Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Familiarity, Not Detection: Rethinking Machine-Learning Hardware Trojan Detection Under Leakage-Free Side-Channel Evaluation

Mani Rupak Gurram

mgurram@pvamu.edu

Prairie View A&M University

Mithun Kumar PK

Wright State University

Sarita Bista

Prairie View A&M University

Fathi Amsaad

Wright State University

Akshay "AK" Raghavendra Kulkarni

Prairie View A&M University

Research Article

Keywords: Hardware Trojan detection, side-channel analysis, machine learning, evaluation methodology, data leakage, generalization, Trust-HUB

Posted Date: November 10th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10872309/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Familiarity, Not Detection: Rethinking Machine-Learning Hardware Trojan Detection Under Leakage-Free Side-Channel Evaluation

Mani Rupak Gurram[1*], Mithun Kumar PK[2], Sarita Bista[1], Fathi Amsaad[2], Akshay Raghavendra Kulkarni[1]

[1]Department of Computer and Electrical Engineering, Prairie View A&M University, 700 University Dr, Prairie View, 77446, Texas, USA.
[2]Department of Computer Science and Engineering, Wright State University, 3640 Colonel Glenn Hwy, Dayton, 45435, Ohio, USA.

*Corresponding author(s). E-mail(s): mgurram@pvamu.edu;
Contributing authors: pk.2@wright.edu; sbista@pvamu.edu;
fathi.amsaad@wright.edu; arkulkarni@pvamu.edu;

Abstract

Machine-learning detectors for hardware Trojans (HTs) from power side-channel traces are widely reported to reach 87–100% accuracy on the standard Trust-HUB AES benchmarks. We show that these figures largely reflect *Trojan familiarity rather than Trojan detection*: they are obtained under random trace-level splits that place traces of the *same* Trojan in both training and test. We re-evaluate the problem under a protocol that is leakage-free by construction, partitioning strictly at the Trojan level, and report five findings on the Trust-HUB AES power dataset (12 Trojans, AES-T400–T2000). First, within a benchmark, Trojan activity is detectable for 11 of 12 Trojans (mean ROC-AUC **0.84**), but the discriminative signal is diffuse and low-SNR, so global-average-pooling CNN/Transformer backbones common in this literature collapse to chance. Second, zero-shot detection of an *unseen Trojan family* falls to **0.53** ROC-AUC—below chance for several Trojans—a **0.30** drop that is robust to an independent, taxonomy-based family definition. Third, the collapse is a controlled *donor* effect: one Trojan's detectability swings from **0.97** to **0.00** depending solely on whether a single related Trojan is present in training. Fourth, the failure is neither a capacity nor a representation problem—nine model families, including a hybrid CNN-attention network, self-supervised contrastive pretraining, and a selective state-space (Mamba-style) model, all remain at the wall—and it does not follow

the functional taxonomy by which Trojans are categorized (payload/trigger type explain $< \mathbf{1\%}$ of transfer variance). Fifth, once a handful of the target chip's own labels are available, the simplest detector—nearest-centroid on those labels—beats every cross-Trojan transfer scheme, while pooling other Trojans actively hurts. We conclude that side-channel HT detection cannot be deployed as a universal pretrained model and requires per-chip calibration, and we recommend Trojan-disjoint evaluation as standard practice.

Keywords: Hardware Trojan detection, side-channel analysis, machine learning, evaluation methodology, data leakage, generalization, Trust-HUB.

1 Introduction

Modern integrated circuits (ICs) are produced at enormous volume and distributed worldwide, where they end up in critical infrastructure, medical devices, avionics, and military systems. A malicious modification hidden inside such a chip—a *hardware Trojan* (HT)—can leak secret keys, degrade performance, or disable a system on command, and the consequences of a single compromised design propagating through the supply chain can be severe and large-scale. This risk is not hypothetical: fabrication of a modern IC is not performed in a single trusted facility but spans many parties—design houses, third-party IP vendors, foundries, and test and packaging services—and a Trojan can be inserted at any one of these stages. The breadth of this attack surface makes pre-deployment screening of fabricated chips for HTs a first-order security requirement rather than an optional safeguard.

Because post-fabrication logic testing can only observe a Trojan that is actually triggered—and stealthy Trojans are engineered to activate under rare, attacker-chosen conditions—non-invasive *side-channel analysis* (SCA) has become a primary detection modality. SCA exploits the fact that anomalous circuitry perturbs a chip's physical emissions (power, current, or electromagnetic radiation) even while dormant, allowing a defender to reason about the presence of an HT from measured traces without decapsulating the die or accessing the netlist. Building on the public Trust-HUB AES power/EM benchmarks, a substantial machine-learning (ML) literature reports strong HT detection accuracy: Siamese networks at 83–87%, stacked ensembles up to ∼99–100%, and reinforcement-learning approaches at ∼84%.

