computers

MDPI

Article

MobiTrust: Risk-Bounded Adaptive Split Inference Under Mobility, Privacy Leakage, and Latent-Space Attacks

Salam Al-E'mari [1], Yousef Sanjalawe [2,*], Muder Almiani [3] and Mohammad A. Alsharaiah [2]

[1] Department of Information Security, Faculty of Information Technology, University of Petra (UoP), Amman 11196, Jordan; salam.ammari@uop.edu.jo
[2] Department of Information Technology, King Abdullah II School for Information Technology, University of Jordan (JU), Amman 11942, Jordan; m.alsharaiah@ju.edu.jo
[3] MIS Department, Gulf University for Science and Technology, Hawally 32093, Kuwait; almiani.m@gust.edu.kw
* Correspondence: y.sanjalawe@ju.edu.jo

Abstract

Split inference lets a mobile device run the first layers of a neural network locally and hand the intermediate representation to an edge server, trading raw data disclosure for communication and offloaded computation. The common assumption that keeping raw inputs on the device is enough for privacy does not survive contact with modern attacks: intermediate activations can be inverted back to inputs, tested for membership, or probed for sensitive attributes, and they can be tampered with in transit. These risks are not fixed properties of a network; they shift with the split point, with any protection applied to the representation, with the trust placed in the serving edge, and with the sensitive attributes at stake in a given context. Schedulers that pick a split to minimise latency or energy, or that fold risk into a single weighted objective, routinely cross privacy and integrity limits whenever the network briefly makes offloading attractive. We present MobiTrust, an adaptive split-inference controller that treats privacy and latent-integrity risk as explicit constraints on measured risk scores rather than terms to be traded away. Two measurable quantities drive it: Context-Dependent Semantic Exposure (CDSE), a worst-channel estimate of privacy leakage (assembled from reconstruction, shadow-model membership, and attribute attacks) that accounts for the sensitive attributes a deployment cares about, and Latent Integrity Risk (LIR), a detector score for tampered representations. A feasibility-filtering controller selects, at each inference, the split point, the representation protection, and the serving edge that minimise cost subject to per-inference risk and deadline budgets, and it falls back to on-device execution when no offload is safe. We frame this as a controlled simulation study of the constraint mechanism: it runs on a reduced-scale, fully reproducible testbed with an image and a sensing workload and a transparent analytical latency/energy model, so the claims concern the controller's decisions and their effect on measured risk, not absolute timings on a particular device. CDSE tracks measured attack success far better than the split-depth and tensor-size heuristics used by leakage-agnostic schedulers (Spearman ρ of 0.59 and 0.87 versus 0.33 and 0.72 for depth). Over mixed mobility traces, MobiTrust holds the profiled privacy and adversarial constraint violations at 0% while the latency-only and weighted-sum baselines violate the privacy budget in 88% of inferences and the integrity budget in up to 58%; a weight sweep shows that, across the weightings we tested, the weighted-sum baseline reaches zero violations only by ceasing to offload. The cost of the guarantee is energy: MobiTrust saves device energy by offloading when it is safe, but a zero-violation local-only baseline attains higher accuracy and lower tail latency, so the improvement is a favourable energy trade-off rather than a uniform win. Because zero violations certify compliance with the profiled risk

Check for updates

Academic Editor: Paolo Bellavista

Received: 18 September 2026
Revised: 8 October 2026
Accepted: 9 October 2026
Published: 14 October 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

scores rather than an independently validated security bound, we also show that, under estimation uncertainty, the controller's primal–dual margin holds the realised violation rate at its tolerance.

Keywords: split inference; edge intelligence; privacy leakage; adversarial robustness; constrained optimisation; mobility; trustworthy machine learning

1. Introduction

A phone that recognises a scene, a wearable that classifies an activity, a vehicle that reads its surroundings, each of these runs a neural network under constraints that a data-centre model never faces. Battery, thermal envelope, and a radio link whose quality changes as the user moves all push against the wish to run large, accurate models. Split inference is one of the more practical answers. The network is cut at some layer; the device evaluates the early part and sends the resulting activations to an edge server, which finishes the computation and returns a prediction [1,2]. Because the raw input never leaves the device, split inference is often presented as the privacy-preserving option, and because most of the arithmetic can move to the edge, it promises lower latency and energy than running everything locally.

Both halves of that pitch deserve scrutiny. The privacy half rests on an intuition, raw data stays home, that a decade of attacks has undermined. An honest-but-curious edge that receives intermediate activations can train a decoder to invert them and recover a recognisable version of the input, can test whether a particular example was in the training set, and can probe the representation for attributes the user never meant to share, such as identity or demographics [1,3,4]. Split learning, the training-time cousin of split inference, leaks in similar ways [5–7]. The performance half is real but conditional: offloading only helps when the link can carry the activations within the deadline, and the very act of offloading opens a second front, because an adversary on the path or a compromised edge can perturb or replace the representation and steer the prediction. None of these risks is a constant. Cut the network deeper and reconstruction gets harder but the device does more work; add noise or quantisation and leakage falls but so does accuracy; route to a trusted edge and integrity risk drops but that edge may be slower or farther away. Which trade-off is acceptable depends on what a given inference is exposing and to whom.

1.1. The Gap

Adaptive split-inference schedulers have largely optimised for the resource side of this picture. A controller watches bandwidth and battery and chooses a split point, or a partition across several users, to minimise latency or energy [8–11]. When privacy or security enter at all, they usually enter as one more weighted term in a scalar objective. That framing has a quiet but serious failure mode: a weight that looks balanced on average will still sacrifice privacy in any individual inference where the latency term happens to dominate, exactly the good-connectivity moments when offloading a leaky representation is most tempting. A user is not protected by a scheduler that respects their privacy on average; they are exposed in precisely the instants the average hides. We show this concretely in Section 6: a carefully tuned weighted-sum controller violates a privacy budget in 88% of inferences on our testbed even though its weights nominally price privacy as highly as latency.

Two further gaps compound the first. Schedulers that do consider leakage tend to use a structural proxy for it: deeper splits are assumed safer, or smaller tensors are assumed to leak less, without measuring whether the proxy holds once protection mechanisms are in play. It does not: a shallow split with strong noise can leak less than a deep split with none,

so a depth-ordered heuristic misranks the options; and almost no adaptive scheduler treats the integrity of the transmitted representation as a first-class risk with its own budget, even though a tampered activation is as damaging as a leaked one.

1.2. Contributions

MobiTrust is built around a simple stance: privacy and integrity are limits to be respected on every inference, not quantities to be averaged against speed. To make that stance operational, we contribute the following.

- Context-Dependent Semantic Exposure (CDSE), a privacy-risk estimate that combines reconstruction, membership, and attribute-inference leakage into a single worst-channel score, weighted by the sensitive attributes a deployment declares it cares about. The membership channel is measured with a shadow-model attack, which we validate recovers a strong signal on a deliberately overfit model, so a low score reflects a genuinely weak leak rather than a weak adversary. On our testbed, CDSE ranks operating conditions by measured attack success with Spearman $\rho = 0.59$ (image) and 0.87 (sensing), against 0.33 and 0.72 for a split-depth heuristic and essentially zero for tensor size (Section 6, Table 7).
- Latent Integrity Risk (LIR), a lightweight detector combining a class-conditional Mahalanobis score with a reconstruction-error term, together with a route-trust model that turns the choice of serving edge into a bounded integrity risk. We characterise what the detector catches and, just as importantly, what it misses, across four distinct detector-aware adaptive attacks, and we show the route-trust term can be estimated from observed edge behaviour rather than hand-assigned.
- A risk-bounded controller that, at each inference, selects the split point, representation protection, and serving edge minimising a latency–energy cost subject to explicit CDSE, LIR, utility, and deadline budgets, with a safe on-device fallback when no offload is feasible. In the nominal case with exact, state-independent risk scores, this online rule is a per-inference feasibility filter with a deadline-adaptive heuristic; the Lagrangian primal–dual machinery it is derived from becomes load-bearing under estimation uncertainty, where a dual-driven safety margin holds the realised violation rate at its tolerance (Section 6.9). We are explicit about this distinction rather than overclaiming an always-active dual.
- A fully reproducible evaluation on an image workload and a sensing workload, comparing against seven baselines including a clairvoyant oracle, with bootstrap confidence intervals, paired significance tests with multiple-comparison correction, effect sizes, a component ablation, and sensitivity sweeps over the risk budgets, attacker strength, and the cost-model parameters themselves. Every number in the paper is produced by released code.

We frame the work as a controlled simulation study of the constraint mechanism rather than a mobile-deployment system paper. The results below come from a reduced-scale testbed run entirely on CPU, with a transparent analytical model for latency and energy rather than measurements on phones, single-board computers, or a live radio. This choice buys full reproducibility and lets us isolate the behaviour of the constraint machinery, and it is the right vehicle for the claims we actually make, which are about the controller's decisions and their effect on measured risk, not about absolute milliseconds on a particular device. A cost-model sensitivity study (Section 6.10) confirms that the qualitative conclusions do not depend on the specific cost constants. Section 9.2 states the limitations plainly, and the released pipeline is structured so that the cost model can be swapped for hardware traces without touching the risk logic.

The rest of the paper is organised as follows:

Section 2 reviews adaptive split inference, representation-level privacy and integrity attacks, and constrained control for edge systems. Section 3 sets up the system and threat model and states the constrained optimisation problem. Section 4 develops CDSE, LIR, and the primal–dual controller, with algorithms and a complexity analysis. Sections 5 and 6 give the experimental setup and the main results and comparative analysis. Sections 7 and 8 report the ablation and the sensitivity study. Section 9 discusses implications, limitations, and threats to validity, and Section 10 concludes.

2. Related Work

MobiTrust sits at the meeting point of three lines of work: adaptive split and collaborative inference at the edge, privacy and integrity attacks on intermediate representations, and constrained control for resource-limited systems. We take each in turn and then state what, to our reading, remains open.

2.1. Adaptive Split and Collaborative Inference

The idea of cutting a network between device and server goes back to early work on private feature extraction, which already noticed that the split point governs both cost and how much the shared features reveal [1]. Since then, the offloading question has been studied mostly as a resource-allocation problem. Xu et al. [8] schedule DNN inference across mobile edge clouds to minimise energy under latency constraints; Tang et al. [9] jointly choose split points and computational resources for several users sharing an edge; and a large body of reinforcement learning work places services and allocates compute dynamically as conditions change [10,12]. Recent systems extend the idea to large models and heterogeneous edges, sharding transformer inference across devices to fit memory and latency budgets [11,13]. Mobility makes the problem harder still, because the link and the reachable edges change during a session; vehicular and aerial settings have driven work on joint offloading and resource management under motion [14–16]. A recurring theme in this literature is a scalar objective, some weighted combination of latency, energy, and occasionally accuracy, optimised by a learned or heuristic policy. Privacy and representation integrity are, with few exceptions, absent from the objective, and when present, they appear as an additional weighted penalty rather than as a constraint. Communication-efficient split learning has also pursued smaller transmitted representations through learned or adaptive compression [6,17], and split learning over wireless links has been studied as a joint parallel-design and resource-management problem [18], which reduces cost but does not by itself bound leakage.

2.2. Attacks on Intermediate Representations

The privacy case against "raw data never leaves the device" has been made repeatedly and from several angles. Model-inversion and reconstruction attacks train a decoder to map activations back to inputs, and they succeed on the shallow splits that offloading favours [1,3,19]. Membership inference asks whether a specific record was used in training and remains a live threat for models deployed on devices, including under incremental updates [4,20]. Attribute inference recovers sensitive properties, identity, demographics, that the task never required. Defences reduce but do not remove these leaks: differential privacy applied to the representation trades accuracy for protection [3,21], and generative or domain-adaptation defences raise the attacker's cost at some utility penalty [4]. Split learning has its own attack surface, including label-inference attacks that exploit the gradients or smashed data exchanged during training [5,6], and privacy concerns carry into mobile and vehicular deployments where split and federated variants are combined [22]; a recent survey documents how persistent these leaks are across split-learning variants [7].

Two points from this literature shape our design. First, leakage is multi-channel: reconstruction, membership, and attribute attacks stress different weaknesses, so a single scalar tied to one of them understates the risk. Second, protection and split depth interact, so any leakage estimate that reads off depth alone will misrank protected options, an effect we measure directly in Section 6.

Integrity has received less attention in the split-inference setting than privacy, yet a representation in flight can be perturbed or replaced. Adversarial perturbations crafted in the latent space can flip predictions, and a compromised or spoofed edge can substitute activations wholesale. Detecting anomalous representations is related to out-of-distribution detection, and simple class-conditional distance scores are a reasonable first line; we adopt one and report candidly where it fails, in particular against replayed in-distribution representations.

2.3. Constrained Control for Edge Systems

Treating limits as constraints rather than penalties has a mature theory. Constrained Markov decision processes formalise sequential decision-making subject to long-run cost budgets [23], and Lagrangian methods turn such problems into saddle-point computations that primal–dual updates can track online. In wireless and edge resource allocation, learning constrained policies in this way is by now standard [24–26]. MobiTrust borrows this machinery but points it at a different target. Where prior constrained formulations bound latency, energy, or reliability, we bound privacy exposure and representation integrity, using online multipliers to hold chance constraints over a mobility trace and adding a feasibility filter plus a safe fallback so that the per-inference guarantees do not depend on the multipliers having converged.

2.4. What Remains Open

Pulling these threads together, three gaps motivate this work. (i) Adaptive schedulers optimise resources with a scalar objective, so they give up privacy in the individual inferences where speed dominates, even when the objective looks balanced on average. (ii) The leakage proxies that appear, split depth, tensor size, are structural and do not track measured attack success once protection is applied. (iii) Representation integrity is rarely a budgeted risk, and the choice of serving edge is rarely tied to it. MobiTrust addresses all three by making CDSE and LIR measurable, budgeting them explicitly, and selecting split, protection, and edge jointly under those budgets.

Table 1 places representative prior work against the capabilities this requires. The pattern is consistent: resource-oriented schedulers cover mobility and cost well but treat privacy, if at all, as a soft term; attack and defence studies quantify leakage but do not schedule under it; and constrained-control methods bound resources rather than privacy or integrity. MobiTrust is the combination, measured, multi-channel privacy and integrity risk, each held as a per-inference constraint with a safe fallback, under mobility.

Table 1. Where representative prior work sits against the capabilities MobiTrust targets. ✓: supported; ●: partial or soft (e.g., risk as a weighted penalty, or a structural leakage proxy); ✗: not addressed. "Multi-channel leakage" means reconstruction, membership, and attribute exposure are all measured; "per-inference guarantee" means risk limits hold each inference rather than on average.

	Adaptive Split	Mobility-Aware	Privacy Measured	Multi-Channel	Integrity Budget	Per-Inference	Safe Fallback
Resource offloading [8,9]	✓	●	✗	✗	✗	✗	✗
DRL edge control [10,25]	✓	✓	✗	✗	✗	✗	✗
Collaborative LLM/edge [11,13]	✓	●	✗	✗	✗	✗	✗
Vehicular/aerial MEC [14,15]	●	✓	✗	✗	✗	✗	✗

Table 1. *Cont.*

	Adaptive Split	Mobility-Aware	Privacy Measured	Multi-Channel	Integrity Budget	Per-Inference	Safe Fallback
Split-learning privacy [5,7]	●	✗	✓	●	✗	✗	✗
Representation defences [3,21]	✗	✗	✓	●	✗	✗	✗
Constrained resource control [23,24]	✗	●	✗	✗	✗	●	✗
MobiTrust (this work)	✓	✓	✓	✓	✓	✓	✓

3. Problem Formulation

Figure 1 sketches the setting: a device evaluates the first layers of a network, optionally protects the resulting representation, and offloads it over a changing link to one of several edges of differing trust, which returns a prediction. Each choice the device makes, where to split, how to protect, which edge to use, moves both the cost and the risk, and it must be made under budgets that hold on every inference. The rest of this section makes that precise.

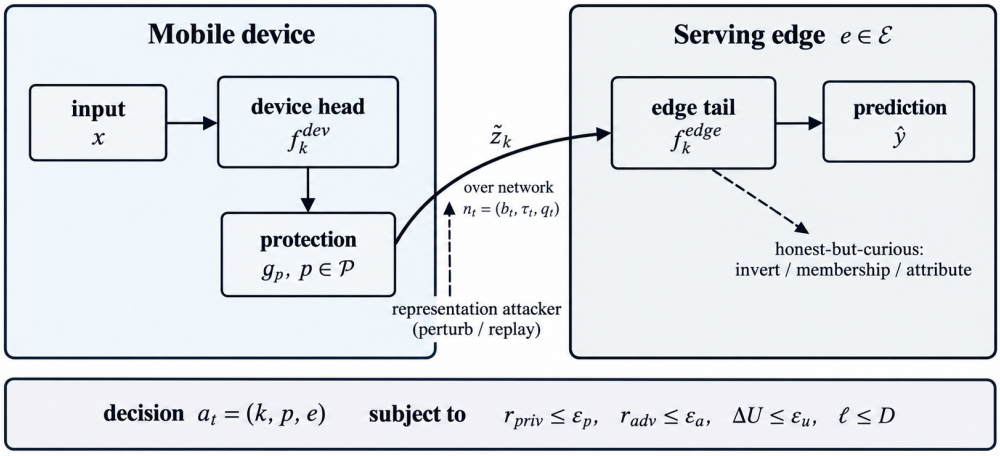


Figure 1. System and threat model. The device computes $z_k = f_k^{dev}(x)$, applies a protection g_p, and transmits $\tilde{z}_k$ over a time-varying network to a serving edge e, which runs the tail f_k^{edge} and returns $\hat{y}$. An honest-but-curious edge mounts offline privacy attacks on $\tilde{z}_k$; a representation attacker on the path may perturb or replay it. At each inference, the controller chooses $a_t = (k, p, e)$ subject to the privacy, integrity, utility, and deadline budgets shown.

3.1. System Model

A device holds a trained classifier $f : \mathcal{X} \to \mathcal{Y}$ organised as a stack of L layers, $f = h_L \circ \cdots \circ h_1$. A split point $k \in \{0, 1, \ldots, L\}$ divides the network into a device-side head $f_k^{\text{dev}} = h_k \circ \cdots \circ h_1$ and an edge-side tail $f_k^{\text{edge}} = h_L \circ \cdots \circ h_{k+1}$. For an input x, the device computes the representation $z_k = f_k^{\text{dev}}(x)$ and, if it offloads, transmits z_k to a serving edge that returns $\hat{y} = f_k^{\text{edge}}(z_k)$. Two boundary cases matter: $k = 0$ transmits the raw input (full offload), and $k = L$ keeps everything on device (local execution). Before transmission, the device may apply a protection map g_p from a set $\mathcal{P}$ of modes, identity, quantisation, additive noise, a differential-privacy mechanism, or a learned bottleneck, producing $\tilde{z}_k = g_p(z_k)$ at some utility cost. It may also choose among a set $\mathcal{E}$ of reachable edges of differing trust, for example a trusted operator edge or an opportunistic untrusted one.

Inference happens repeatedly as the user moves. At epoch t, the network state is $n_t = (b_t, \tau_t, q_t)$: available uplink bandwidth b_t, round-trip time τ_t, and edge queueing delay q_t. These evolve according to a mobility regime, stationary, walking, vehicular, during a handover, or under sudden congestion, that the controller does not observe directly and must infer from the recent trace. A decision at epoch t is a tuple $a_t = (k_t, p_t, e_t)$ choosing

split, protection, and edge (with a distinguished local action), drawn from a finite action set $\mathcal{A}$ built from $\{0, \ldots, L\} \times \mathcal{P} \times \mathcal{E}$ plus the local option.

3.2. Costs and Risks of an Action

Each action induces a latency and an energy cost that depend on the network state, and a set of risks that depend on the action and the deployment context. We write $\ell(a, n)$ for end-to-end latency, $\eta(a, n)$ for device energy, and collect the risks as follows.

Privacy exposure $r_{\text{priv}}(a, c) \in [0, 1]$ measures how much a curious edge can learn about the input from $\tilde{z}_k$, given a context c that declares which sensitive attributes matter. It aggregates three leakage channels, reconstruction, membership, and attribute inference, and is developed in Section 4.2 as CDSE.

