND CHEN, S. Human as ai mentor: Enhanced human-in-the-loop reinforcement learning for safe and efficient autonomous driving. *Communications in Transportation Research 4* (2024), 100127.

[20] ISO/SAE 21434, C. Road vehicles—cyber security engineering, 2021.

[21] JOHANSEN, I. L., AND RAUSAND, M. Foundations and choice of risk metrics. *Safety science 62* (2014), 386–399.

[22] KASTNER, T., ERDOGDU, M. A., AND FARAHMAND, A.-M. Distributional model equivalence for risk-sensitive reinforcement learning. *Advances in Neural Information Processing Systems 36* (2023), 56531–56552.

[23] KELLY, M., SIDRANE, C., DRIGGS-CAMPBELL, K., AND KOCHENDERFER, M. J. Hg-dagger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)* (2019), IEEE, pp. 8077–8083.

[24] KUMAR, A., ZHOU, A., TUCKER, G., AND LEVINE, S. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems 33* (2020), 1179–1191.

[25] LI, Q., PENG, Z., AND ZHOU, B. Efficient learning of safe driving policy via human-ai copilot optimization. *arXiv preprint arXiv:2202.10341* (2022).

[26] MANDLEKAR, A., XU, D., MARTÍN-MARTÍN, R., ZHU, Y., FEI-FEI, L., AND SAVARESE, S. Human-in-the-loop imitation learning using remote teleoperation. *arXiv preprint arXiv:2012.06733* (2020).

[27] MILLER, C., AND VALASEK, C. Remote exploitation of an unaltered passenger vehicle. *Black Hat USA 2015*, S 91 (2015), 1–91.

[28] NAGARAJAN, J., MANSOURIAN, P., SHAHID, M. A., JAEKEL, A., SAINI, I., ZHANG, N., AND KNEPPERS, M. Machine learning based intrusion detection systems for connected autonomous vehicles: A survey. *Peer-to-Peer Networking and Applications 16*, 5 (2023), 2153–2185.

[29] NO, U. R. 155-cyber security and cyber security management system. *The United Nations Economic Commission for Europe (UNECE)* (2021).

[30] PAREKH, D., PODDAR, N., RAJPURKAR, A., CHAHAL, M., KUMAR, N., JOSHI, G. P., AND CHO, W. A review on autonomous vehicles: Progress, methods and challenges. *Electronics 11*, 14 (2022), 2162.

[31] PASQUALETTI, F., DÖRFLER, F., AND BULLO, F. Attack detection and identification in cyber-physical systems. *IEEE transactions on automatic control 58*, 11 (2013), 2715–2729.

[32] PEARL, J. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier, 2014.

[33] PENDLETON, M., GARCIA-LEBRON, R., CHO, J.-H., AND XU, S. A survey on systems security metrics. *ACM Computing Surveys (CSUR) 49*, 4 (2016), 1–35.

[34] QUEENEY, J., AND BENOSMAN, M. Risk-averse model uncertainty for distributionally robust safe reinforcement learning. *Advances in Neural Information Processing Systems 36* (2023), 1659–1680.

[35] REN, H., AND HUANG, T. Adversarial example attacks in the physical world. In *International Conference on Machine Learning for Cyber Security* (2020), Springer, pp. 572–582.

[36] SAUNDERS, W., SASTRY, G., STUHLMUELLER, A., AND EVANS, O. Trial without error: Towards safe reinforcement learning via human intervention. *arXiv preprint arXiv:1707.05173* (2017).

[37] SCHULMAN, J., WOLSKI, F., DHARIWAL, P., RADFORD, A., AND KLIMOV, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[38] SHA, L. Using simplicity to control complexity. *IEEE Software 18*, 4 (2001), 20.

[39] SHIN, H., KIM, D., KWON, Y., AND KIM, Y. Illusion and dazzle: Adversarial optical channel exploits against lidars for automotive applications. In *International conference on cryptographic hardware and embedded systems* (2017), Springer, pp. 445–467.

[40] STOOKE, A., ACHIAM, J., AND ABBEEL, P. Responsive safety in reinforcement learning by pid lagrangian methods. In *International Conference on Machine Learning* (2020), PMLR, pp. 9133–9143.

[41] SULTANA, R., GROVER, J., TRIPATHI, M., AND SHARMA, P. Ladetects: Local and adaptive data-centric misbehavior detection framework for vehicular technology security. *IEEE Open Journal of Vehicular Technology* (2024).