Integrity risk $r_{\text{adv}}(a) \in [0, 1]$ measures the chance that the returned prediction has been corrupted, whether by perturbation of $\tilde{z}_k$ in transit or by a compromised edge. It combines the route trust of the chosen edge with the protection afforded by the latent-integrity detector at split k (Section 4.3).

Utility loss $\Delta U(a) \geq 0$ is the drop in task accuracy that protection introduces relative to the unprotected full model.

3.3. Threat Model

We consider three adversaries, consistent with the attack literature reviewed in Section 2, and state what each can and cannot do.

An honest-but-curious edge executes the tail correctly but stores everything it receives and mounts offline privacy attacks: it may train reconstruction decoders, membership classifiers, and attribute probes on auxiliary data of the same modality. It does not deviate from the protocol. This adversary is the target of the privacy budget.

A representation attacker modifies $\tilde{z}_k$ before it reaches the tail, using perturbations crafted against the latent space, or replays a previously seen valid representation. We distinguish two sub-cases because they are defended differently. (i) An external on-path attacker does not hold the authentication key; it can inject or alter packets but cannot produce a valid tag. (ii) A compromised or untrusted serving edge is the legitimate recipient of $\tilde{z}_k$ and computes the tail, so it can tamper with the representation it actually feeds to f_k^{edge} or return a manipulated $\hat{y}$. This adversary is the target of the integrity budget and the route-trust model.

A network observer sees packet timing and sizes but not plaintext contents; we assume the link is encrypted, so this adversary motivates limiting transmitted volume but is not otherwise modelled.

Trust Assumptions and Component Placement

The device and the trusted edge share a message-authentication key κ and are assumed uncompromised; model weights are public (white-box), the conservative assumption for both inversion and latent perturbation. The authentication-and-freshness check (Section 4.3, Equation (6)) is verified by the recipient that holds κ, i.e., the device–trusted-edge channel. The LIR detector runs at the party that consumes the representation and is trusted to act on its verdict: at the trusted edge for the trusted-edge channel, or on a device-side/trusted-monitor check of returned computations when an untrusted edge is used. The two mechanisms therefore cover different adversaries. Against the external on-path attacker, the tag over $(\tilde{z}_k, \nu, c)$ already rejects any modification, perturbation or replay, since the attacker cannot forge a fresh tag; LIR adds nothing there. LIR earns its place against the untrusted/compromised edge, where no shared key constrains the recipient and the route-trust risk r_{adv} budgets the residual exposure. The perturbation-detection experiments in Section 6 should therefore be read as the no-authentication/untrusted-edge setting,

which is where a distance-based detector is the operative defense; on an authenticated trusted-edge channel, those perturbations would already be rejected by the tag.

3.4. Constrained Optimisation

The device operator specifies per-inference budgets: a privacy budget ε_p, an integrity budget ε_a, a utility budget ε_u, and a latency deadline D, together with tolerated exceedance probabilities δ_{C} for each constraint $\mathrm{C} \in \{p, a, u, \mathrm{T}\}$. These tolerances define long-run empirical violation-rate requirements over the trace. The goal at each epoch t is to select the feasible action with minimum operating cost while maintaining each constraint's violation rate below its prescribed tolerance.

The scalar operating cost is defined as $\phi(a, n_t) = \ell(a, n_t) + \lambda_e \eta(a, n_t)$, where λ_e denotes the energy weight. The per-epoch optimization problem is formulated as:

$$\begin{aligned} a_t^\star &= \arg\min_{a \in \mathcal{A}} \phi(a, n_t) \\ \text{s.t.} \quad & r_{\mathrm{priv}}(a, c_t) \le \varepsilon_p, \qquad r_{\mathrm{adv}}(a) \le \varepsilon_a, \\ & \Delta U(a) \le \varepsilon_u, \qquad \ell(a, n_t) \le D. \end{aligned} \tag{1}$$

Over a trace of length T, the additional requirement for each constraint $\mathrm{C} \in \{p, a, u, \mathrm{T}\}$ is

$$\frac{1}{T} \sum_{t=1}^{T} \mathbb{I}[\text{constraint C is violated at epoch } t] \le \delta_{\mathrm{C}}. \tag{2}$$

Here, $a = (k, p, e)$ denotes an action selected from the finite action set $\mathcal{A}$; $a_t^\star$ is the optimal action at epoch t; $\phi(a, n_t)$ is the scalar operating cost; $\ell(a, n_t)$ is the end-to-end latency; $\eta(a, n_t)$ is the device energy consumption; and $\lambda_e \ge 0$ is the energy weight.

Furthermore, n_t denotes the network state at epoch t; c_t is the corresponding context; $r_{\mathrm{priv}}(a, c_t)$ represents the CDSE privacy exposure under context c_t; and $r_{\mathrm{adv}}(a)$ denotes the integrity risk, with both risk measures taking values in $[0, 1]$. The utility loss is represented by $\Delta U(a) \ge 0$.

The parameters ε_p, ε_a, and ε_u represent the privacy, integrity, and utility budgets, respectively, while D denotes the latency deadline. The function $\mathbb{I}[\cdot]$ is the indicator function; T denotes the trace length; and $\delta_{\mathrm{C}} \in [0, 1]$ specifies the tolerated long-run violation rate associated with constraint C.

Because n_t is only partially observed and may vary within an epoch, the controller operates using a predicted network state $\hat{n}_t$. The latency deadline D is tightened online when latency violations accumulate, as described in Section 4.4.

If no action $a \in \mathcal{A}$ satisfies the risk constraints under $\hat{n}_t$, the controller does not relax the prescribed budgets ε_p and ε_a. Instead, it invokes the safe fallback described below, prioritizing privacy and integrity compliance even when offloading must be avoided.

Two properties of (1) are worth stating now because they shape the solution. First, the risk constraints and the resource objective can conflict: the cheapest action is frequently the least private, so the feasible set can exclude the cost minimiser and, in poor network conditions, can shrink to the local action alone. Second, the feasible set depends on the state through the deadline only; the risk constraints are state-independent, which means privacy and integrity feasibility can be precomputed per action and cached, leaving only a cheap latency check at decision time. We exploit this in the controller's complexity.

4. Methodology

4.1. Overview

MobiTrust turns the constrained problem of Section 3 into a system with two measurement components and one decision component. The measurement side estimates

the two risks that the objective must respect: CDSE for privacy, and Latent Integrity Risk (LIR) together with a route-trust model for integrity. The decision side is a primal–dual controller that, per inference, filters the action set to those satisfying every risk and deadline budget and returns the cheapest survivor, falling back to on-device execution when the filter is empty.

The pipeline has seven parts. A device head computes the representation at the chosen split. A protection stage applies the chosen mode g_p. The CDSE estimator scores privacy exposure for a given split, protection, and context. The LIR detector scores representations for tampering and yields the detector's discriminative power per split. A route-trust model converts the chosen edge into an integrity risk moderated by that detector. The primal–dual controller selects the action. A safe-fallback manager guarantees a risk-compliant action always exists. The first five are profiled once, offline; only the last two run on the inference path, and they cost a constant number of table lookups per decision (Section 4.5). Figure 2 shows how the parts connect: an offline stage measures the risks and caches them into the action set, and an online loop filters that set against the budgets each inference, tightening its behaviour through the primal–dual multipliers and retreating to the fallback when nothing safe fits.

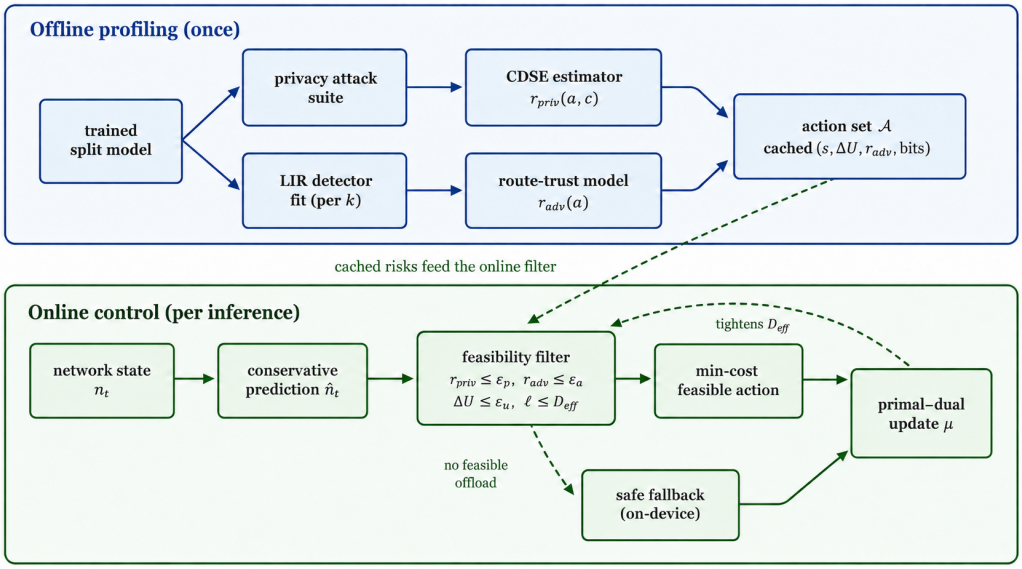


Figure 2. The MobiTrust framework. Offline (**top**), the privacy attack suite and the LIR detector are run once to produce the CDSE and route-trust risks, which are cached with each action. Online (**bottom**), the controller predicts the network state conservatively, filters actions against the privacy, integrity, utility, and deadline budgets, and returns the least-cost feasible action or, when no offload is safe, the on-device fallback; primal–dual updates then adjust the effective deadline.

4.2. Context-Dependent Semantic Exposure

Privacy leakage from a representation is not one quantity but several, and which of them matters depends on the deployment. CDSE measures each channel empirically and then aggregates them under a context that says what the deployment is trying to protect.

4.2.1. Leakage Channels

For a split k, protection p, and an auxiliary attack set, we measure three normalised leakage scores in $[0, 1]$. Reconstruction exposure s_{rec} is the structural similarity between inputs and the best decoder inversion of $\tilde{z}_k$, so that a faithfully invertible representation scores near one. Membership exposure is $s_{\text{mem}} = 2\max(0, \text{AUC}_{\text{mem}} - \frac{1}{2})$, the calibrated advantage over chance of a shadow-model membership attack (Shokri-style): $n_{\text{sh}} = 24$ shadow

models replicate the target's training recipe on random data splits; a learned attack classifier is trained on their in-/out- membership features (loss, confidence, entropy, true-class logit, and prediction margin), and it is evaluated on the target's members versus non-members. Section 6.2 validates that this adversary recovers a strong signal on an overfit reference model, so a low deployed score reflects a weak leak, not a weak attack.

Attribute exposure $s_{\text{attr}} = 2\max(0, \text{AUC}_{\text{attr}} - \frac{1}{2})$ is the advantage of a probe trained to read a declared sensitive attribute from $\tilde{z}_k$. Collect them as $s(a) = (s_{\text{rec}}, s_{\text{mem}}, s_{\text{attr}})$.

4.2.2. Context Weighting and Aggregation

A context c assigns non-negative weights $w(c) = (w_{\text{rec}}, w_{\text{mem}}, w_{\text{attr}})$ to the channels. In the default context, the weights are balanced; a context that declares identity sensitive raises w_{attr}, one handling clinical records raises w_{mem}, and so on. CDSE budgets the worst weighted channel, because a privacy limit must bound the strongest leak, not the average one:

$$r_{\text{priv}}(a, c) \;=\; \text{CDSE}_{\text{wc}}(a, c) \;=\; \max_{j \in \{\text{rec,mem,attr}\}} w_j(c)\, s_j(a). \tag{3}$$

Here, j indexes the three leakage channels, reconstruction, membership, and attribute; $s_j(a) \in [0, 1]$ is the measured leakage of channel j for action a (so s_{rec} is the reconstruction similarity, and $s_{\text{mem}}, s_{\text{attr}}$ are the chance-corrected membership and attribute advantages defined above); $w_j(c) \geq 0$ is the weight that context c places on channel j; and max selects the strongest weighted channel so that the score bounds the worst leak rather than the average. We also define a mean variant $\text{CDSE}_{\text{mean}} = \sum_j w_j s_j / \sum_j w_j$ for comparison. The mean has slightly higher rank correlation with average attack success on our data, but the worst-case form is the correct operational choice: a deployment that averages a benign membership signal against a severe reconstruction leak would declare an unsafe action safe. We report both in Section 6 and use the worst-case form in the constraint.

4.2.3. Calibration

The raw score in (3) is monotone in exposure but not itself a probability. When a deployment wants a calibrated breach likelihood, for example to set ε_p from a target breach rate, we fit an isotonic regression from CDSE to the empirical worst-channel attack success measured on held-out attackers, which preserves ranking while producing calibrated values. Section 6 reports the resulting reliability and Brier score.

The design choice that CDSE encodes, and that we test directly, is that structural proxies for leakage are unreliable once protection is in play. Split depth and tensor size are computable without any attack, which is why prior schedulers use them, but a shallow split under strong noise can leak less than a deep unprotected one. CDSE pays the offline cost of measuring leakage so that the online ranking is correct.

4.3. Latent Integrity Risk and Route Trust

4.3.1. Detector

The LIR detector models benign representations at split k and flags departures. It fits class-conditional means μ_y and a pooled covariance Σ on benign training activations and scores a representation by its class-conditional Mahalanobis distance,

$$d_M(z) \;=\; \min_y (z - \mu_y)^\top \Sigma^{-1} (z - \mu_y), \tag{4}$$

where z is the representation being scored, y ranges over the class labels, μ_y is the mean of benign training activations of class y at this split, Σ is the covariance pooled across classes, and Σ^{-1} its inverse; the minimum over y takes the distance to the nearest class

centroid, so a representation far from every benign class scores high. combined with the reconstruction error of a small autoencoder trained on the same benign activations. The two terms are standardised against their benign distribution and summed, then squashed to $[0, 1]$, giving a score that is low for in-distribution representations and high for perturbed ones. Because the detector is differentiable in z through (4), we can also use its gradient to build detector-aware adaptive attackers for evaluation (Section 6). To avoid reading the evasion/effectiveness trade-off of a single construction, we evaluate four such strategies: (i) a sign-based penalty attack maximising task loss while descending the detector score, $g = \text{sign}(\nabla_z \mathcal{L}) - \lambda \text{ sign}(\nabla_z d_M)$ with $\lambda = 2$; (ii) its normalised-gradient variant; (iii) an on-manifold centroid attack that moves toward the nearest wrong-class benign mean, evading d_M by construction; and (iv) a C&W-style objective trading a task margin against the detector score with a proximity term. All run for 15–30 projected steps within a relative η-ball ($\eta = 0.15$ unless swept), the same budget as the non-adaptive PGD reference.

4.3.2. Route Trust

The detector cannot, on its own, bound the risk of a corrupted prediction, because that risk also depends on who is serving the tail. We attach to each edge e a base route-trust risk $\rho(e)$, low for a trusted operator edge, high for an opportunistic untrusted one, and let the detector's discriminative power at split k, its area under the ROC curve AUROC_k, discount it:

$$r_{\text{adv}}(a) \;=\; \text{clip}_{[0,1]}\Big(\rho(e)\,\big(1 - \tfrac{1}{2}\,\text{AUROC}_k\big)\Big). \tag{5}$$

Here, e is the serving edge chosen by action a; $\rho(e) \in [0, 1]$ is the base route-trust risk of that edge (small for a trusted edge, large for an untrusted one); $\text{AUROC}_k \in [0, 1]$ is the area under the LIR detector's ROC curve at split k, a measure of how reliably tampering is caught there; the factor $1 - \frac{1}{2}\text{AUROC}_k$ discounts the route risk by up to one half as detectability improves; and $\text{clip}_{[0,1]}$ confines the result to a valid risk. A more detectable split earns a larger discount because tampering there is more likely to be caught. Routing to an untrusted edge raises r_{adv} toward its base value, which the integrity budget ε_a then constrains; a trusted edge keeps it low.

By default $\rho(e)$ is an operator trust policy, an administrative assignment of trust to a serving domain. This is not the same kind of object as the split-depth proxy the paper argues against: depth is used elsewhere as a stand-in for a quantity (leakage) that can be measured directly and that depth predicts poorly once protection is applied, whereas $\rho(e)$ encodes a trust decision about an administrative party. That said, the reviewers rightly ask whether route trust can itself be measured rather than declared, and it can: we show in Section 6.11 that a Beta–Bernoulli estimator driven by the integrity layer's observed flag rate per edge converges to an empirical flag-rate score $\hat{s}(e)$ that orders the edges correctly (and, when the detector's error rates are known, deconvolves to the true tampering probability).

We are careful about what this licenses. Because Equation (5) tests r_{adv} against a fixed budget ε_a, the feasible set depends on the numerical value substituted for $\rho(e)$, not only on the edge ordering, so the estimated score is not a drop-in replacement for a policy value on a different scale: using it in the loop requires recalibrating ε_a to the flag-rate scale or first mapping the score onto the policy scale (e.g., by deconvolution). The claim we stand behind is therefore the narrower one, that the trust ordering is measurable online from observable behaviour, rather than that the raw score can replace $\rho(e)$ under an unchanged budget. We therefore treat the hand-assigned value as a prior/policy default and the estimator as the measured alternative, with the scale caveat above, rather than defending an unmeasured proxy.

4.3.3. Freshness Verification Against Replay

Equation (4) scores distance from the benign manifold, so it is blind to an attack that stays on that manifold. A replay attack that substitutes a genuine representation from a different input is exactly such a case: the replayed z is in-distribution, and the detector cannot distinguish it, even though it corrupts the prediction. We close this gap with a freshness check rather than by asking the manifold detector to do something it cannot. The device binds each transmitted representation to a fresh per-inference nonce and a monotonically increasing sequence counter under a message-authentication code keyed on the device and the trusted edge, and the verifier accepts a payload only when

$$\mathrm{MAC}_{\kappa}(\tilde{z}_k, \nu, c) \text{ valid} \quad \wedge \quad \nu \notin \mathcal{N} \quad \wedge \quad c > c^{\star}, \tag{6}$$

where κ is the shared MAC key, ν the per-inference nonce, c the payload's sequence counter, $\mathcal{N}$ the cache of recently seen nonces, $c^{\star}$ the highest counter accepted so far, and $\wedge$ logical conjunction. An attacker without κ cannot forge a fresh tag, so a replayed payload is rejected either because its nonce is still cached or because its counter is stale. Section 6 measures that this catches every exact replay at zero false-positive cost. Because the tag covers $\tilde{z}_k$ itself, the same check also rejects any external modification of the representation, including a latent perturbation, whose tag would no longer verify. The LIR detector is therefore redundant against an external on-path attacker on an authenticated channel; its role (and the perturbation-detection results of Section 6) belong to the complementary threat model of Section 3, an untrusted or compromised serving edge, which is the legitimate recipient, holds no obligation to a shared key, and can tamper with the representation it feeds to the tail. There, authentication does not help and a distance-based detector plus the route-trust budget r_{adv} are the operative defences. What freshness does not address is an attacker forwarding a genuine representation of its own current input; that is a routing-trust question, captured by r_{adv}, not a replay.

We state the scope of this check narrowly, because it is weaker than a round-the-clock guarantee in two specific ways. First, recomputation runs on the sampled fraction only, so its coverage against an untrusted edge is probabilistic: an edge that tampers with an unsampled inference is caught with probability equal to the sampling rate, not with certainty. Second, scoring $\tilde{z}_k$ with LIR inspects the representation the device sent; it does not reveal a different representation that an untrusted edge may use internally before computing the tail, because the monitor never observes that internal value. Consequently, we do not claim that the only attack escaping both checks is an ineffective substitution: an untrusted recipient can substitute internally and return the corresponding prediction on an unsampled inference, escaping recomputation (the inference was not sampled) and LIR (which sees only the authenticated $\tilde{z}_k$), and this is an effective attack bounded only by the sampling rate. This is precisely the untrusted-recipient limitation acknowledged in Section 9.2; the monitor is a sampled deterrent against it, not a per-inference guarantee, and r_{adv} accordingly remains a score rather than a bound.

We also distinguish this end-to-end arrangement from the component-level detection results in Section 6 (replay and perturbation): those measure how well LIR and the freshness check separate representations the detector actually inspects, and are not measurements of this monitoring arrangement against an untrusted edge. Finally, the sampled recomputation of the tail and the monitor's LIR scoring are accounted for in the complexity analysis (Section 4.5) but are not included in the empirical latency and energy figures of Section 6, which report only the controller's own on-path actions; this verification overhead, which scales with the sampling rate, is an additional cost not reflected in those measurements.

4.3.4. The Integrity Monitor, Concretely

For the untrusted-edge case, we make the check explicit. A trusted monitor, the device itself, or a trusted verifier, receives, for a sampled fraction of inferences, the authenticated representation $\tilde{z}_k$ (carried under the tag of Equation (6), so the monitor knows it is the representation the device actually sent) together with the edge's returned prediction $\hat{y}$. The binding between $\hat{y}$ and $\tilde{z}_k$ is established not by trusting the edge but by recomputation: on the sampled inferences, the monitor evaluates the public tail $f_k^{edge}(\tilde{z}_k)$ itself and flags a mismatch with the returned $\hat{y}$, and on every inference, it scores $\tilde{z}_k$ with LIR and flags an anomalous representation. An untrusted edge that perturbs $\tilde{z}_k$ or fabricates $\hat{y}$ is therefore caught either by the recomputation disagreement or by the LIR score; its only tamper that evades both is an in-distribution substitution that also happens to produce the "correct" tail output, which is not a useful attack. The cost is the sampled recomputation of the tail at the monitor and the LIR score per inference, both of which the complexity analysis already accounts for. We note that the monitor's power is bounded by its sampling rate and by LIR's detection rate, so r_{adv} remains a score, not a guarantee.

We state the scope of this check narrowly, because it is weaker than a round-the-clock guarantee in two specific ways. First, recomputation runs on the sampled fraction only, so its coverage against an untrusted edge is probabilistic: an edge that tampers with an unsampled inference is caught with probability equal to the sampling rate, not with certainty. Second, scoring $\tilde{z}_k$ with LIR inspects the representation the device sent; it does not reveal a different representation that an untrusted edge may use internally before computing the tail, because the monitor never observes that internal value. Consequently, we do not claim that the only attack escaping both checks is an ineffective substitution: an untrusted recipient can substitute internally and return the corresponding prediction on an unsampled inference, escaping recomputation (the inference was not sampled) and LIR (which sees only the authenticated $\tilde{z}_k$), and this is an effective attack bounded only by the sampling rate.

This is precisely the untrusted-recipient limitation acknowledged in Section 9.2; the monitor is a sampled deterrent against it, not a per-inference guarantee, and r_{adv} accordingly remains a score rather than a bound. We also distinguish this end-to-end arrangement from the component-level detection results in Section 6 (replay and perturbation): those measure how well LIR and the freshness check separate representations the detector actually inspects, and are not measurements of this monitoring arrangement against an untrusted edge. Finally, the sampled recomputation of the tail and the monitor's LIR scoring are accounted for in the complexity analysis (Section 4.5) but are not included in the empirical latency and energy figures of Section 6, which report only the controller's own on-path actions; this verification overhead, which scales with the sampling rate, is an additional cost not reflected in those measurements.

4.4. Risk-Bounded Controller

4.4.1. What the Online Rule Is, Precisely

We derive the controller from a constrained (Lagrangian) view of problem (1), but we are deliberate about what the implemented online rule does, because the two can be conflated. In the nominal setting of this paper, the privacy, integrity, and utility risks are state-independent and profiled exactly offline, so the per-inference rule is a feasibility filter, keep the actions whose cached risks are within budget and whose latency meets the (possibly tightened) deadline, then take the least-cost survivor, together with a deadline-adjustment heuristic driven by the latency multiplier μ_{T}. The privacy and integrity multipliers μ_p, μ_a are updated by (8) but, in this nominal case, have nothing to correct: the filter already guarantees the selected action's profiled risk is within budget, so

the realised risk equals the profiled risk and the violation indicator is identically zero. This is why the A6 ablation (duals off) is indistinguishable from the full method in Section 7, and we state it as such rather than claiming an always-active dual. The primal–dual machinery becomes genuinely load-bearing once the risk scores carry estimation uncertainty, so that realised risk can exceed the profiled estimate: a dual-driven safety margin then tightens the effective risk budgets and holds the realised violation rate at its tolerance. We demonstrate this directly in Section 6.9 and discuss it under limitations.

4.4.2. Scores Versus Validated Bounds

A clarification that we carry throughout: holding a constraint "at zero violations" means the selected actions comply with the profiled risk scores and the assumed route-trust model, not that an independently validated probability of a successful attack is bounded. The scores are monotone, calibrated summaries of attacker success (and, for integrity, a detector's discrimination), not certified upper bounds; in particular, a detector's AUROC measures ranking across thresholds, not the acceptance probability of a successful attack at a deployed threshold. All "risk-bounded" and "zero-violation" statements should be read in that conditional sense.

4.4.3. Offline Profiling

All state-independent quantities are computed once. For every split and protection the profiler runs the privacy attack suite to obtain $s(a)$ and measures the utility loss $\Delta U(a)$; for every split, it fits the LIR detector and records AUROC_k; it then materialises the action set with each action's cached leakage vector, integrity risk (5), utility loss, and transmitted size. Algorithm 1 states this.

Algorithm 1: Offline action profiling.

Input: trained split model f; splits $\mathcal{K}$; protections $\mathcal{P}$; edges $\mathcal{E}$; benign data; attack suite
Output: action set $\mathcal{A}$ with cached risks
foreach $k \in \mathcal{K}$ **do**
  fit LIR detector on benign activations at k; record AUROC_k;
  foreach $p \in \mathcal{P}$ **do**
    measure $s(a) = (s_{\mathrm{rec}}, s_{\mathrm{mem}}, s_{\mathrm{attr}})$ via attacks on $g_p(z_k)$;
    measure utility loss ΔU of g_p at split k; record transmitted bits;
    foreach $e \in \mathcal{E}$ **do**
      $r_{\mathrm{adv}} \leftarrow \mathrm{clip}(\rho(e)(1 - \frac{1}{2}\mathrm{AUROC}_k))$;
      add action (k, p, e) with $(s, \Delta U, r_{\mathrm{adv}}, \mathrm{bits})$ to $\mathcal{A}$;
add raw-offload and local actions to $\mathcal{A}$;
return $\mathcal{A}$

4.4.4. Online Decision

At epoch t, the controller forms a conservative estimate of the network state, tightens the deadline according to the latency multiplier, filters the action set to those satisfying all four budgets, and returns the cheapest feasible action; if no offload is feasible, it invokes the fallback. It then updates the multipliers from realised violations. Algorithm 2 gives the procedure.

State prediction is deliberately pessimistic about bandwidth. We keep an exponentially weighted mean $\bar{b}_t = \alpha b_t + (1 - \alpha)\bar{b}_{t-1}$ and predict $\hat{b}_t = \min(b_t, \bar{b}_t)$, so that a controller planning an offload assumes the worse of the current and the smoothed bandwidth. In

these expressions, b_t is the bandwidth observed at epoch t, $\bar{b}_t$ its exponentially weighted running mean, $\alpha \in (0, 1]$ the smoothing rate, and $\hat{b}_t$ the bandwidth used for planning. This guards against committing to a split whose transmission will overrun the deadline when the link is deteriorating.

The deadline is tightened as latency violations accrue. With latency multiplier $\mu_{\mathrm{T}} \geq 0$, the effective deadline is

$$D_{\mathrm{eff}} \;=\; D\Big(1 - 0.4\big(2\sigma(\mu_{\mathrm{T}}) - 1\big)\Big), \tag{7}$$

where D is the nominal deadline, μ_{T} is the Lagrange multiplier for the latency constraint, and $\sigma(u) = 1/(1 + e^{-u})$ is the logistic function; since $\sigma(0) = \frac{1}{2}$, the tightening term vanishes at $\mu_{\mathrm{T}} = 0$ (so $D_{\mathrm{eff}} = D$) and grows toward $0.4D$ as violations push μ_{T} up, shrinking D_{eff} to as little as $0.6D$. After each decision the multipliers follow projected sub-gradient steps that track the chance constraints,

$$\mu_{\mathrm{C}} \leftarrow \big[\mu_{\mathrm{C}} + \beta\big(\mathbb{I}[\mathrm{C\ violated}] - \delta_{\mathrm{C}}\big)\big]_{+}, \qquad \mathrm{C} \in \{p, a, \mathrm{T}\}, \tag{8}$$

where μ_{C} is the multiplier for constraint C, $\beta > 0$ the dual step size, $\mathbb{I}[\mathrm{C\ violated}]$ the indicator that constraint C was violated at this epoch, δ_{C} its tolerated violation rate, and $[\,\cdot\,]_{+} = \max(0, \cdot)$ the projection onto non-negative multipliers. A constraint that is violated more often than its tolerance drives its multiplier up, which, through (7) for latency, and through the profiling that keeps risk actions out of the feasible set, makes the controller more conservative until the violation rate falls back to δ_{C}.

Algorithm 2: Online risk-bounded control at epoch t.

Input: action set $\mathcal{A}$; observed state $n_t = (b_t, \tau_t, q_t)$; context c_t; budgets $\varepsilon_p, \varepsilon_a, \varepsilon_u, D$; multipliers μ
Output: action a_t; realised outcome
$\bar{b} \leftarrow \alpha b_t + (1 - \alpha)\bar{b}$; $\quad \hat{n}_t \leftarrow (\min(b_t, \bar{b}), \tau_t, q_t)$;
$D_{\mathrm{eff}} \leftarrow D\,(1 - 0.4(2\sigma(\mu_{\mathrm{T}}) - 1))$;
$\mathcal{F} \leftarrow \{\, a \in \mathcal{A} : r_{\mathrm{priv}}(a, c_t) \leq \varepsilon_p,\ r_{\mathrm{adv}}(a) \leq \varepsilon_a,\ \Delta U(a) \leq \varepsilon_u,\ \ell(a, \hat{n}_t) \leq D_{\mathrm{eff}} \,\}$;
if $\mathcal{F}$ *contains an offload action* **then**
  $a_t \leftarrow \arg\min_{a \in \mathcal{F}} \phi(a, \hat{n}_t)$;
else
  `// safe fallback: never relax the risk budgets`
  $\mathcal{S} \leftarrow \{\, a : r_{\mathrm{priv}} \leq \varepsilon_p,\ r_{\mathrm{adv}} \leq \varepsilon_a,\ \Delta U \leq \varepsilon_u \,\}$;
  $a_t \leftarrow \arg\min_{a \in \mathcal{S}} \ell(a, \hat{n}_t)$ if $\mathcal{S} \neq \emptyset$ else local;
execute a_t; observe realised latency and risks under true n_t;
update each μ_{C} by (8);
return a_t

4.4.5. Safe Fallback

The fallback is what makes the per-inference guarantee hold regardless of whether the multipliers have converged. When the feasible set contains no offload, the controller does not widen the budgets; it restricts to actions that satisfy the risk constraints, which always includes local execution, since on-device inference discloses nothing and cannot be tampered with in transit, and among those picks the one with least latency. Privacy and integrity are therefore never traded for latency, even transiently. The visible cost of this discipline is a higher fallback rate and somewhat higher latency and energy in poor conditions, which we quantify rather than gloss.

4.5. Complexity and Overhead

Offline profiling runs the attack suite once per split–protection pair and fits one detector per split, so its cost is $O(|\mathcal{K}||\mathcal{P}|\,T_{\text{atk}} + |\mathcal{K}|\,T_{\text{fit}})$ and is paid before deployment. Online, each decision evaluates the four budget checks for every action; the privacy, integrity, and utility checks are table lookups because those risks are state-independent and cached, and only the latency term is computed from the state, at $O(1)$ per action. A decision is therefore $O(|\mathcal{A}|)$ with a small constant, plus $O(1)$ multiplier updates; with the action sets used here ($|\mathcal{A}| < 40$), the controller adds negligible overhead to the inference path. The one online model cost, if integrity checking is enabled at run time, is a single Mahalanobis score per received representation, which is a matrix–vector product in the latent dimension.

4.6. Implementation

The reference implementation is written in NumPy so that every step, the split network's forward and backward passes, the attacks, the detector, and the controller, is explicit and inspectable, and so that the whole study runs on CPU. The split network is a multilayer perceptron with hidden widths 128, 64, 32, and candidate splits after each hidden layer; its analytic gradients were checked against numerical differences to a maximum relative error of 4×10^{-10}, which matters because the latent attacks rely on true gradients. Protection modes are quantisation, two levels of additive noise, a stronger differential-privacy style perturbation, and a channel-keeping bottleneck. Latency and energy come from a workload-representative analytical model, described in Section 5, that stands in for a MobileNet-class network on constrained hardware; the risk quantities, by contrast, are measured on the actual trained network. All randomness is seeded, and the released code regenerates every table and figure.

5. Experimental Setup

5.1. Workloads

We evaluate on two workloads chosen to differ in modality and in which leakage channel dominates. The image workload uses the DIGITS dataset (1797 images, 8×8 greyscale, ten classes), a standard vision proxy small enough to run the full attack suite many times under seeding. The trained split network reaches a test accuracy of 0.967. Its dominant leakage channel is reconstruction, which makes it a stringent test of whether protection can decouple leakage from split depth. The sensing workload is a synthetic six-class human-activity signal (36 channels) constructed with overlapping per-class frequency signatures, additive noise, and a binary orientation attribute that acts as the sensitive attribute; the network reaches 0.956 test accuracy with a deliberate train/test gap so that membership signal is present. We label this workload synthetic throughout and do not claim it substitutes for a real accelerometer corpus; it is included to check that the mechanisms behave on a non-vision modality where attribute inference, not reconstruction, is the binding channel. Each input carries a binary sensitive attribute (high ink for images, orientation for sensing) used by the attribute probe.

5.2. Cost Model

Because no mobile hardware or live radio was available, latency and energy are produced by an explicit analytical model, stated here in full so the reader can judge it. The model represents a MobileNet-class network of 3×10^8 multiply–accumulates. The device executes at 2.4×10^6 MAC ms^{-1} and the edge at 2.4×10^8 MAC ms^{-1}, so full local execution takes 125 ms and edge computation is roughly a hundred times faster. The fraction of computation performed on the device grows with the split index (0.25, 0.60, 0.88 for the three splits), and the transmitted feature-map size falls with depth (280, 70, 10 KB), matching

the shape of real convolutional networks where early layers produce large activations and late layers small ones. Protection scales the transmitted size by its bit-width. Transmission time is the payload divided by the instantaneous bandwidth plus one round trip; energy sums a compute term, a radio term, and an idle term with powers of 2500, 800, and 50 mW. Local execution transmits nothing. This model is transparent and, by construction, swappable.

We note explicitly that the cost model is coupled to action selection: the per-split compute fractions and payload sizes above represent a MobileNet-class network and differ from the literal MACs and activation dimensions of the small profiling MLP, so they influence which split the controller prefers, not only absolute timings. Section 6.10 therefore sweeps the device/edge speeds, payload sizes, and compute fractions over wide ranges and confirms the qualitative conclusions are invariant. The released code isolates the cost model in one module so that measured traces can replace it without changing the risk logic or the controller. The claims we make concern the controller's decisions and their effect on measured risk, and are robust to the absolute constants here; we return to this in Section 9.3.

5.3. Mobility Traces

Network states come from a trace generator with seven regimes, stationary with good link, stationary congested, walking, vehicular, handover, intermittent, and sudden congestion, each with characteristic bandwidth, round-trip time, and queue distributions. Unless noted, controllers are evaluated on a mixed trace that transitions among regimes, so that no single connectivity condition dominates. Traces are 1500 epochs long and generated under ten seeds for the main comparison; the sensitivity study uses shorter traces for the many-run sweeps, as noted in Section 8.

5.4. Risk Budgets

The operating point used throughout, unless a sweep varies it, is a privacy budget $\varepsilon_p = 0.88$, an integrity budget $\varepsilon_a = 0.30$, a utility budget $\varepsilon_u = 0.18$, and a deadline $D = 150$ ms, with a tolerated exceedance $\delta = 0.05$ for the risk constraints and 0.10 for the deadline. These are operator choices, not tuned to flatter the method; the sensitivity study in Section 8 sweeps them and shows the qualitative behaviour is stable. The budgets are calibrated to the measured action ranges so that the feasible set is non-trivial and depends on the network: under good connectivity, several protected splits are feasible, while under congestion, the feasible set can contract to local execution.

5.5. Baselines

We compare against seven controllers spanning the design space. B1 FullLocal always runs on device. B2 FullEdge always offloads the raw input. B3 StaticSplit fixes a mid-network split with no protection. B4 LatencyOnly greedily minimises latency. B5 EnergyLatency minimises a latency–energy cost. B9 WeightedTrust minimises a scalar objective that adds privacy and integrity as weighted penalties,

$$J(a,n) = \ell(a,n) + \lambda_e\,\eta(a,n) + \lambda_c\,\frac{b(a)}{10^3} + w_P\,r_{\text{priv}}(a,c) + w_A\,r_{\text{adv}}(a), \tag{9}$$

where ℓ is latency in ms, η energy in mJ, $b(a)$ transmitted bytes, and $r_{\text{priv}}, r_{\text{adv}} \in [0,1]$. All controllers share $\lambda_e = 0.5$ and $\lambda_c = 0.2$; the risk weights are $w_P = w_A = 60$, chosen so the maximum risk penalty (60) is of the same order as the latency and energy terms ($\sim$100–150 and $\sim$50–150), i.e., risk is priced as highly as speed. Because a single weighting cannot settle the question, we do not rely on this operating point alone: Section 6 sweeps $w_P = w_A$ across three orders of magnitude and reports the resulting privacy–offload frontier. This is the strongest representative of the common "fold risk into the objective"

approach. B10 Oracle is a clairvoyant upper bound that sees the true current network state and picks the cheapest action satisfying every budget; it shows how much MobiTrust loses to its conservative prediction. MobiTrust itself uses the full machinery of Section 4.

5.6. Evaluation Protocol and Data Separation

Table 2 states the sample counts and the separation between attack training, risk profiling, calibration, and final evaluation, so that no evaluation data is reused for fitting. The image workload has 1078 training (member) and 719 test (non-member) examples; the sensing workload 1080/720. Attackers are trained on one side and scored on held-out data: the reconstruction decoder and attribute probe are fit on training-split representations and evaluated on test-split representations; the shadow membership attack trains 24 shadow models on random halves of the pooled data and is evaluated on the target's members vs. non-members. The CDSE calibration study spans 36 operating conditions = 3 splits $\times$ 6 protections $\times$ 2 contexts: the default context with equal channel weights $w = (1, 1, 1)$ over (reconstruction, membership, attribute), and an identity-sensitive context with $w = (1.3, 1.0, 1.3)$. The isotonic CDSE $\rightarrow$ breach calibration is fit on a calibration attacker and the predictive validity is measured against independent held-out attacker seeds (11, 12, 13), so CDSE is judged on generalisation. The calibration quality reported in Section 6 is a Brier score of 0.004 for isotonic-CDSE against 0.29 for the depth predictor on both workloads.

Table 2. Evaluation protocol: sample counts and stage separation.

Stage	Data Used	Evaluated on
Target training	train split (1078/1080)	—
Reconstruction/attribute	train-split reps	test-split reps
Shadow membership	24 shadows on pooled halves	target members vs. non-members
Risk profiling (CDSE/LIR)	train-split reps, seed 0	—
CDSE isotonic calibration	calibration attacker (seed 0)	—
Predictive-validity target	held-out attackers (seeds 11–13)	36 conditions
Controller evaluation	10 mixed traces, 1500 epochs	per-inference metrics

5.7. Metrics and Statistics

For a controller run over a trace we summarise each quantity as follows. Writing ℓ_t for the realised latency at epoch t over a trace of length T, the tail-latency metric is the empirical percentile

$$p_{95} \;=\; \inf\Big\{x : \tfrac{1}{T}\textstyle\sum_{t=1}^{T}\mathbb{I}[\ell_t \le x] \ge 0.95\Big\}, \tag{10}$$

and p_{50}, p_{99} are defined the same way at 0.50 and 0.99; $\mathbb{I}[\cdot]$ is the indicator function. The violation rate of a constraint C with realised risk r_t^{C} and budget ε_{C} is the fraction of inferences that breach it,

$$V_{\mathrm{C}} \;=\; \frac{1}{T}\sum_{t=1}^{T}\mathbb{I}\big[r_t^{\mathrm{C}} > \varepsilon_{\mathrm{C}}\big], \tag{11}$$

instantiated for privacy ($r_t^{\mathrm{C}} = r_{\mathrm{priv}}$, $\varepsilon_{\mathrm{C}} = \varepsilon_p$), integrity, utility, and, with $\ell_t > D$, the deadline. Mean device energy is $\bar{\eta} = \frac{1}{T}\sum_t \eta_t$, the fallback rate is $F = \frac{1}{T}\sum_t \mathbb{I}[\text{fallback fired at } t]$, and task accuracy net of protection is $\mathrm{Acc} = \mathrm{Acc}_{\mathrm{full}} - \frac{1}{T}\sum_t \Delta U(a_t)$, where $\mathrm{Acc}_{\mathrm{full}}$ is the unprotected model's accuracy and $\Delta U(a_t)$ the utility loss of the action taken.