[42] TAMAR, A., CHOW, Y., GHAVAMZADEH, M., AND MANNOR, S. Policy gradient for coherent risk measures. *Advances in neural information processing systems 28* (2015).

[43] TAMAR, A., GLASSNER, Y., AND MANNOR, S. Optimizing the cvar via sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2015), vol. 29.

[44] YANG, Z., JIN, H., TANG, Y., AND FAN, G. Risk-aware constrained reinforcement learning with non-stationary policies. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems* (2024), pp. 2029–2037.





USENIX VehicleSec '26 Artifact Appendix: Risk-Aware Human-in-the-Loop Framework with Adaptive Intrusion Response for Autonomous Vehicles

Dawood Wasif[1], Terrence J. Moore[2], Seunghyun Yoon[3], Hyuk Lim[3], Dan Dongseong Kim[4], Frederica F. Nelson[2], Jin-Hee Cho[1]

[1]Virginia Tech [2]U.S. Army Research Laboratory [3]Korea Institute of Energy Technology
[4]University of Queensland

A Artifact Appendix

A.1 Abstract

This artifact provides the complete implementation of RAIL, a risk-aware human-in-the-loop framework for autonomous vehicles that unifies runtime cyber–physical risk estimation, adaptive shielding, action arbitration, and risk-prioritized learning. It contains the RAIL framework (the IRI risk score via weighted Noisy-OR cue fusion, a contextual bandit for shield selection, a risk-conditioned action-blending executor, and a risk-prioritized replay buffer), the `RAILEnv` wrapper with CAN-injection and LiDAR-spoofing attacks, all comparison baselines, and end-to-end scripts. It supports five paper claims (C1–C5; see Major Claims).

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

The artifact is defensive and poses no security or privacy risk to evaluators. All experiments run entirely in simulation (MetaDrive and CARLA 0.9.9.4); no real vehicles, physical CAN buses, hardware sensors, or human subjects are involved. The CAN-injection and LiDAR-spoofing attacks reproduce previously documented automotive cyber–physical scenarios as parameterized simulator perturbations and add no novel offensive capability. Human demonstration data for some baselines was collected by a co-author via keyboard control in simulation only. No personal data is collected or processed.

A.2.2 How to access

The artifact is permanently archived on Zenodo with a stable DOI: **10.5281/zenodo.20451598** (`https://doi.org/10.5281/zenodo.20451598`).

A.2.3 Badges

This artifact was awarded the *Artifacts Available* badge (source code, configuration files, and the human-demonstration dataset publicly archived on Zenodo with a stable DOI) and the *Artifacts Functional* badge (all major components described in the paper, with documentation and end-to-end scripts that run the reported experiments).

A.2.4 Hardware dependencies

A modern x86-64 CPU ($\geq$4 cores, 8+ recommended for Ray parallelism); $\geq$16 GB RAM (32 GB with CARLA); one NVIDIA GPU with $\geq$8 GB VRAM (all results obtained on an RTX 3090, 24 GB; `--num-gpus=0` allows slow CPU-only runs); $\geq$20 GB disk (plus 15–30 GB for CARLA); and a connected display or virtual framebuffer (`Xvfb`) for rendered runs (set `use_render: False` for headless training). Training takes $\sim$2.5 h per method (MetaDrive, 30 K budget) and $\sim$3 h (CARLA) on one RTX 3090.

A.2.5 Software dependencies

Ubuntu 20.04/22.04 and Python 3.7, managed with Conda. Core libraries: `metadrive-simulator==0.2.4`, `ray==1.0.0`, `tensorflow==2.4.0` (with `tensorflow-probability==0.11.1`), `gym==0.19.0`, the standard SciPy-stack packages, and CUDA 11.0 with cuDNN. The CARLA experiment (Table 5) additionally needs the CARLA 0.9.9.4 server and `di-engine==0.2.0`; the GAIL/HG-DAgger/IWR baselines need `pytorch==1.7.1` in a separate Conda environment.

A.2.6 Benchmarks

All experiments are simulation-only. MetaDrive 0.2.4 uses procedurally generated maps (50 training maps, seeds 100–149; 50 held-out maps, seeds 500–549); CARLA uses Town02/Town05 routes via DI-Drive. No external dataset is required for generation. A human-demonstration

dataset (`data/human_demos.pkl`, 49 K steps collected in simulation) is included in the Zenodo archive for the imitation/offline-RL baselines and is regenerable via the provided script.