Each metric is reported as a mean over $S = 10$ seeds with a 95% bootstrap confidence interval. Given per-seed values $m_1, \ldots, m_S$, we draw $B = 10^4$ resamples with replacement, and the interval is the 2.5th and 97.5th percentiles of the resampled means $\{\bar{m}^{(b)}\}_{b=1}^{B}$,

$$\bar{m}^{(b)} = \frac{1}{S}\sum_{i=1}^{S} m_{\kappa_i^{(b)}}, \qquad \mathrm{CI}_{95} = \big[Q_{2.5}(\{\bar{m}^{(b)}\}),\, Q_{97.5}(\{\bar{m}^{(b)}\})\big], \tag{12}$$

where $\kappa_i^{(b)}$ are indices sampled uniformly from $\{1, \ldots, S\}$ and Q_α is the α-percentile.

For the headline comparisons, we test MobiTrust against a baseline on the paired per-seed values (a_i, b_i) with the Wilcoxon signed-rank statistic

$$W = \sum_{i=1}^{S} \operatorname{sgn}(a_i - b_i)\, R_i, \tag{13}$$

where R_i is the rank of $|a_i - b_i|$ among the non-zero differences and sgn the sign function; this is a distribution-free paired test that makes no normality assumption. Across the family of comparisons, we control the family-wise error rate with the Holm–Bonferroni procedure: sorting the p-values $p_{(1)} \leq \cdots \leq p_{(K)}$, the adjusted value is

$$p_{(k)}^{\mathrm{Holm}} = \min\Big(1, \max_{j \leq k} (K - j + 1)\, p_{(j)}\Big), \tag{14}$$

with K the number of comparisons; a result is significant at level 0.05 when $p^{\mathrm{Holm}} \leq 0.05$. Effect size is reported as Cliff's delta,

$$\delta = \frac{1}{S^2} \sum_{i=1}^{S} \sum_{j=1}^{S} \Big(\mathbb{I}[a_i > b_j] - \mathbb{I}[a_i < b_j] \Big) \in [-1, 1], \tag{15}$$

the difference between the probability that a MobiTrust seed exceeds a baseline seed and the reverse; $\delta = -1$ means every MobiTrust value is below every baseline value (uniformly better on a metric where lower is better), and $\delta = +1$ the opposite. The image workload is our primary reporting surface; sensing results, which agree qualitatively, are summarised where they differ and included in full in Section 6.14.

6. Results and Comparative Analysis

We organise the results around the claims of Section 1: that leakage must be measured rather than inferred from structure (CDSE), that latent integrity is a real and partly detectable risk (LIR), and that budgeting these risks as constraints protects every inference where a scalar objective would not (the controller).

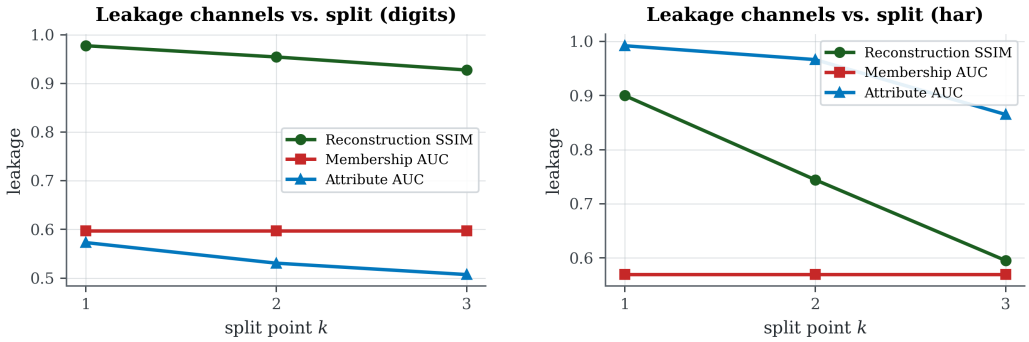
6.1. Leakage Is Multi-Channel and Depth Is a Poor Proxy

Table 3 profiles the image model across splits. On this image workload, it is reconstruction that dominates: structural similarity stays near 0.93–0.98 even at the deepest split we consider, so a legibly invertible representation leaves the device regardless of where we cut. The other two channels are comparatively weak here; attribute AUC is close to chance (0.57, 0.53, 0.51 across $k = 1, 2, 3$) and shadow membership AUC is 0.60, which is itself the point: the binding channel is a property of the workload, not a constant, so a privacy estimate keyed to any single channel mis-measures exposure on some workload. (The sensing workload in Section 6.14 is the mirror image, with attribute inference strong and reconstruction weak.) Figure 3 shows how the three channels move with depth, and Figure 4 shows decoder inversions at each split: the recovered digits remain legible well into the network.

Protection changes the picture in a way depth cannot capture. Table 4 shows that a strong differential-privacy style perturbation at a shallow split drives reconstruction similarity below that of an unprotected deep split, at the price of accuracy. The ordering of options by leakage is therefore not the ordering by depth, the fact that makes structural proxies unsafe.

Table 3. Per-split profile on the digits model (test accuracy 0.967). Reconstruction (SSIM/PSNR), membership (MI AUC, from a 24-shadow-model attack) and attribute (Attr AUC) leakage, and latent-integrity detectability (LIR AUROC). Membership leakage is split-independent because it derives from the final prediction.

Split	$\lvert z_k \rvert$	Client kMAC	Edge kMAC	SSIM	PSNR	MI AUC	Attr AUC	LIR AUROC
$k = 1$	128	8.2	10.6	0.977	21.9	0.597	0.573	0.574
$k = 2$	64	16.4	2.4	0.954	19.0	0.597	0.531	0.703
$k = 3$	32	18.4	0.3	0.928	17.1	0.597	0.507	0.947



(**a**) image (**b**) sensing

Figure 3. Leakage channels versus split depth. Reconstruction and attribute leakage dominate; membership is weak. The channels do not move together, which is why a single scalar keyed to one of them mis-estimates exposure.

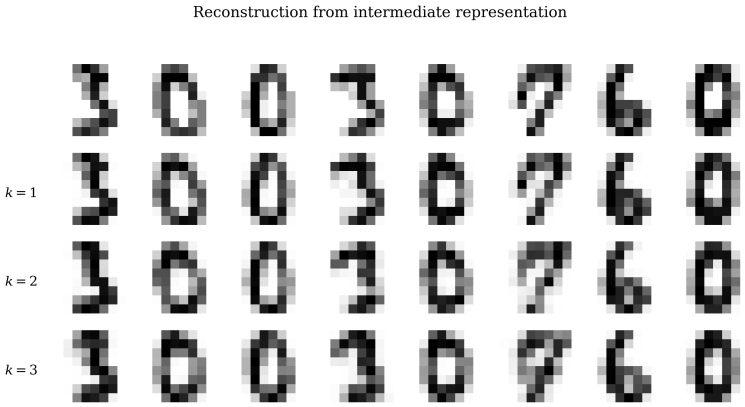


Figure 4. Decoder inversions of the intermediate representation at each split on the image workload. Top row: originals. Lower rows: reconstructions at $k = 1, 2, 3$. Legible inversions persist at the deepest split, so "raw data stays on device" does not imply privacy.

Table 4. Effect of representation protection at split $k = 2$ (digits). Protection reduces reconstruction/attribute leakage at a utility cost, and different modes trade off differently across channels.

Protection	SSIM	PSNR	Attr AUC	Acc	ΔU
none	0.954	19.0	0.531	0.967	0.000
quant4	0.948	18.5	0.505	0.965	0.001
noise_lo	0.921	16.6	0.524	0.951	0.015
noise_hi	0.797	12.6	0.531	0.741	0.225
dp_gauss	0.644	10.1	0.527	0.490	0.477
bottleneck	0.859	14.2	0.524	0.868	0.099

6.2. A Strong Membership Adversary Confirms a Weak Signal

The near-chance membership numbers could, in principle, reflect a weak attacker rather than a weak leak. Table 5 and Figure 5 settle this with a 24-shadow-model attack. On the deployed model, the shadow attack lifts membership AUC only modestly, from 0.53 (loss-threshold) to 0.60, with a true-positive rate at 1% false positives near the 1% floor, membership is genuinely hard to infer from a model that generalises (1.00 train vs. 0.97 test accuracy). The same attack applied to a deliberately overfit reference model (trained on a small subset with no regularisation, 1.00 train vs. 0.88 test) recovers a clear signal, AUC 0.74. A strong adversary that scores high where it should and low where it should is evidence that CDSE's low deployed membership score is a property of the model, not an artifact of a weak test; the sensing workload tells the same story (Table 6).

Table 5. Membership inference on digits. The shadow-model attack is a strong adversary: on a deliberately overfit reference model (trained on a small subset, no regularisation), it recovers a clear membership signal, whereas on the deployed, well-generalised model, the signal stays weak. The low deployed AUC is therefore a property of the model, not of a weak attacker.

Setting	Train Acc	Test Acc	MI AUC	TPR@1%FPR
Deployed (loss-threshold)	1.000	0.967	0.532	–
Deployed (24-shadow)	1.000	0.967	0.597	0.021
Overfit reference (24-shadow)	1.000	0.880	0.742	0.020

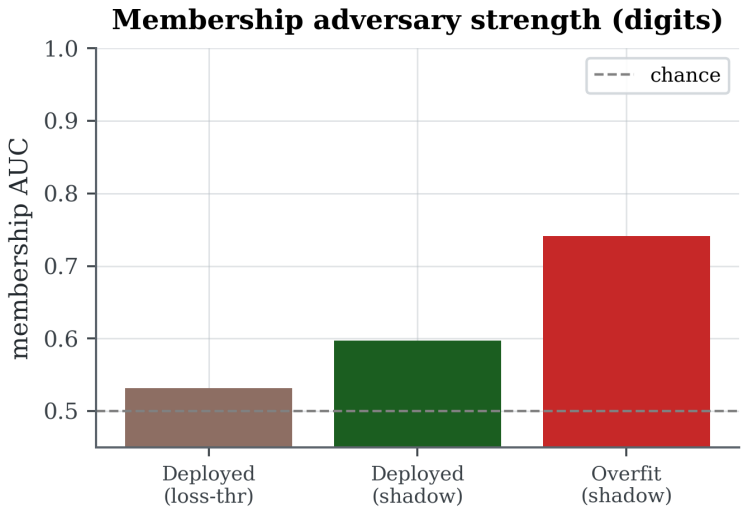


Figure 5. Membership-adversary strength (image). The shadow attack recovers a clear signal on an overfit reference (0.74) but little on the deployed, well-generalised model (0.60): a weak leak, not a weak attack.

Table 6. Membership inference on har. The shadow-model attack is a strong adversary: on a deliberately overfit reference model (trained on a small subset, no regularisation), it recovers a clear membership signal, whereas on the deployed, well-generalised model, the signal stays weak. The low deployed AUC is therefore a property of the model, not of a weak attacker.

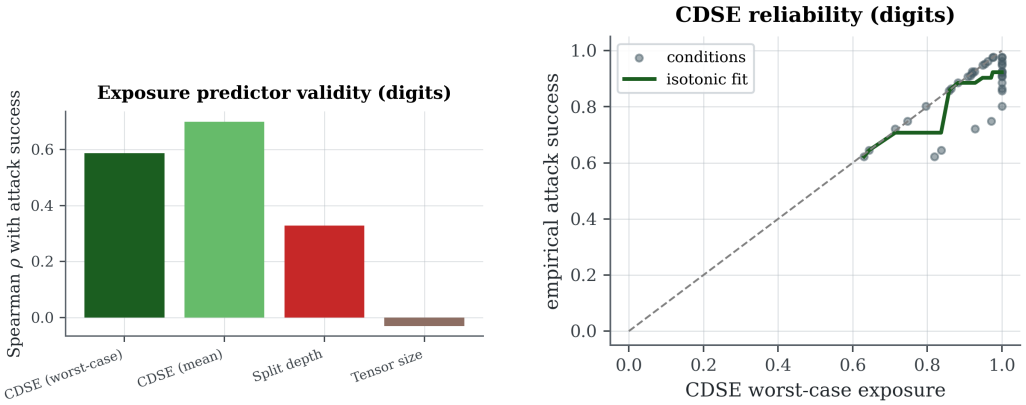
Setting	Train Acc	Test Acc	MI AUC	TPR@1%FPR
Deployed (loss-threshold)	1.000	0.956	0.534	–
Deployed (24-shadow)	1.000	0.956	0.569	0.019
Overfit reference (24-shadow)	1.000	0.892	0.723	0.020

6.3. CDSE Predicts Attack Success; Structural Proxies Do Not

This is the central test of the privacy estimator. Over all split, protection, and context conditions, we correlate three decision-time predictors, CDSE worst-case, split depth, and transmitted size, against the empirical worst-channel attack success measured with independent held-out attackers. Table 7 and Figure 6 give the result. On the image workload, CDSE achieves Spearman $\rho = 0.59$ ($p < 10^{-3}$) against 0.33 for split depth and essentially zero for tensor size; on the sensing workload, the gap is wider, $\rho = 0.87$ for CDSE against 0.72 for depth. The mean-CDSE variant correlates marginally higher than worst-case on the image data (0.70), consistent with averaging being smoother, but as argued in Section 4.2, the worst-case form is the correct constraint because a privacy budget must bound the strongest channel. Figure 6b shows the isotonic calibration of CDSE against measured breach, which is close to the diagonal.

Table 7. Predictive validity of exposure estimators against empirically measured worst-channel leakage (digits, 36 operating conditions). CDSE ranks conditions far better than the structural heuristics used by leakage-agnostic schedulers.

Predictor	Spearman ρ	p	Kendall τ
CDSE (worst-case)	0.587	1.7×10^{-4}	0.486
CDSE (mean)	0.699	2.1×10^{-6}	0.557
Split depth	0.328	5.1×10^{-2}	0.280
Tensor size	−0.030	8.6×10^{-1}	−0.021



(**a**) predictor validity (**b**) CDSE reliability

Figure 6. (**a**) Spearman correlation of each decision-time predictor with measured attack success (image workload); CDSE dominates split depth and tensor size. (**b**) Isotonic calibration of CDSE against empirical worst-channel breach.

6.4. Latent Integrity: What the Detector Catches and Misses

Table 8 reports the LIR detector against an attack suite at a mid-network split, and Figure 7 shows the ROC curves and how detection varies with depth. The detector separates perturbation attacks from benign traffic at AUROC around 0.70 (PGD 0.703, FGSM 0.709, Gaussian 0.721), with detectability rising for deeper splits where the benign manifold is tighter. Two results are negative, and we report them as such. A replay attack, which substitutes a genuine representation from another input, is undetectable (AUROC 0.50) precisely because it is in-distribution, yet it flips the prediction 91% of the time. The manifold detector is the wrong tool against replay; the freshness verifier of Section 4.3 is the right one, and we evaluate it below. A detector-aware adaptive attack does evade the detector (AUROC 0.213, below chance, because it pushes representations toward class

centroids that look more typical than average), but evasion costs it almost all its effect: its flip rate collapses from 0.26 for unconstrained PGD to 0.036.

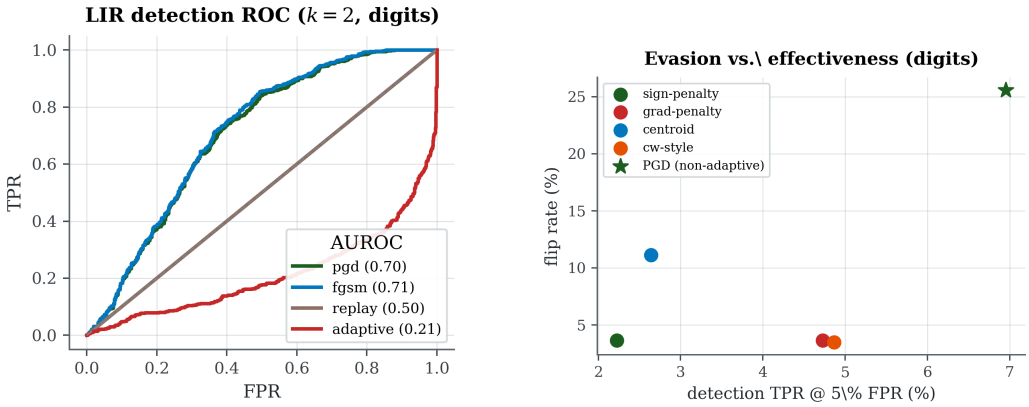
Table 8. Latent-integrity detection at split $k = 2$ (digits). Flip rate is the pre-detection attack success; the adaptive attack is detector-aware and is the hardest to catch.

Attack	Flip Rate	AUROC	AUPRC	FPR@95TPR
gaussian	0.033	0.721	0.651	0.682
fgsm	0.246	0.709	0.635	0.683
pgd	0.256	0.703	0.631	0.691
pgd-targeted	0.082	0.690	0.620	0.719
replay	0.910	0.500	0.500	0.950
adaptive	0.036	0.213	0.363	1.000

The detector thus forces an attacker to choose between being caught and being effective, which is the security-relevant property even though the detector is not individually strong. To check that this trade-off is not an artifact of one attack construction, we evaluate four distinct detector-aware strategies (Section 4.3): Table 9 and Figure 7 report their flip rate against their detectability (TPR at 5% FPR). None of the four escapes the tension: the gradient-penalty and C&W-style attacks reach near-chance detectability (AUROC ≈ 0.49) but flip only 3.5% of predictions; the on-manifold centroid attack raises the flip rate a little (11%) but is caught more often; and all sit far below non-adaptive PGD's 26% flip rate, which is itself the most detectable. Across the four constructions we evaluated, buying evasion costs effectiveness; we report this as an empirical pattern over these attacks rather than a general impossibility.

Table 9. Four detector-aware adaptive strategies at split $k = 2$ (digits), with non-adaptive PGD for reference. Across all constructions, lowering detectability (AUROC/TPR) costs flip rate: evasion and effectiveness are in tension regardless of the specific attack.

Strategy	Flip Rate	LIR AUROC	TPR@5%FPR
sign-penalty	0.036	0.213	0.022
grad-penalty	0.036	0.487	0.047
centroid	0.111	0.260	0.026
cw-style	0.035	0.494	0.049
PGD (non-adaptive)	0.256	0.703	0.070



(**a**) detection ROC (**b**) evasion vs. effectiveness

Figure 7. Latent-integrity detection on the image workload. (**a**) Detection ROC. (**b**) Across four detector-aware adaptive strategies, lower detectability comes only with a lower flip rate; non-adaptive PGD (star) is effective but detectable.

6.5. Closing the Replay Gap with Freshness

The replay result above is not the end of the story but a pointer to the right mechanism. Before comparing the two mechanisms, we fix the setting they are evaluated in, to keep this section consistent with the threat model of Sections 3 and 4.3. Table 10 and Figure 8 evaluate the untrusted-edge case: the serving edge is the recipient and may tamper with the representation it feeds to the tail, so no device–edge authentication tag constrains it, and a distance-based detector (LIR) plus a freshness check are the available defences. On an authenticated device–trusted-edge channel, the situation is different and simpler: because the tag of Equation (6) covers the representation itself, any external modification of it, perturbation or replay, fails verification, so LIR is redundant there. The comparison below is therefore about the untrusted-edge setting, where authentication does not apply.

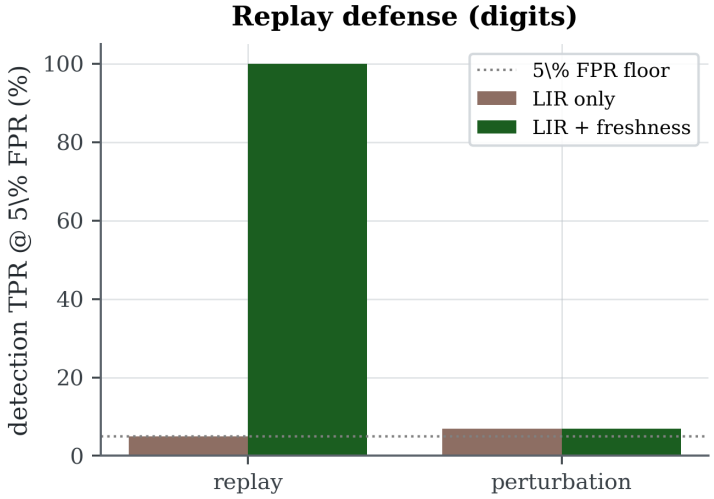


Figure 8. Replay defense on the image workload (all bars are detection TPR at a 5% FPR operating point). LIR alone cannot separate an in-distribution replay; adding freshness verification detects every exact replay, while neither mechanism's role in perturbation detection changes.

Table 10. Replay defense at split $k = 2$ (digits). Detection columns (LIR TPR, Combined TPR) are both true-positive rates at a 5% false-positive operating point, so they are directly comparable; LIR AUROC is a separate threshold-free summary. LIR cannot separate an in-distribution replay (AUROC ≈ 0.5, TPR near the 5% FPR floor); adding freshness verification lifts combined replay detection to 100%, while perturbation attacks remain LIR's responsibility.

Attack	Flip Rate	LIR AUROC	LIR TPR	Combined TPR
Replay	0.910	0.500	0.050	1.000
Perturbation (PGD)	0.256	0.703	0.070	0.070

Table 10 and Figure 8 evaluate the freshness verifier of Equation (6) alongside LIR. To keep the comparison metric-consistent, both the LIR-only and the combined detection columns in Table 10, and both bar groups in Figure 8, are true-positive rates at the same 5% false-positive operating point; LIR's threshold-free AUROC is listed separately for reference. Where LIR detects replay only at the 5%-FPR floor (AUROC 0.50), the freshness check detects 100% of exact replays with a 0% false-positive rate on benign traffic, because a replayed payload always carries a stale sequence counter or a cached nonce. Within this untrusted-edge setting, the two mechanisms are complementary rather than redundant: an untrusted edge can submit a fresh tampered representation (it is the legitimate sender, so freshness verification passes), which is the perturbation case LIR must catch; conversely, LIR cannot catch an in-distribution replay.

Note that this complementarity is specific to the untrusted-edge case; on an authenticated channel, the tag already rejects both, as noted above. These are component-level detection measurements: they quantify how well the freshness check and LIR separate the representations the detector actually inspects, for a tampered payload that reaches the detector. They are not end-to-end guarantees for the monitoring arrangement of Section 4.3: an untrusted recipient that substitutes a representation internally, after a valid payload has been received, is never presented to these detectors and is addressed only by the sampled recomputation check, with the probabilistic coverage and the acknowledged limitation of Section 9.2. The numbers below therefore characterise the detectors, not a bound on an untrusted edge. The false-positive rate stays at zero across nonce-cache windows from 32 to 512, so the memory budget for the cache can be set by traffic rate without affecting correctness. The cost is a per-inference authentication tag on the representation channel, a small constant overhead relative to the transmitted feature map.

6.6. Constraint Satisfaction Under Mobility

The main comparison is in Table 11. Over ten mixed mobility traces, MobiTrust holds privacy, integrity, and deadline violations at 0%. The leakage-agnostic baselines do not: LatencyOnly violates the privacy budget in 87.5% of inferences and the integrity budget in 58.4%, because it grabs the fast raw-offload path and routes to the tempting untrusted edge whenever the link is good. WeightedTrust, despite pricing risk as highly as latency, still violates privacy in 87.9% of inferences, the failure mode predicted in Section 1, where a balanced-on-average weight is overridden in every individual inference that latency dominates, though its integrity violations are lower (7.4%) because its weighting happens to suppress untrusted routing more effectively. The always-on baselines that never offload leaky representations (B1) or always do (B2, B3) sit at the extremes of the safety/latency trade-off.

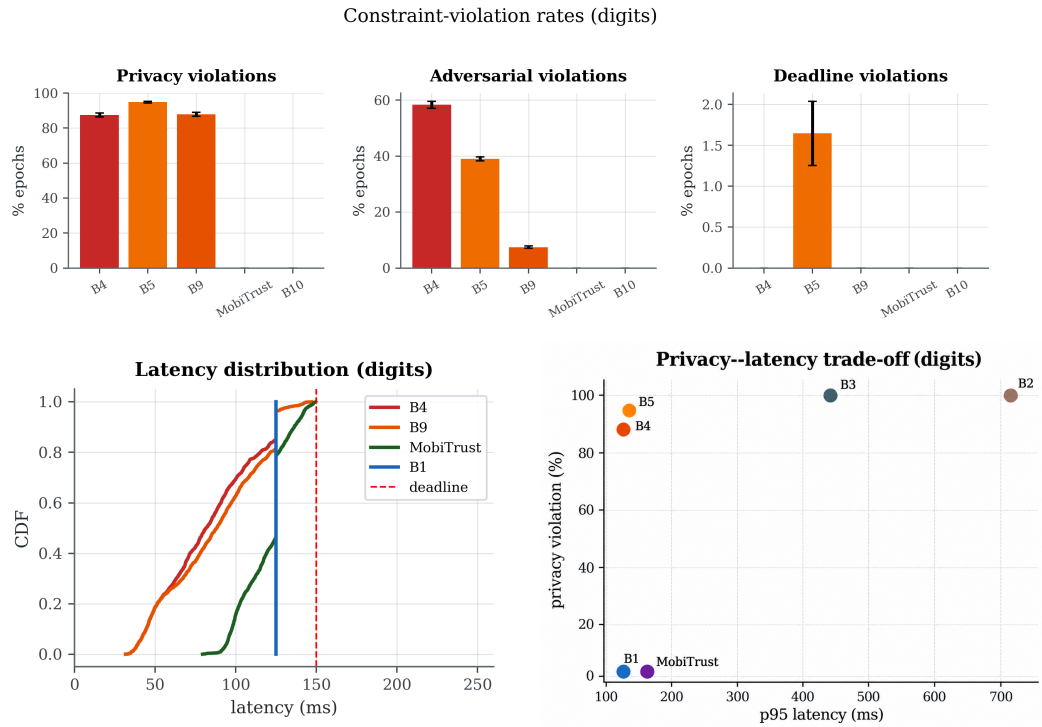
Table 11. Controller comparison on digits over 10 mixed-mobility traces of 1500 epochs each (means). MobiTrust holds privacy and adversarial violations near zero while keeping latency close to the leakage-agnostic baselines.

Controller	p95 (ms)	Energy (mJ)	Priv %	Adv %	Dline %	FB %	Acc
B1-FullLocal	125	312	0.0	0.0	0.0	0.0	0.967
B2-FullEdge	712	224	100.0	0.0	28.6	0.0	0.967
B3-StaticSplit	441	300	100.0	0.0	44.7	0.0	0.967
B4-LatencyOnly	125	110	87.5	58.4	0.0	0.0	0.967
B5-EnergyLatency	133	84	94.9	39.0	1.6	0.0	0.967
B9-WeightedTrust	125	102	87.9	7.4	0.0	0.0	0.967
MobiTrust	142	237	0.0	0.0	0.0	30.6	0.898
B10-Oracle	143	236	0.0	0.0	0.0	0.0	0.896

FullLocal Deserves Explicit Comparison, Not Dismissal

Running everything on device (B1) also attains zero privacy, integrity, and deadline violations, and in Table 11, it does so with higher accuracy (0.967 vs. 0.898) and lower tail latency (125 vs. 142 ms) than MobiTrust. MobiTrust is therefore not a uniform improvement over local execution. Its advantage over FullLocal is energy: it offloads whenever an offload is safe, cutting mean device energy from 312 to 237 mJ (−24%), at the price of some accuracy and tail latency from the protection and fallbacks it uses. The honest reading is that MobiTrust is the right choice when offloading energy matters and the risk budgets must still be respected; when neither offloading nor energy is needed, local execution is a perfectly good, safe baseline. What MobiTrust provides over the offloading controllers (B2–B5, B9) is that it offloads and respects the budgets, which none of them does.

MobiTrust's safety is not free, and we state the cost precisely. Its 95th percentile latency is 142 ms against 125 ms for the greedy baselines, a 13% increase, and its mean energy is higher because it offloads less and falls back to local execution in 30.6% of inferences when no safe offload exists. Its accuracy is 0.898 against the full model's 0.967, reflecting the utility cost of the protection it must sometimes apply. The right way to read these numbers is against the oracle: a clairvoyant controller with the same budgets reaches a 95th percentile of 143 ms and accuracy 0.896, so MobiTrust is within one millisecond and essentially at oracle accuracy. The latency and energy it spends buy the elimination of tens of percentage points of privacy and integrity violation. Figure 9 shows the violation rates with confidence intervals, the latency distribution, and the privacy–latency trade-off across controllers; MobiTrust and the oracle occupy the low-violation corner that no greedy or weighted controller reaches.



(**a**) latency distribution (**b**) privacy–latency trade-off

Figure 9. Controller comparison on the image workload. (**Top**) Constraint-violation rates with 95% confidence intervals. (**a**) Latency distributions against the deadline. (**b**) Privacy violation versus tail latency; among the offloading controllers only MobiTrust reaches the low-violation region occupied by the oracle. FullLocal (B1) also attains zero violations but by never offloading, it is the safe no-offload reference, not an offloading competitor.

6.7. Statistical Significance

Table 12 reports paired tests across seeds. On both privacy and integrity violation, MobiTrust beats LatencyOnly and WeightedTrust with $p = 2 \times 10^{-3}$ before correction and Holm-adjusted significance at the 0.05 level, with Cliff's $\delta = -1.0$, the maximal effect size, meaning MobiTrust is lower on every paired seed. On tail latency, the sign reverses, $\delta = +1.0$, which is the honest cost: MobiTrust is reliably slower than the greedy baselines by the 13% noted above. There is no seed on which a greedy controller matches MobiTrust's safety, and none on which MobiTrust matches a greedy controller's speed; the trade is consistent, not marginal.

Table 12. Paired Wilcoxon signed-rank tests across 10 traces (digits), Holm-corrected. Negative Cliff's δ means MobiTrust is lower (better) on violations.

Comparison	Metric	MobiTrust	Baseline	p	p_{Holm}	Sig.	Cliff δ
MobiTrust vs. B4	priv-viol	0.000	0.875	2.0×10^{-3}	1.2×10^{-2}	yes	−1.00
MobiTrust vs. B9	priv-viol	0.000	0.879	2.0×10^{-3}	1.2×10^{-2}	yes	−1.00
MobiTrust vs. B4	adv-viol	0.000	0.584	2.0×10^{-3}	1.2×10^{-2}	yes	−1.00
MobiTrust vs. B9	adv-viol	0.000	0.074	2.0×10^{-3}	1.2×10^{-2}	yes	−1.00
MobiTrust vs. B4	p95	141.799	125.000	2.0×10^{-3}	1.2×10^{-2}	yes	+1.00
MobiTrust vs. B9	p95	141.799	125.221	2.0×10^{-3}	1.2×10^{-2}	yes	+1.00

6.8. A Weight Sweep: The Weighted-Sum Controller Trades Offloading for Violations

A single high-violation weighting does not prove that no weighting works, so we sweep WeightedTrust's risk weight across three orders of magnitude (Table 13, Figure 10). To keep the sweep one-dimensional, we tie the two risk weights together, $w_P = w_A$; this is a representative slice of the weight space, not an exhaustive search over the full $(w_P, w_A, \lambda_e, \lambda_c)$ space, and the conclusions below are stated for the settings we tested. Along this slice, the privacy-violation rate tracks the offload rate almost exactly, because the weighted objective has no mechanism to tell a safe offload from a leaky one, it only trades total offloading against total risk. At the submitted $w = 60$, it violates privacy on 88% of inferences while offloading 88%; raising the weight to 120 halves both to ≈56%; and only at $w \geq 300$, where it has effectively stopped offloading (≈0% offload, i.e., collapsed onto FullLocal), does the violation rate reach zero. MobiTrust reaches zero violations while still offloading 70% of the time. We therefore do not claim that no scalar objective can ever reach this operating point; we show that none of the tested weightings does, and we explain the mechanism: a scalar penalty cannot encode a per-inference feasibility test, that makes the constrained formulation the natural way to reach it.

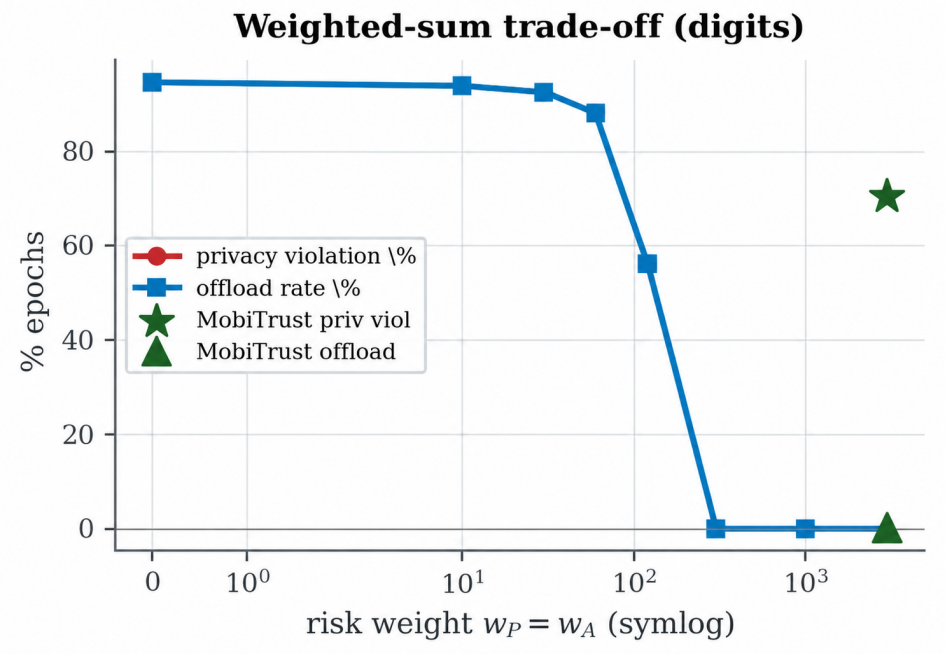


Figure 10. Weighted-sum risk-weight sweep (image). Privacy violations fall only as offloading falls; zero violations arrive only when offloading has ceased. MobiTrust (stars) attains zero violations at 70% offload.

Table 13. Weighted-sum risk-weight sweep (digits). As the risk weight grows, privacy violations fall only because the controller offloads less; by the time violations approach zero, it has essentially stopped offloading (offload $\to 0$), i.e., collapsed onto FullLocal. MobiTrust reaches zero violations while still offloading.

Risk Weight $w_P = w_A$	Priv Viol %	Adv Viol %	Offload %	p95 (ms)
0	94.5	58.8	94.5	131
10	93.9	47.6	93.9	128
30	92.6	20.2	92.6	127
60	88.1	6.8	88.1	126
120	56.0	0.0	56.0	125
300	0.0	0.0	0.0	125
1000	0.0	0.0	0.0	125
3000	0.0	0.0	0.0	125
MobiTrust (constraint)	0.0	0.0	70.5	142

6.9. Risk Bounds Under Estimation Uncertainty: The Dual Becomes Active

The zero-violation results above certify compliance with the profiled CDSE/LIR scores, not an independently validated attack probability (Section 4.4). We therefore examine what happens when the score can be wrong. We let the realised worst-channel leakage of an offloaded action be drawn around its cached CDSE estimate with uncertainty $\sigma = 0.08$ (a stronger or differently-seeded attacker than the profiling one) and compare two online rules: the pure feasibility filter (the nominal rule, with the privacy dual off) and the margin-tightened rule in which the privacy multiplier drives an effective-budget margin.

6.9.1. The Margin Rule, Precisely

The multiplier, margin, and effective budget are related as follows. The privacy multiplier $\mu_p \geq 0$ follows the same projected update as the other duals,

$$\mu_p \leftarrow \left[\, \mu_p + \beta \left(\mathbb{I}[\, \ell_t^{\text{real}} > \varepsilon_p \,] - \delta \right) \right]_+, \tag{16}$$

where ℓ_t^{real} is the realised worst-channel leakage at epoch t, ε_p the privacy budget, δ the tolerance and β the step. It induces a safety margin and an effective budget

$$m(\mu_p) = m_{\max} \left(2\sigma(\mu_p) - 1 \right), \qquad \varepsilon_p^{\text{eff}} = \varepsilon_p - m(\mu_p), \tag{17}$$

with $m_{\max} = 0.25$ the maximum tightening and σ the logistic function, so that $m = 0$ at $\mu_p = 0$ and grows toward $m_{\max}$ as violations accumulate. The only change to Algorithm 2 is that the privacy feasibility test uses $\varepsilon_p^{\text{eff}}$ in place of ε_p (and, symmetrically, $\varepsilon_a^{\text{eff}}$ for integrity); every other line is unchanged. When $\mu_p = 0$, this recovers the nominal feasibility filter exactly, which is why the nominal results and the A6 ablation are unaffected.

6.9.2. Where the Feedback Comes from

Equation (16) needs a realised-violation signal $\mathbb{I}[\ell_t^{\text{real}} > \varepsilon_p]$. In this study, that signal is available only through the simulation: we draw ℓ_t^{real} around the cached CDSE estimate with uncertainty $\sigma_{\text{unc}} = 0.08$ (modelling a stronger or differently-seeded attacker than the profiling one). We state plainly that this is a modelling device, not an operational measurement: realised leakage is not directly observable per inference in a real deployment. Making the mechanism deployable would require an auditing channel, for example, periodically running the attack suite on a sample of transmitted representations, or consuming breach reports from an external monitor, to supply the indicator in Equation (16). We therefore present the result below as evidence under the stated uncertainty model, not as a guarantee on the actual probability of a successful attack.

Table 14 and Figure 11 show the outcome: with the dual off, the realised privacy-violation rate is 27.5%, far above the 5% tolerance, because the filter admits actions whose point estimate is within budget but whose realised leakage is not, whereas with the dual on the margin tightens until the realised rate falls to 5.3% $\approx \delta$. This comes at a substantial, not modest, cost in availability: on the image workload offloading falls from 70.6% to 13.0%, because its feasible actions sit close to the budget so even a small margin excludes most of them (on sensing, which has more headroom, the drop is milder, 71% to 60%). The mechanism should therefore be read as a knob that trades off offloading for a bounded realised violation rate under the assumed uncertainty model, not as a free improvement and not as a bound on true attack probability. This is the setting in which the primal–dual mechanism is genuinely load-bearing.

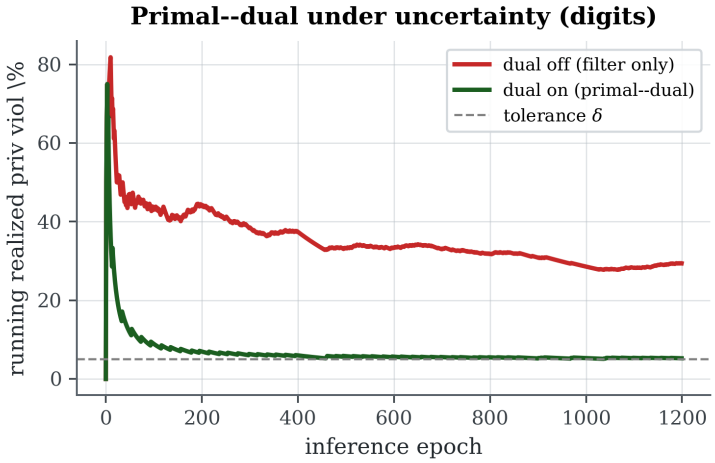


Figure 11. Primal–dual control under estimation uncertainty (image). The feasibility filter alone (dual off) lets the realised violation rate settle well above tolerance; the dual-driven margin (dual on) holds it at δ.