A.3 Set-up

A.3.1 Installation

Obtain the code from the Zenodo DOI, then create the MetaDrive environment:

```
conda create -n RAIL python=3.7 -y
conda activate RAIL
pip install -e .
conda install cudatoolkit=11.0 -y
conda install -c nvidia cudnn -y
```

Optionally create two more environments (same `python=3.7`, `cudatoolkit=11.0`): `RAIL-carla` (add `di-engine==0.2.0`, `markupsafe==2.0.1`) for the CARLA experiment, and `RAIL-torch` (add `pytorch==1.7.1`) for the GAIL/HG-DAgger/IWR baselines.

A.3.2 Basic Test

Verify the installation and environment wrapper with a short quick-start run:

```
conda activate RAIL
cd RAIL/scripts/run_rail
python train_rail_keyboard_easy.py \
  --num-gpus=1
```

Expected: Ray starts, MetaDrive launches, and runtime metrics such as `success`, `takeover`, and `disturbance` print to the console; $\sim$5000 steps finish in under 15 minutes on a single GPU.

A.4 Evaluation workflow

A.4.1 Major Claims

(C1): *Nominal MetaDrive performance.* RAIL reaches expert-level success while attaining the lowest safety and training-time violations at the 30 K interaction budget. Proven by (E1); Table 4.

(C2): *CARLA generalization.* RAIL transfers to CARLA with TSR $= 0.41$ and TR $= 1609.70$ at only 8 K effective steps. Proven by (E2); Table 5.

(C3): *Adversarial robustness.* Under CAN injection and LiDAR spoofing, RAIL attains the lowest attack-success and disengagement metrics. Proven by (E3); Table 6.

(C4): *Complementary components.* Removing any component (a cue, the bandit, cue weighting, or shield-effort regularization) degrades $\geq$1 key metric. Proven by (E4); Table 7.

(C5): *Sensitivity stability.* Performance is stable across IRI thresholds and cue weights, with the default on the safety–efficiency frontier. Proven by (E5); Tables 8–9.

A.4.2 Experiments

(E1): *Nominal benchmarking* [$\sim$2.5 h/method]: run `train_rail.py` in `scripts/run_rail/`, plus the baseline `train_*.py` scripts in `run_haim/` and `run_baselines/` (HAIM-DRL, HACO, SAC/PPO $\pm$Lagrangian, CQL with `ray==1.2.0`, and GAIL/HG-DAgger/IWR in `RAIL-torch`). Matches Table 4 within $\pm$0.05 TSR / $\pm$0.10 TR over 5 seeds, ordering preserved (validates C1).

(E2): *CARLA generalization* [$\sim$3 h]: start `./CarlaUE4.sh -carla-rpc-port=9000`, then in `RAIL-carla` run `train_rail_in_carla.py` (baselines: `train_haim_drl_in_carla.py`, `train_haco_in_carla.py`). TSR within $\pm$0.03, TR within $\pm$50 of Table 5 (validates C2).

(E3): *Adversarial robustness*: train with `train_rail.py --attack-mode=cyber` (CAN) and `=physical` (LiDAR); evaluate via `eval_rail.py --ckpt-idx 200 --num-episodes 50 --attack-mode {cyber,physical}`. ASR/DRA within $\pm$0.05 of Table 6 (validates C3).

(E4): *Ablation*: in the `train_rail.py` config, zero a cue (`iri_w_ttc`/`iri_w_curv`/`iri_w_ood`=0), use a fixed dominant-cue arm (no bandit), set all weights to 0.333, or set `lambda5`=0. Each degrades $\geq$1 metric (most when TTC is dropped), per Table 7 (validates C4).

(E5): *Sensitivity*: sweep `iri_tau`$\in \{0.1, \ldots, 0.5\}$ and the cue weights per Table 9, re-running `train_rail.py` after each edit. SR stays $>$0.79; default near the best trade-off (validates C5).

A.5 Notes on Reusability

RAIL is modular: add risk cues in `scripts/utils/iri_calculator.py` (return $[0,1]$ values and register them in `IRICalculator.compute()`); add shield actions in `shield_bandit.py`; reuse `RAILEnv` in environments exposing curvature mismatch, obstacle-closing risk, and LiDAR-like ranges; and pair the runtime safety layer with other RL backbones, since the wrapper and replay buffer are learner-independent. All major parameters are configuration-exposed for systematic sweeps.

A.6 Version

Based on the LaTeX template for Artifact Evaluation V20260101. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at https://secartifacts.github.io/vehiclesec2026/.