Table 14. Primal–dual control under estimation uncertainty (digits, realised worst-channel leakage drawn around the CDSE estimate with $\sigma = 0.08$). With the privacy dual off, the feasibility filter admits actions whose realised risk exceeds the budget more often than the tolerance; the dual-driven margin tightens the effective budget until the realised violation rate falls to δ, at a modest offloading cost.

Controller Variant	Realised Priv Viol %	Offload %
Feasibility filter only (dual off)	27.5	70.6
Primal–dual margin (dual on)	5.3	13.0
Tolerance δ	5.0	–

6.10. Robustness to the Cost Model

Because the cost model influences which split the controller selects (Section 5), we re-run the controller comparison under eight wide perturbations of the device/edge speeds, payload sizes, and per-split compute fractions (Table 15, Figure 12). Across all of them, MobiTrust keeps privacy and adversarial violations at 0%; only its fallback rate and tail latency move, and sensibly so, a device made twice as slow drives fallback from 30% to 87% and $p95$ from 142 to 250 ms, while a faster device does the reverse. The greedy and weighted baselines keep violating privacy (46–95%) throughout. The conclusion is a property of the constraint, not of the particular cost constants; CDSE's ranking advantage is cost-model-independent by construction, since leakage is measured on the network.

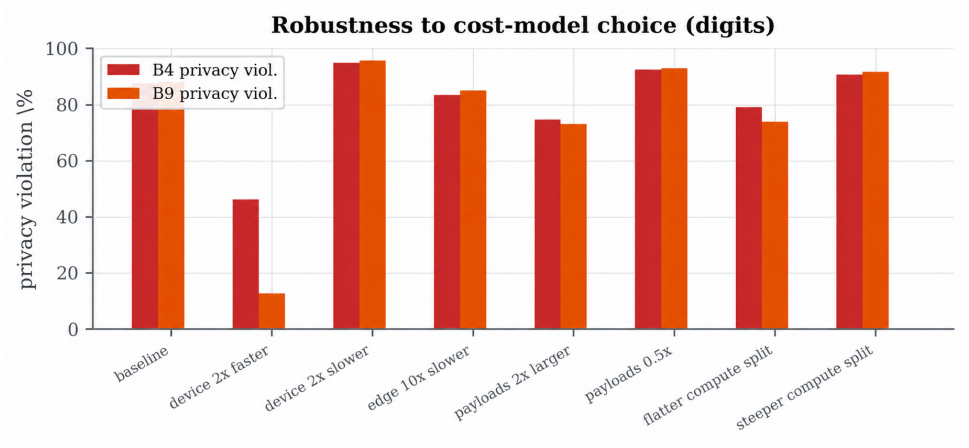


Figure 12. Robustness to cost-model choice (image). MobiTrust's privacy violations stay at zero across every perturbation; the baselines stay high.

Table 15. Cost-model sensitivity (digits). Across wide perturbations of device/edge speed, payload size, and the per-split compute fractions, MobiTrust (MT) keeps privacy and adversarial violations at zero, only its fallback rate and tail latency move, while the greedy (B4) and weighted (B9) baselines keep violating privacy. The conclusion is a property of the constraint, not of particular cost constants.

Cost-Model Scenario	MT Priv %	MT Adv %	MT FB %	MT p95	B4 Priv %	B9 Priv %
baseline	0.0	0.0	29.5	142	87.7	88.1
device 2x faster	0.0	0.0	10.9	75	46.3	12.7
device 2x slower	0.0	0.0	87.1	250	95.0	95.8
edge 10x slower	0.0	0.0	33.1	143	83.5	85.1
payloads 2x larger	0.0	0.0	40.3	142	74.8	73.1
payloads 0.5x	0.0	0.0	23.8	141	92.5	93.0
flatter compute split	0.0	0.0	26.2	140	79.2	73.9
steeper compute split	0.0	0.0	34.1	143	90.6	91.7

6.11. Route Trust Can Be Measured, Not Just Declared

Finally, we address the status of the base route-trust risk $\rho(e)$. Rather than leave it a hand-assigned input, a Beta–Bernoulli estimator driven by the integrity layer's observed per-edge flag rate recovers each edge's behaviour online (Section 4.3). We are careful about what this estimator recovers. A flag is not a tampering event: it conflates true tampering with the detector's false positives and misses, so the quantity that converges is an empirical flag-rate score $\hat{s}(e) \in [0, 1]$, the long-run fraction of interactions the integrity layer flags, not the underlying tampering probability, and not, in general, on the same scale as the policy value $\rho(e)$ for which Equation (5) was written.

This scale distinction matters for the constraint, and we are careful not to overstate it. Equation (5) multiplies $\rho(e)$ by a detector factor and compares the result to a fixed integrity budget ε_a, so whether an action is feasible depends on the numerical value substituted for $\rho(e)$, not only on how the edges rank. Preserving the edge ordering therefore does not by itself preserve which actions satisfy the constraint: substituting $\hat{s}(e)$ for a policy $\rho(e)$ can change the feasible set whenever the two live on different scales, even when they rank the edges identically. We thus do not claim the flag-rate score and the policy value are interchangeable. Putting the estimator in the loop requires either recalibrating ε_a to the flag-rate scale, or first mapping $\hat{s}(e)$ onto the policy scale, for instance by the deconvolution below, when the detector's error rates are known. What the experiment establishes is the narrower, accurate claim that the ordering of edge trust is recoverable online from observable flags; it does not establish that the raw score can replace $\rho(e)$ under an unchanged budget.

Table 16 and Figure 13 show $\hat{s}(e)$ converging, from live flags alone, to the stable flag rate of three edges with hidden tamper probabilities 0.02/0.20/0.45, with tight credible intervals and in the correct order. When the detector's false-positive and detection rates are known, the tampering probability itself can be recovered from the score by deconvolution, $\hat{p}_{\text{tamper}} = (\hat{s} - \text{fp})/(\text{detect} - \text{fp})$; the last column of Table 16 shows this estimate (0.03/0.23/0.45) matching the true probabilities. It is this deconvolved value, on the tampering-probability scale, that would be the principled input to a recalibrated Equation (5), not the raw flag-rate score; absent the error rates, we use only the recovered ordering.

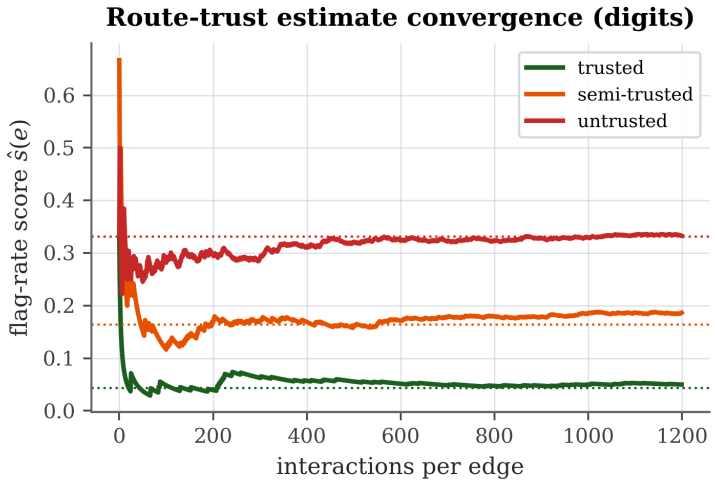


Figure 13. Behavioural route-trust estimation. The empirical flag-rate score converges from observed integrity-layer flags to each edge's stable flag rate (dotted), recovering the correct trust ordering. The tampering probability is recovered separately by deconvolution (Table 16, last column) when the detector's error rates are known.

Table 16. Behavioural route-trust estimation (digits integrity-layer parameters). The estimator recovers an empirical flag-rate score $\hat{s}(e)$, which conflates true tampering with the detector's false-positive and miss rates; it is this score (monotone in tampering) that supplies the ordering the route-trust budget needs. When the detector's false-positive and detection rates are known, deconvolution recovers the underlying tampering probability (last column), which matches the true values.

Edge	True Tamper Prob.	Flag-Rate Score $\hat{s}(e)$	95% CI	Deconv. $\hat{p}_{\text{tamper}}$
trusted	0.02	0.050	$\pm$0.012	0.030
semi-trusted	0.20	0.186	$\pm$0.022	0.233
untrusted	0.45	0.333	$\pm$0.027	0.452

6.12. Adaptation over a Trace

Figure 14 traces MobiTrust through a segment of a mixed trace, showing realised latency and the chosen split. When connectivity is good, the controller offloads to a feasible protected split; as the link degrades, it retreats to deeper splits and finally to local execution, keeping latency under the deadline and never crossing a risk budget. The behaviour is the visible signature of the feasibility filter and the conservative bandwidth prediction working together.

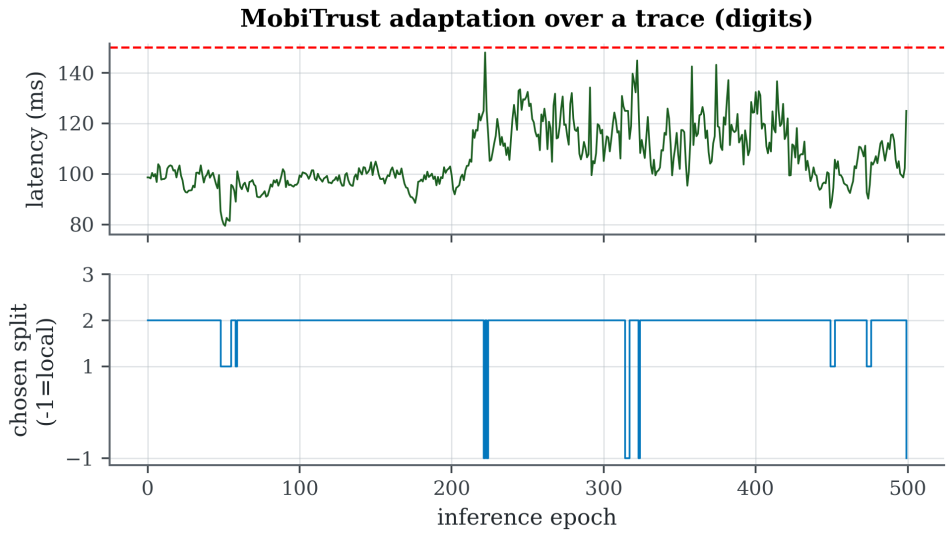


Figure 14. MobiTrust over a trace segment (image workload): realised latency against the deadline (**top**) and the chosen split, with −1 denoting local fallback (**bottom**). Offloading is used when it is safe and fast, and abandoned when it is not.

6.13. Where the Fallback Fires: Per-Regime Behaviour

The aggregate fallback rate hides useful structure, so Table 17 and Figure 15 break the controllers down by mobility regime. The pattern is the one the design intends. Under a good stationary link, MobiTrust never falls back; every inference finds a feasible protected offload, yet its privacy and adversarial violations are still zero, whereas the greedy and weighted baselines violate privacy on essentially every inference precisely because the good link makes a fast, leaky offload tempting. As connectivity degrades, the fallback rate climbs, to 27% under vehicular motion, 48% during handovers, and 85% under a congested stationary link, while the violation rate stays pinned at zero: MobiTrust buys its guarantee by declining to offload when no safe offload exists, not by loosening the budget. The baselines show no such adaptation, violating privacy in 65–100% of inferences across every regime. This is the per-regime signature of a constrained controller, and it is exactly what an operator needs to reason about availability under a given mobility profile.

Table 17. Per-regime behaviour (digits, violation and fallback rates in %, latency in ms). MobiTrust (MT) holds privacy and adversarial violations at zero in every regime, trading fallback rate for connectivity; the greedy (B4) and weighted (B9) baselines violate privacy throughout.

Regime	MT Priv	MT Adv	MT Fallback	MT p95	B4 Priv	B4 Adv	B9 Priv
Stat. good	0	0	0	105	100	4	100
Stat. cong.	0	0	85	144	65	65	68
Walking	0	0	5	137	98	71	96
Vehicular	0	0	27	145	87	69	87
Handover	0	0	48	146	79	77	80
Intermit.	0	0	15	144	90	86	90
Sudden cong.	0	0	33	125	94	41	93

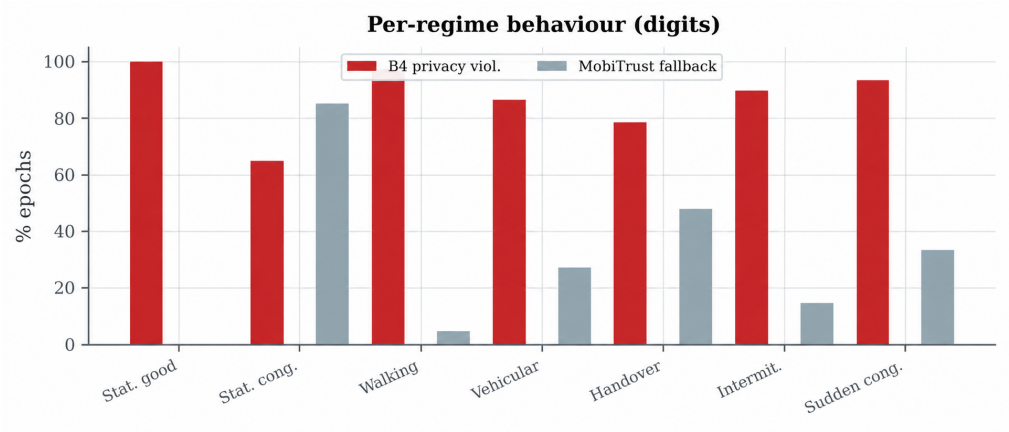


Figure 15. Per-regime behaviour on the image workload. MobiTrust's privacy violations (green, at zero everywhere) are invisible against the axis; its fallback rate (grey) rises as connectivity worsens, while the latency-only baseline (red) violates privacy across all regimes.

6.14. Sensing Workload

The sensing workload is the complementary case: here, attribute inference, not reconstruction, is the channel that binds, which lets us check that the mechanisms are not tuned to a single modality. Table 18 profiles the model. Attribute AUC is very high at shallow splits and falls with depth (from 0.99 at $k = 1$ to 0.87 at $k = 3$), so the sensitive orientation attribute is what a curious edge would read most easily, while reconstruction is comparatively weaker on this lower-dimensional signal.

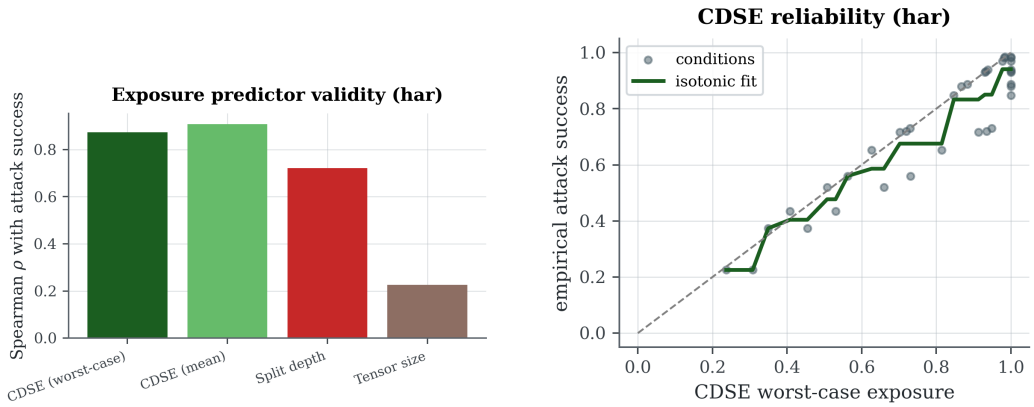
Table 18. Per-split profile on the har model (test accuracy 0.956). Reconstruction (SSIM/PSNR), membership (MI AUC, from a 24-shadow-model attack) and attribute (Attr AUC) leakage, and latent-integrity detectability (LIR AUROC). Membership leakage is split-independent because it derives from the final prediction.

Split	$\lvert z_k \rvert$	Client kMAC	Edge kMAC	SSIM	PSNR	MI AUC	Attr AUC	LIR AUROC
$k = 1$	128	4.6	10.4	0.900	25.6	0.569	0.992	0.569
$k = 2$	64	12.8	2.2	0.745	22.2	0.569	0.966	0.627
$k = 3$	32	14.8	0.2	0.595	20.7	0.569	0.865	0.688

Because the binding channel differs, the CDSE result is, if anything, sharper here. Table 19 shows CDSE tracking measured attack success at Spearman $\rho = 0.87$ against 0.72 for split depth and 0.23 for tensor size; attribute leakage is less monotone in depth than reconstruction was, so the depth heuristic loses more ground. Figure 16 shows the predictor comparison and the reliability curve.

Table 19. Predictive validity of exposure estimators against empirically measured worst-channel leakage (har, 36 operating conditions). CDSE ranks conditions far better than the structural heuristics used by leakage-agnostic schedulers.

Predictor	Spearman ρ	p	Kendall τ
CDSE (worst-case)	0.873	4.0×10^{-12}	0.743
CDSE (mean)	0.907	2.5×10^{-14}	0.789
Split depth	0.721	6.9×10^{-7}	0.607
Tensor size	0.226	1.8×10^{-1}	0.160



(**a**) (**b**)

Figure 16. CDSE on the sensing workload: predictor validity (left) and calibration against measured breach (right). The advantage over the depth heuristic is larger than on the image workload because attribute leakage is less depth-monotone. (**a**) Predictor validity. (**b**) CDSE reliability.

The latent-integrity picture carries over with the detector slightly weaker on this signal (Table 20): perturbation attacks are caught at AUROC around 0.63, the replay attack is again undetectable by LIR at 0.50 while flipping 83% of predictions, and the adaptive attacker again buys evasion only by dropping its flip rate to 0.04. The freshness verifier again closes the replay gap completely (100% detection, zero false positives; Table 21), so the combined defense is modality independent: perturbations to LIR, replay to freshness, and the adaptive attacker neutralised by its own effectiveness collapse.

Table 20. Latent-integrity detection at split $k = 2$ (har). Flip rate is the pre-detection attack success; the adaptive attack is detector-aware and is the hardest to catch.

Attack	Flip Rate	AUROC	AUPRC	FPR@95TPR
gaussian	0.046	0.636	0.602	0.822
fgsm	0.219	0.633	0.590	0.799
pgd	0.222	0.627	0.586	0.793
pgd-targeted	0.082	0.607	0.570	0.801
replay	0.833	0.500	0.500	0.950
adaptive	0.043	0.216	0.352	1.000

The controller comparison, Table 22, mirrors the image result closely. MobiTrust holds privacy and integrity violations at 0% with a fallback rate of 29.8%, an accuracy of 0.950 against the full model's 0.956, and a 95th-percentile latency of 142 ms, one millisecond off the oracle. The greedy and weighted baselines again violate privacy in about 88% of inferences and, for LatencyOnly, integrity in 58%. Figure 17 shows the violation rates and the privacy–latency trade-off.

Table 21. Replay defense at split $k = 2$ (HAR). Detection columns (LIR TPR, Combined TPR) are both true-positive rates at a 5% false-positive operating point, so they are directly comparable; LIR AUROC is a separate threshold-free summary. LIR cannot separate an in-distribution replay (AUROC ≈ 0.5, TPR near the 5% FPR floor); adding freshness verification lifts combined replay detection to 100%, while perturbation attacks remain LIR's responsibility.

Attack	Flip Rate	LIR AUROC	LIR TPR	Combined TPR
Replay	0.833	0.500	0.050	1.000
Perturbation (PGD)	0.222	0.627	0.065	0.065

Table 22. Controller comparison on HAR over 10 mixed-mobility traces of 1500 epochs each (means). MobiTrust holds privacy and adversarial violations near zero while keeping latency close to the leakage-agnostic baselines.

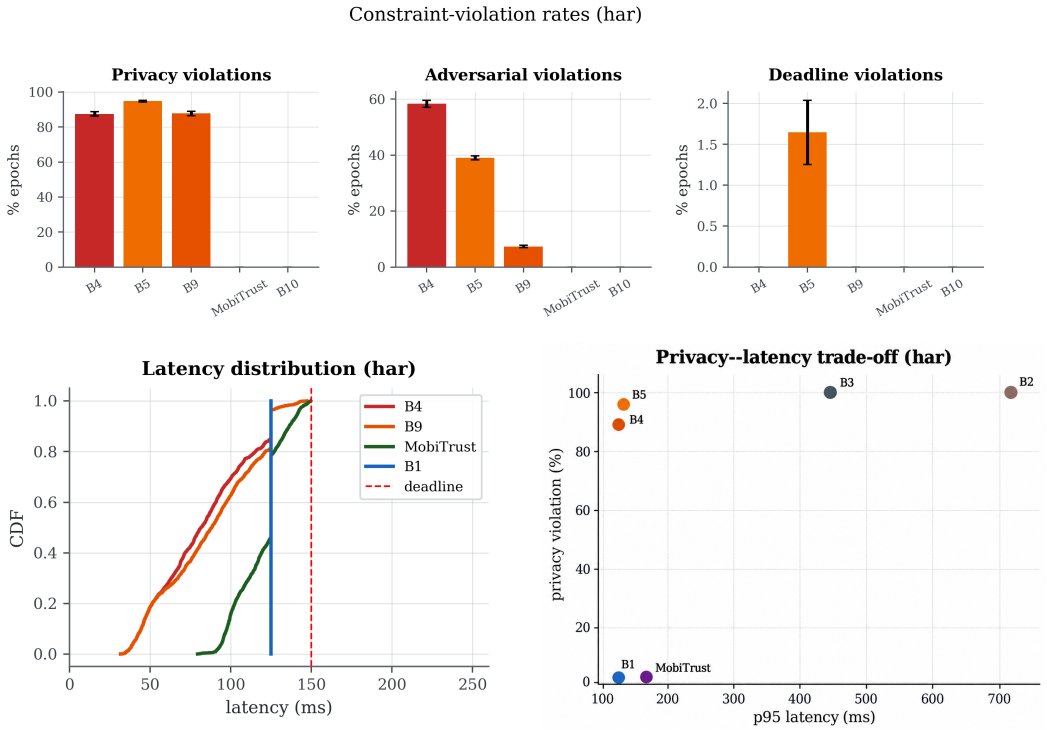
Controller	p95 (ms)	Energy (mJ)	Priv %	Adv %	Dline %	FB %	Acc
B1-FullLocal	125	312	0.0	0.0	0.0	0.0	0.956
B2-FullEdge	712	224	100.0	0.0	28.6	0.0	0.956
B3-StaticSplit	441	300	100.0	0.0	44.7	0.0	0.956
B4-LatencyOnly	125	110	87.5	58.4	0.0	0.0	0.956
B5-EnergyLatency	133	84	94.8	39.0	1.6	0.0	0.956
B9-WeightedTrust	125	102	87.8	7.4	0.0	0.0	0.956
MobiTrust	142	237	0.0	0.0	0.0	29.8	0.950
B10-Oracle	143	236	0.0	0.0	0.0	0.0	0.949

To assess the statistical significance of the observed performance differences, paired Wilcoxon signed-rank tests were conducted across 10 HAR traces, with Holm correction applied to account for multiple comparisons. Table 23 summarizes the results for privacy violations, integrity violations, and 95th-percentile latency (p95). MobiTrust consistently achieves significantly lower privacy and integrity violation rates than both B4 and B9 ($p_{\text{Holm}} = 0.012$), with Cliff's $\delta = -1.00$ indicating a maximal effect in favor of MobiTrust. However, MobiTrust exhibits significantly higher p95 latency ($p_{\text{Holm}} = 0.012$, Cliff's $\delta = +1.00$), highlighting the trade-off between stronger security guarantees and increased latency.

Table 23. Paired Wilcoxon signed-rank tests across 10 traces (har), Holm-corrected. Negative Cliff's δ means MobiTrust is lower (better) on violations.

Comparison	Metric	MobiTrust	Baseline	p	p_{Holm}	Sig.	Cliff δ
MobiTrust vs. B4	priv-viol	0.000	0.875	2.0×10^{-3}	1.2×10^{-2}	yes	−1.00
MobiTrust vs. B9	priv-viol	0.000	0.878	2.0×10^{-3}	1.2×10^{-2}	yes	−1.00
MobiTrust vs. B4	adv-viol	0.000	0.584	2.0×10^{-3}	1.2×10^{-2}	yes	−1.00
MobiTrust vs. B9	adv-viol	0.000	0.074	2.0×10^{-3}	1.2×10^{-2}	yes	−1.00
MobiTrust vs. B4	p95	141.799	125.000	2.0×10^{-3}	1.2×10^{-2}	yes	+1.00
MobiTrust vs. B9	p95	141.799	125.143	2.0×10^{-3}	1.2×10^{-2}	yes	+1.00

The ablation is where the two workloads most usefully diverge. On the image data, reconstruction-only privacy estimation (A2) was indistinguishable from full CDSE, because reconstruction was the binding channel. On the sensing data, it is not: Table 24 shows A2 violating the privacy budget in 78.2% of inferences, because a reconstruction-only estimate is blind to the attribute leakage that actually binds here and so declares attribute-leaky actions safe. This is the concrete justification for CDSE aggregating all three channels rather than tracking the one that happens to dominate a given dataset, the "binding" channel is a property of the deployment, not a constant, and an estimator that hard-codes it will fail when it changes. The depth heuristic (A1) fails on both workloads for the same reason as before, and dropping the integrity constraint (A3) again produces adversarial violations above 80%.



(**a**) (**b**)

Figure 17. Controller comparison on the sensing workload, matching the image result: among the *offloading* controllers only MobiTrust reaches the low-violation region occupied by the oracle. As on the image workload and in Table 22, FullLocal (B1) also attains zero violations, by never offloading; it is the safe no-offload reference rather than an offloading competitor. (**a**) Latency distribution. (**b**) Privacy–latency trade-off.

Table 24. Ablation study on har. Removing CDSE (A1), the adversarial constraint (A3), protection (A7) or edge selection (A8) each degrades a distinct guarantee. Mobility prediction (A4), the safe-fallback latency step (A5) and the temporal duals (A6) show no measurable effect in this nominal configuration—an expected consequence of exact, state-independent risk scores (Section 4.4); the duals become load-bearing under estimation uncertainty (Section 6.9).

Variant	p95 (ms)	Priv Viol %	Adv Viol %	Deadline Viol %	Fallback %	Acc
Full MobiTrust	142	0.0	0.0	0.0	29.8	0.950
A1: no CDSE (depth)	138	88.2	0.0	0.0	11.6	0.956
A2: reconstruction-only	140	78.2	0.0	0.0	21.7	0.903
A3: no LIR/adv constraint	143	0.0	80.7	0.0	18.3	0.949
A4: no mobility prediction	143	0.0	0.0	0.0	29.1	0.949
A5: no safe fallback	142	0.0	0.0	0.0	29.8	0.950
A6: no temporal duals	142	0.0	0.0	0.0	29.8	0.950
A7: no protection	134	0.0	0.0	0.0	60.2	0.956
A8: untrusted edge only	125	0.0	0.0	0.0	100.0	0.956

Figure 18 places the two workloads side by side on the four headline metrics. The controllers behave almost identically across modalities: the greedy and weighted baselines sit high on privacy violation on both, MobiTrust and the oracle sit at zero, and MobiTrust's tail latency and fallback rate are within a point or two of each other on image and sensing. The mechanisms are not tuned to a modality; what transfers is the behaviour of the constraint, not a set of numbers specific to one dataset.

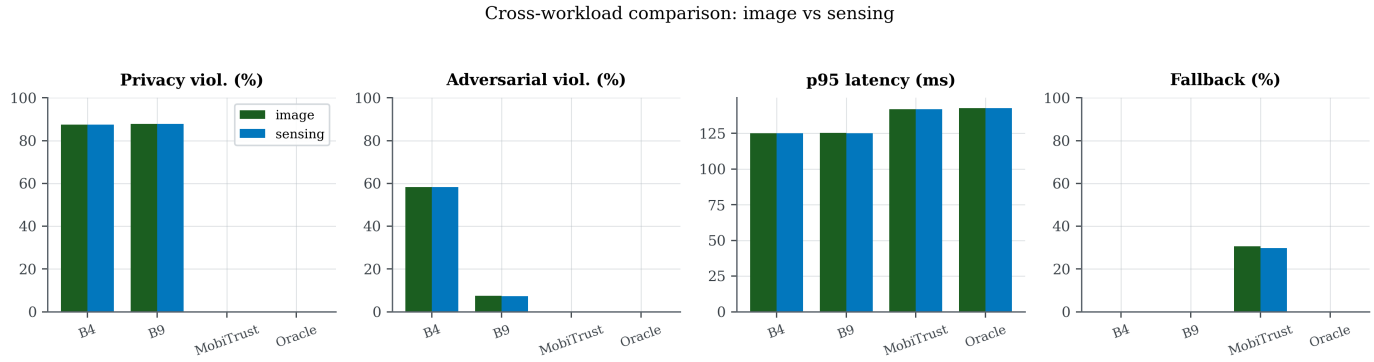


Figure 18. Cross-workload comparison on the four headline metrics. Image and sensing results agree closely: the leakage-agnostic baselines violate privacy and (for B4) integrity on both, while MobiTrust matches the oracle at zero violations with a modest latency and fallback cost.

7. Ablation Study

To attribute MobiTrust's behaviour to its parts, we disable one component at a time and re-run the mixed-trace evaluation. In every variant, the risk outcomes are still measured with the true CDSE and route model, so a component that merely hides risk from the decision, rather than removing it, shows up as increased measured violations. Table 25 reports the image workload.

Table 25. Ablation study on digits. Removing CDSE (A1), the adversarial constraint (A3), protection (A7) or edge selection (A8) each degrades a distinct guarantee. Mobility prediction (A4), the safe-fallback latency step (A5) and the temporal duals (A6) show no measurable effect in this nominal configuration—an expected consequence of exact, state-independent risk scores (Section 4.4); the duals become load-bearing under estimation uncertainty (Section 6.9).

Variant	p95 (ms)	Priv Viol %	Adv Viol %	Deadline Viol %	Fallback %	Acc
Full MobiTrust	142	0.0	0.0	0.0	30.6	0.898
A1: no CDSE (depth)	138	88.2	0.0	0.0	11.6	0.967
A2: reconstruction-only	142	0.0	0.0	0.0	30.6	0.898
A3: no LIR/adv constraint	143	0.0	80.5	0.0	19.5	0.887
A4: no mobility prediction	143	0.0	0.0	0.0	29.8	0.896
A5: no safe fallback	142	0.0	0.0	0.0	30.6	0.898
A6: no temporal duals	142	0.0	0.0	0.0	30.6	0.898
A7: no protection	125	0.0	0.0	0.0	100.0	0.967
A8: untrusted edge only	125	0.0	0.0	0.0	100.0	0.967

Four components are decisively load-bearing. Replacing CDSE with the split-depth heuristic (A1) sends the privacy-violation rate from 0% to 88.2%: the controller now believes deep unprotected splits are safe and offloads them, while their true reconstruction leakage is high. This is the clearest single demonstration that measured exposure, not structural depth, is what the constraint needs. Dropping the integrity constraint (A3) raises adversarial violations to 80.5%, as the controller routes to fast untrusted edges it no longer has any reason to avoid. Disabling protection (A7) or restricting the system to the untrusted edge (A8) drives the fallback rate to 100%: with no protected option, or no trusted edge, no offload can ever satisfy the budgets, so the controller correctly computes locally on every inference. Both results confirm that protection and edge selection are what make safe offloading possible at all.

The reconstruction-only privacy variant (A2) matches full CDSE on this workload, which is expected rather than disappointing: reconstruction is the binding channel for images, so ignoring the weaker membership and attribute channels changes nothing here. The distinction matters on the sensing workload, where the attribute channel binds and a reconstruction-only estimate would under-report exposure, the reason CDSE aggregates all three.

We are equally candid about the components that show little effect in this configuration. Removing mobility prediction (A4), the temporal duals (A6), or the safe fallback's latency-optimising step (A5) leaves the violation rates essentially unchanged on the mixed trace. The reason is structural: in this reduced-scale cost model, offload latency is either comfortably under the deadline (good link) or far over it (poor link), with little middle ground in which an optimistic prediction or an accumulated multiplier would tip a marginal decision. When we tighten the deadline below the local-execution time on a high-variance vehicular trace, the deadline becomes unsatisfiable for all variants including local, so that stress test does not cleanly isolate these components either. For the temporal duals (A6) the null result is not merely empirical but expected: with exact, state-independent risk scores the feasibility filter already keeps the selected action within budget, so the privacy/integrity violation indicators are zero and their multipliers never move (Section 4.4).

The duals are designed to correct estimation error, and Section 6.9 shows that once the risk scores carry uncertainty, the dual-driven margin is decisively load-bearing, cutting the realised violation rate from 27.5% to the 5% tolerance. We therefore do not claim the duals improve the nominal mixed-trace result, and we present A6 as the expected consequence of exact scores rather than as a disappointment. We likewise do not claim a benefit for mobility prediction (A4) or the fallback's latency step (A5) in this setting; we expect them to matter where the feasible frontier is genuinely latency-sensitive, which a full-scale deployment with large, bandwidth-bound activations would exhibit more strongly, and we leave that demonstration to hardware evaluation. Reporting these results faithfully is part of an honest ablation: four of MobiTrust's seven components (CDSE, the adversarial constraint, protection, edge selection) are load-bearing in the nominal configuration, the duals become load-bearing under uncertainty, and we decline to manufacture an effect for prediction and the fallback latency step.

8. Robustness and Sensitivity

The operating point in Section 5 is one choice among many, so we sweep the budgets and the attacker strength to check that MobiTrust's behaviour is a property of the constraint mechanism rather than of a particular setting. The sweeps use shorter traces and fewer seeds than the main comparison, which is adequate for trends.

8.1. Privacy Budget

Figure 19 varies the privacy budget ε_p from strict to permissive. MobiTrust's measured privacy-violation rate stays at essentially 0% across the whole range, by construction, since it never selects an action whose CDSE exceeds the budget, while the weighted-sum controller violates the budget in roughly 88% of inferences regardless of where the budget is set, because its scalar objective does not treat the budget as a limit at all. What moves for MobiTrust is the fallback rate: as the budget tightens, fewer protected splits qualify, and the controller retreats to local execution more often. This is the correct and legible response of a constrained system, safety held fixed, availability of offloading traded against the budget, and it is exactly the behaviour a weighted objective cannot produce.

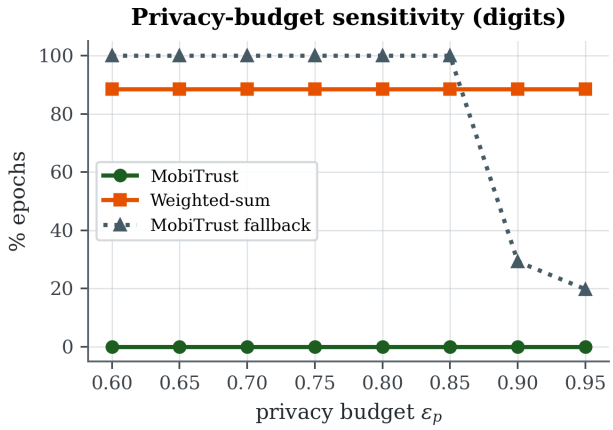
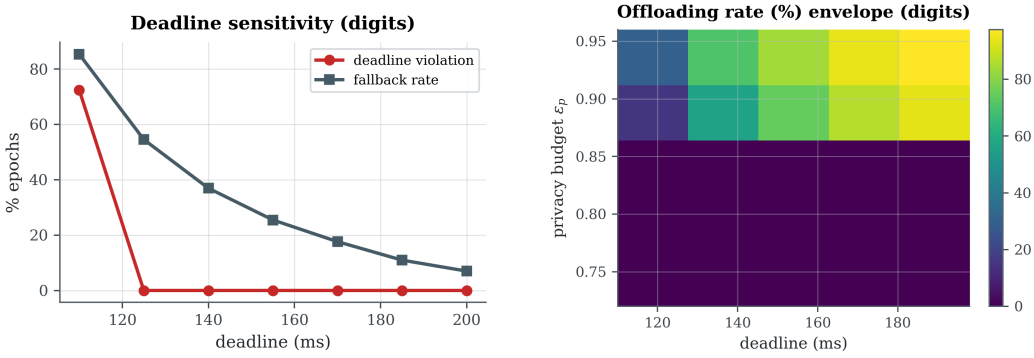


Figure 19. Privacy-budget sensitivity (image workload). MobiTrust's privacy violations stay near zero across the whole range while the weighted-sum baseline stays high; MobiTrust instead trades fallback rate for a tighter budget.

8.2. Deadline and Operating Envelope

Figure 20a sweeps the deadline. As it loosens, MobiTrust offloads more, and the fallback rate falls, with deadline violations held near their tolerance over the feasible range. There is one exception we flag explicitly: at the tightest setting (110 ms), the deadline is below the 125 ms local-execution latency. We are precise about what this does and does not mean. It does not make the whole action set infeasible: a fast offload can still beat a 110 ms deadline when connectivity is good, and in those states the controller meets it. What a sub-125 ms deadline removes is the guarantee of a feasible choice in the other states: whenever no offload happens to meet the deadline, the on-device fallback, the controller's safety net, at 125 ms, also misses it, so there is no deadline-feasible action at all and a violation is unavoidable.

The spike in Figure 20a is therefore the fallback losing its ability to meet the deadline in poor-connectivity states, not every offloading action becoming infeasible. For any deadline at or above the local-execution time, the fallback is always deadline-feasible, and the violation rate sits at its tolerance. Figure 20b maps the offloading rate over the joint space of privacy budget and deadline, giving the operating envelope: a bright region of frequent safe offloading when the budget is permissive and the deadline generous, shading to near-total local execution when either is strict. An operator can read the achievable offloading rate for a target risk posture directly from this map.



(**a**) (**b**)

Figure 20. Deadline sensitivity and the privacy-budget/deadline operating envelope (image workload). The envelope shows the offloading rate an operator can expect for a chosen risk posture. (**a**) Deadline sweep. (**b**) Operating envelope.

8.3. Attacker Strength

Figure 21 varies the latent attacker's budget. As perturbation strength grows, the flip rate of an unconstrained PGD attack rises and the detector's AUROC rises with it, since larger perturbations leave the benign manifold more visibly. The detector-aware adaptive attacker traces the opposite curve: it keeps AUROC low, but its flip rate stays near zero, so across the whole attacker-strength range, and across the four adaptive strategies of Section 6, we did not find an attack that is both evasive and effective against this detector. We state this as an empirical observation over the attacks and budgets we evaluated, not as a proof that no such attack exists: a stronger or structurally different adaptive attack could in principle shift the frontier, and establishing a worst-case bound would require a different, formal argument. The route-trust term, not the detector, remains the defense of last resort against the in-distribution replay attack, which no perturbation budget affects.

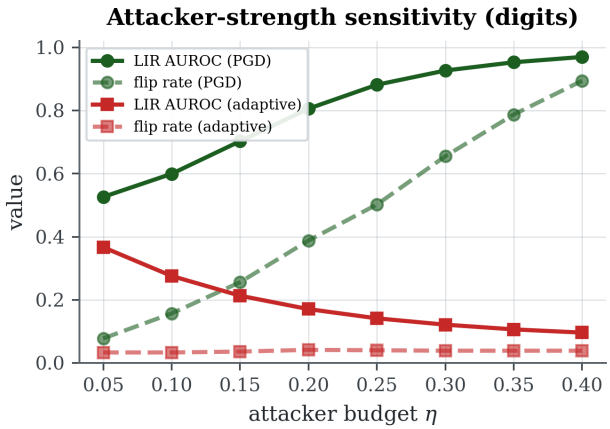


Figure 21. Attacker-strength sensitivity (image workload). Unconstrained PGD becomes both more effective and more detectable as its budget grows; the adaptive attacker stays undetected only by staying ineffective.

9. Discussion

9.1. What the Results Support

The evidence backs a narrow but useful claim. When privacy and integrity are written as per-inference constraints and the leakage that fills them is measured rather than guessed from structure, an adaptive split-inference controller can hold those risks at their budget across a mobility trace, at a quantified and modest cost in latency and energy, and can come within a millisecond of a clairvoyant oracle at the tail. The comparison with the weighted-sum controller is the sharpest lesson: a scalar objective that prices risk highly still fails on most individual inferences, because averaging is the wrong operation for a guarantee that has to hold each time. The comparison with the depth heuristic is the second lesson: structural proxies for leakage misrank options once protection is available, so a scheduler that wants to bound privacy has to pay the offline cost of measuring it. Two qualifications keep the claim honest.

First, "holding risk at budget" means compliance with the profiled CDSE/LIR scores under the assumed route model, not a certified attack probability; where the scores carry uncertainty, it is the primal–dual margin (Section 6.9), not the filter alone, that bounds the realised violation rate. Second, the useful comparison is against the offloading controllers: local-only execution also attains zero violations, with higher accuracy and lower tail latency, so MobiTrust's contribution is to offload safely and save energy, not to beat on-device inference outright. Because this is a controlled simulation study, the quantified "cost in latency and energy" is a property of the analytical model; what transfers is the controller's

decision behaviour, which the cost-model sensitivity study (Section 6.10) shows is invariant to the cost constants.

9.2. Limitations

The most important limitation is scale. Everything here runs on CPU at reduced scale, with a multilayer perceptron standing in for a deep network and an analytical model standing in for hardware latency and energy. This was a deliberate choice in favour of full reproducibility and clean isolation of the constraint machinery, and the claims we make are chosen to be the ones that scale argument does not threaten; they concern the controller's decisions behaviour effect on measured risk, not absolute milliseconds. Still, several quantities would change on real systems.

The absolute latencies and energies are only as good as the cost model. The attack strengths are those of the specific decoders, membership classifiers, and probes we trained; a stronger attacker raises the measured leakage and would shift the feasible set, though not the logic that responds to it. The risk budgets that make the feasible set interesting were calibrated to our measured ranges; on another model the same budgets might be trivially loose or tight, which is why we sweep them.

Mobility prediction and the fallback's latency step showed little effect on the mixed trace, for the structural reason given in Section 7, and we do not claim otherwise. The temporal duals are a special case we have now resolved: their nominal null effect is the expected consequence of exact risk scores, and Section 6.9 shows they become load-bearing precisely when scores are uncertain, so we present them as an uncertainty-correction mechanism, not as a nominal-case contribution. On the integrity side, route trust $\rho(e)$ is an operator policy by default, but Section 6.11 shows it can be estimated from observed edge behaviour, which narrows this limitation to the choice of whether to trust a policy label or a behavioural estimate.

The latent detector, as we show, is blind to in-distribution replay, but this is not left unaddressed: the freshness verifier of Equation (6) catches every exact replay at zero false-positive cost, so the residual integrity gaps are narrower than the detector alone would suggest. Two honest caveats remain. First, freshness assumes a working key on the device and trusted edge and a per-inference authentication tag; it does not defend an untrusted edge that is the legitimate recipient, which is what the route-trust term is for. Second, on route trust, we are careful to separate what the flag-rate estimator of Section 6.11 already provides from what remains open. That estimator recovers the trust ordering of edges online from observed integrity-layer flags, so route trust is no longer purely a hand-assigned input; this part is demonstrated, not future work. What it does not yet provide is a calibrated value on the policy scale that could be substituted into Equation (5) under an unchanged budget ε_a: as Section 6.11 explains, the flag-rate score and the policy value live on different scales, so a self-calibrating integrity budget would additionally require recalibrating ε_a to the score scale (or mapping the score onto the policy scale, e.g., by deconvolution when the detector's error rates are known). That calibration is the genuine remaining gap, and we state it as such rather than as the absence of any estimator.

9.3. Threats to Validity

9.3.1. Construct Validity

CDSE is only as meaningful as the attacks that define its channels; if a real adversary has a materially stronger inversion or probe than ours, the measured exposure is an underestimate. We mitigate this by using independent held-out attackers to define the target in the calibration study, so that CDSE is judged on generalisation rather than on

the attacker it was fitted to, but the point stands that CDSE inherits the strength of its attack suite.

9.3.2. Internal Validity

The controller comparison could be confounded if the baselines were weakened relative to MobiTrust. We guard against this by giving the weighted-sum baseline weights that price risk as highly as latency, and by including a clairvoyant oracle as an upper bound; MobiTrust's closeness to the oracle shows the greedy baselines' violations are a property of their objective, not of a handicap.

9.3.3. External Validity

The two workloads are small, and one is synthetic, and the mobility traces are generated rather than measured. The mechanisms are not specific to these choices; nothing in CDSE, LIR, or the controller assumes a modality or a trace source, but we cannot claim the specific numbers transfer, only that the qualitative behaviour is stable across the budget and attacker sweeps we ran. The cost model is the largest external-validity risk, and the code is structured so that replacing it with hardware traces is a localised change.

9.4. Implications

For practitioners, the practical message is that split inference should not be sold as private without a leakage measurement behind the claim, and that if privacy or integrity matters, they belong in the feasibility test, not the objective. The offline profiling that CDSE and LIR require is a one-time cost that buys a per-inference guarantee, and the online overhead is a handful of table lookups. For researchers, the CDSE calibration result suggests that leakage estimation deserves the same treatment as any other predictor, measured, calibrated, and validated on held-out attackers, rather than being replaced by a structural shortcut, and the replay result is a reminder that manifold-based integrity detection has a definite blind spot, one that a cheap freshness check covers cleanly, which is a useful division of labor: distance-based detectors for off-manifold perturbations, authentication for on-manifold replay.

10. Conclusions and Future Work

We set out to make privacy and representation integrity into limits that an adaptive split-inference controller respects on every inference, rather than terms it averages against speed. The two ingredients were measurement and constraint: Context-Dependent Semantic Exposure to estimate privacy leakage from the strongest of several channels under a declared context (with membership measured by a shadow-model attack we validate is strong), Latent Integrity Risk together with a route-trust model to score, rather than certify a bound on, the chance of a corrupted prediction (Section 4.4: the integrity quantities are admission scores, not validated security bounds), and a feasibility-filtering controller that selects split, protection, and edge as the cheapest action satisfying every budget, with a safe on-device fallback. On a fully reproducible reduced-scale testbed, which we present as a controlled study of the constraint mechanism, not a deployment system, the parts did what they were meant to. CDSE ranked operating conditions by measured attack success far better than the split-depth and tensor-size heuristics that leakage-agnostic schedulers rely on. The controller held the profiled privacy and integrity violations at zero across mixed mobility traces where a latency-only controller violated privacy in 88% of inferences and integrity in 58%, and where a weight sweep showed the weighted objective, across the weightings we tested, reached zero violations only by ceasing to offload. A local-only baseline also attains zero violations, so MobiTrust's gain over it is energy (-24%) from safe offloading rather than a uniform win; its gain over the offloading baselines is that it offloads

while respecting the budgets, which none of them does. Where the risk scores are uncertain, a primal–dual margin holds the realised violation rate at tolerance, and the cost-model and route-trust studies show the conclusions survive changes to the cost constants and that the edge trust ordering can be measured rather than declared. We also reported where the approach falls short, two components whose benefit did not materialise at this scale, because those findings are part of the result, and we closed the one integrity gap we could: a freshness check that catches every exact replay the latent detector misses.

The clearest next step is hardware. The cost model is isolated so that real latency and energy traces from phones, single-board computers, and a live radio can replace it without touching the risk logic, and doing so would test the two claims we deliberately did not make here: absolute performance, and the value of mobility prediction and the temporal duals in a regime with a genuinely latency-sensitive feasible frontier. On the integrity side, freshness already closes the exact-replay gap, and the flag-rate estimator of Section 6.11 already recovers the edge trust ordering from observed behaviour rather than a hand-assigned label; what remains is narrower, a calibrated route-trust value on the policy scale (or a budget ε_a recalibrated to the flag-rate scale) so that the estimate can enter Equation (5) without changing the feasible set, which would make the integrity budget truly self-calibrating. On the privacy side, CDSE would benefit from a stronger and more diverse attack suite, since it is only ever as conservative as the attacks that define it, and from contexts learned from a deployment's data-protection policy rather than set by hand. Finally, the constraint framework is not specific to split inference; the same measure-then-bound discipline could govern other edge decisions where a resource objective is quietly allowed to override a risk that should have been a limit.

11. Notation

Table 26 collects the symbols used throughout.

Table 26. Summary of notation.

Symbol	Meaning
f, L	classifier and its number of layers
k	split point; $k = 0$ raw offload, $k = L$ local
$z_k, \tilde{z}_k$	representation at split k, before/after protection
$g_p, \mathcal{P}$	protection map of mode p; set of modes
$e, \mathcal{E}$	serving edge; set of reachable edges
$a = (k, p, e)$	an action; $\mathcal{A}$ the action set
$n_t = (b_t, \tau_t, q_t)$	network state: bandwidth, RTT, queue
$\hat{n}_t$	conservatively predicted state
ℓ, η, ϕ	latency, energy, scalar operating cost
$s_{\text{rec}}, s_{\text{mem}}, s_{\text{attr}}$	reconstruction/membership/attribute leakage
$w(c)$	context channel weights
r_{priv}	CDSE privacy exposure, Equation (3)
r_{adv}	integrity risk, Equation (5)
$\rho(e)$	route-trust base risk of edge e
AUROC_k	LIR detector power at split k
ΔU	utility loss of protection
$\varepsilon_p, \varepsilon_a, \varepsilon_u, D$	privacy/integrity/utility/deadline budgets
δ_{C}	tolerated violation rate of constraint C
μ_{C}	Lagrange multiplier for constraint C
α, β, λ_e	EWMA rate, dual step, energy weight

12. Hyperparameters

Table 27 lists the settings used for all experiments unless a sweep varies them. Values were fixed before the main comparison and not tuned against it.

Table 27. Hyperparameters and operating point.

Group	Parameter	Value
Model	hidden widths	128, 64, 32
	candidate splits	after each hidden layer
	epochs/batch/lr	60/64/0.05
	weight decay	10^{-4}
Protection	noise levels (rel. σ)	0.35, 1.2
	DP-style noise (rel. σ)	2.2
	quantisation	4-bit
	bottleneck keep fraction	0.35
Cost model	workload MACs	3×10^8
	device/edge rate (MAC ms^{-1})	$2.4 \times 10^6 / 2.4 \times 10^8$
	activation size by split (KB)	280, 70, 10
	device compute fraction	0.25, 0.60, 0.88
	powers (compute/tx/idle, mW)	2500/800/50
Budgets	$\varepsilon_p, \varepsilon_a, \varepsilon_u, D$	0.88, 0.30, 0.18, 150 ms
	δ (risk/deadline)	0.05/0.10
Controller	EWMA rate α	0.4
	dual step β	0.05
	energy weight λ_e	0.5
Integrity	LIR autoencoder	shallow, benign-trained
	freshness nonce-cache window	256
Attacks	shadow models n_{sh}	24
	adaptive steps/λ	15–30/2.0
	latent budget η	0.15
Uncertainty	leakage uncertainty σ	0.08
	max dual margin	0.25
Evaluation	trace length/seeds	1500/10
	bootstrap resamples	10^4

13. Additional Results

The released outputs contain the full set of tables and figures for both workloads, including the sensing-workload profile, protection, CDSE-correlation, LIR, controller-main, controller-statistics, and ablation tables, and the corresponding figures. The metrics file records every reported quantity in machine-readable form, and the experiment scripts regenerate all of it end to end.

Author Contributions: Conceptualisation, S.A.-E. and Y.S.; methodology, S.A.-E. and Y.S.; software, S.A.-E.; validation, Y.S., M.A. and M.A.A.; formal analysis, S.A.-E. and M.A.A.; investigation, S.A.-E.; resources, M.A. and Y.S.; data curation, S.A.-E.; writing—original draft preparation, S.A.-E.; writing—review and editing, Y.S., M.A. and M.A.A.; visualisation, S.A.-E.; supervision, Y.S. and M.A.; project administration, Y.S.; funding acquisition, M.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Gulf University for Science and Technology, Kuwait.

Data Availability Statement: The image workload derives from the publicly available Digits dataset included with scikit-learn. The sensing workload is synthetic and is generated deterministically by the released code from a fixed seed. No proprietary or personal data were used. The complete reference implementation, the split network, the attack suite (including the shadow-model membership attack and the four adaptive attacks), the CDSE and LIR components, the freshness verifier, the route-trust

estimator, the controllers, the analytical cost model, and the scripts that regenerate every table and figure in this paper will be deposited in a public repository upon acceptance. Every quantitative claim in this paper is produced by the released code under fixed random seeds. Latency and energy figures come from the transparent analytical cost model described in Section 5, not from hardware measurement; this is stated wherever such figures appear. Results were obtained on CPU at reduced scale, and the limitations of that scope are set out in Section 9.2.

Acknowledgments: The authors acknowledge the support of the Gulf University for Science and Technology, Kuwait. During the preparation of this manuscript, the authors did not use any AI-assisted tools, including generative AI or AI-based language-editing tools. The authors prepared, reviewed, and edited all aspects of the manuscript and take full responsibility for its content.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Osia, S.A.; Taheri, A.; Shamsabadi, A.S.; Katevas, K.; Haddadi, H.; Rabiee, H.R. Deep private-feature extraction. *IEEE Trans. Knowl. Data Eng.* **2018**, *32*, 54–66. [CrossRef]
2. Shuvo, M.M.H.; Islam, S.K.; Cheng, J.; Morshed, B.I. Efficient Acceleration of Deep Learning Inference on Resource-constrained Edge Devices: A Review. *Proc. IEEE* **2022**, *111*, 42–91. [CrossRef]
3. Ye, D.; Shen, S.; Zhu, T.; Liu, B.; Zhou, W. One Parameter Defense—Defending against Data Inference Attacks via Differential Privacy. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 1466–1480. [CrossRef]
4. Huang, H.; Luo, W.; Zeng, G.; Weng, J.; Zhang, Y.; Yang, A. DAMIA: Leveraging Domain Adaptation as a Defense against Membership Inference Attacks. *IEEE Trans. Dependable Secur. Comput.* **2021**, *19*, 3183–3199. [CrossRef]
5. Liu, J.; Lyu, X.; Cui, Q.; Tao, X. Similarity-based Label Inference Attack against Training and Inference of Split Learning. *IEEE Trans. Inf. Forensics Secur.* **2024**, *19*, 2881–2895. [CrossRef]
6. Pham, N.D.; Abuadbba, A.; Gao, Y.; Phan, K.T.; Chilamkurti, N. Binarizing Split Learning for Data Privacy Enhancement and Computation Reduction. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 3088–3100. [CrossRef]
7. Alhindi, A.; Al-Ahmadi, S.; Ben Ismail, M.M. Advancements and challenges in privacy-preserving split learning: Experimental findings and future directions. *Int. J. Inf. Secur.* **2025**, *24*, 125. [CrossRef]
8. Xu, Z.; Zhao, L.; Liang, W.; Rana, O.F.; Zhou, P.; Xia, Q.; Xu, W.; Wu, G. Energy-aware Inference Offloading for DNN-driven Applications in Mobile Edge Clouds. *IEEE Trans. Parallel Distrib. Syst.* **2020**, *32*, 799–814. [CrossRef]
9. Tang, X.; Chen, X.; Zeng, L.; Yu, S.; Chen, L. Joint Multi-user DNN Partitioning and Computational Resource Allocation for Collaborative Edge Intelligence. *IEEE Internet Things J.* **2020**, *8*, 9511–9522. [CrossRef]
10. Liu, T.; Ni, S.; Li, X.; Zhu, Y.; Kong, L.; Yang, Y. Deep reinforcement learning based approach for online service placement and computation resource allocation in edge computing. *IEEE Trans. Mob. Comput.* **2022**, *22*, 3870–3881. [CrossRef]
11. Zhang, M.; Shen, X.; Cao, J.; Cui, Z.; Jiang, S. EdgeShard: Efficient LLM Inference via Collaborative Edge Computing. *IEEE Internet Things J.* **2024**, *12*, 13119–13131. [CrossRef]
12. Zhang, H.; Wang, J.; Zhang, H.; Bu, C. Security computing resource allocation based on deep reinforcement learning in serverless multi-cloud edge computing. *Future Gener. Comput. Syst.* **2024**, *151*, 152–161. [CrossRef]
13. Zhang, S. Model collaboration framework design for space-air-ground integrated networks. *Comput. Netw.* **2025**, *257*, 111013. [CrossRef]
14. Wang, L.; Wu, W.; Dai, M.; Zhang, H. Joint Resource Allocation and Multi-Task Multi-Access Computation Offloading in Vehicular Edge Networks: An Incentive Approach. *IEEE Trans. Veh. Technol.* **2026**, *75*, 20645–20660. [CrossRef]
15. Li, J.; Shi, Y.; Dai, C.; Yi, C.; Yang, Y.; Zhai, X.; Zhu, K. A learning-based stochastic game for energy efficient optimization of UAV trajectory and task offloading in space/aerial edge computing. *IEEE Trans. Veh. Technol.* **2025**, *74*, 9717–9733. [CrossRef]
16. Li, J.; Yi, C.; Chen, J.; Shi, Y.; Zhang, T.; Li, X.; Wang, R.; Zhu, K. A reinforcement learning-based stochastic game for energy-efficient UAV swarm-assisted MEC with dynamic clustering and scheduling. *IEEE Trans. Green Commun. Netw.* **2024**, *9*, 255–270. [CrossRef]
17. Oh, Y.; Lee, J.; Brinton, C.G.; Jeon, Y.S. Communication-efficient split learning via adaptive feature-wise compression. *IEEE Trans. Neural Netw. Learn. Syst.* **2025**, *36*, 10844–10858. [CrossRef]
18. Wu, W.; Li, M.; Qu, K.; Zhou, C.; Shen, X.; Zhuang, W.; Li, X.; Shi, W. Split Learning over Wireless Networks: Parallel Design and Resource Management. *IEEE J. Sel. Areas Commun.* **2023**, *41*, 1051–1066. [CrossRef]
19. Kou, Y. Federated Learning Approaches for Privacy-Preserving Big Data Analytics. *J. Comput. Electron. Inf. Manag.* **2026**, *20*, 8–17. [CrossRef]

20. Zhang, X.; Zhang, H.; Zhang, G.; Yang, Y.; Li, F.; Fan, L.; Huang, Z.; Cheng, X.; Hu, P. Membership inference attacks against incremental learning in iot devices. *IEEE Trans. Mob. Comput.* **2024**, *24*, 4006–4021. [CrossRef]
21. Xu, C.; Ren, J.; Zhang, D.; Zhang, Y.; Qin, Z.; Ren, K. GANobfuscator: Mitigating Information Leakage under GAN via Differential Privacy. *IEEE Trans. Inf. Forensics Secur.* **2019**, *14*, 2358–2371. [CrossRef]
22. Wu, M.; Cheng, G.; Ye, D.; Kang, J.; Yu, R.; Wu, Y.; Pan, M. Federated split learning with data and label privacy preservation in vehicular networks. *IEEE Trans. Veh. Technol.* **2023**, *73*, 1223–1238. [CrossRef]
23. Altman, E. *Constrained Markov Decision Processes*; Chapman and Hall/CRC: Boca Raton, FL, USA, 1999; Volume 7.
24. Eisen, M.; Zhang, C.; Chamon, L.F.O.; Lee, D.D.; Ribeiro, A. Learning Optimal Resource Allocations in Wireless Systems. *IEEE Trans. Signal Process.* **2019**, *67*, 2775–2790. [CrossRef]
25. Ale, L.; King, S.A.; Zhang, N.; Sattar, A.R.; Skandaraniyam, J. D3PG: Dirichlet DDPG for Task Partitioning and Offloading with Constrained Hybrid Action Space in Mobile-Edge Computing. *IEEE Internet Things J.* **2022**, *9*, 19260–19272. [CrossRef]
26. Mohammadi, S.; Akbari, B. Priority-Aware Resource Reallocation in Edge Computing Using Reinforcement Learning. *J. Netw. Syst. Manag.* **2025**, *34*, 20. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.