We argue, and demonstrate empirically, that these accuracies substantially overstate real-world detection capability, because they are obtained under evaluation protocols that *leak*. The prevailing practice pools all traces from all Trojans and draws a random train/test split. Since each Trojan contributes many near-duplicate traces—repeated measurements of the same circuit differing only by acquisition noise—this places traces of the *same* Trojan on both sides of the split. A classifier can then achieve a high score by recognizing a Trojan whose near-identical measurements it has already seen during training. This is not the task a screening tool actually faces in deployment: there, the Trojan present on the device under test is, by assumption, one that was never characterized beforehand—inserted somewhere in a supply chain

the defender does not control. Measuring performance on Trojans already present in training therefore rewards *familiarity* rather than *detection*, and conflates the two.

The distinction is decisive, and it can be tested directly. When we train a detector on 11 of the 12 Trust-HUB AES Trojans and evaluate it on the 12th—held out entirely, as a genuinely unseen Trojan would be—detection collapses from the within-benchmark regime to near chance. The impressive published numbers, in other words, largely evaporate the moment the evaluation reflects the deployment condition they are meant to certify. Given the stakes—chips screened as "clean" on the basis of such numbers may nonetheless carry undetected Trojans into critical systems—this gap between reported and deployable accuracy is precisely what a security-critical screening pipeline cannot afford.

In this work we re-evaluate ML-based side-channel HT detection under a protocol that is leakage-free by construction, partitioning strictly at the Trojan level, and we systematically characterize how detectability degrades as increasingly strict splits remove increasingly subtle forms of leakage. Rather than proposing yet another detector, we ask what the existing evaluation actually measures, expose the mechanism behind the inflated numbers, and delineate the conditions under which side-channel HT detection can and cannot work. Our contributions are:

1. A **leakage-free evaluation protocol** for side-channel HT detection—leave-one-Trojan-out (LOTO) and leave-one-family-out (LOGO) splits, with all fitted preprocessing estimated on training Trojans only—together with an open-source implementation to support reproducible, deployment-faithful evaluation.
2. **The generalization wall, measured.** Within-benchmark detectability averages 0.84 ROC-AUC, but zero-shot detection of an *unseen Trojan family* falls to 0.53—below chance for several Trojans—a 0.30–0.43 gap that quantifies how much of prior accuracy is attributable to familiarity.
3. A **robustness check that rules out circularity.** The wall persists (gap 0.30) when Trojan families are defined by an independent Trust-HUB payload taxonomy rather than by clustering the transfer data, confirming that the effect is not an artifact of how families are drawn.
4. A **controlled mechanism.** A single-donor knock-out shows detectability is transfer from a previously seen relative: AES-T700 scores 0.971 when a related Trojan is present in training and 0.000 when it is removed—the same Trojan, flipped by the presence of one relative.
5. **Two negative results that sharpen the claim.** The wall is architecture- and representation-independent—nine model families, including a hybrid CNN-attention network, self-supervised contrastive pretraining, and a selective state-space (Mamba-style) model, all remain at it—and transferability is not governed by the functional taxonomy by which Trojans are conventionally categorized (payload and trigger type explain $<1\%$ of transfer variance).
6. A **deployment characterization.** Once a small number K of per-chip calibration labels from the device under test are available, a standardized nearest-centroid detector on the target's own support outperforms every cross-Trojan transfer scheme, and pooling additional Trojans *reduces* accuracy—indicating that reliable

side-channel HT screening requires per-chip calibration rather than a universal pretrained model.

The remainder of this paper is organized as follows. Section 2 reviews prior ML-based side-channel HT detection and situates our evaluation critique. Section 3 states the threat model. Section 4 details the leakage-free protocol. Section 6 presents the results, and Sections 7–10 discuss implications, limitations, and conclusions.

2 Background and Related Work

Hardware-Trojan (HT) detection methods divide into three broad families: logic testing that seeks to *activate* the Trojan, side-channel analysis that detects its *passive physical footprint*, and, more recently, machine-learning (ML) detectors trained on side-channel or structural features. We review each, then scope our contribution precisely to the branch our critique concerns.

2.1 Logic-testing and trigger-activation methods

The earliest defenses adapt manufacturing test to HTs by generating input patterns that activate rarely-exercised logic. The canonical example is MERO [1], which applies *multiple excitation of rare occurrence*: it constructs a compact vector set that drives each rare node to its rare value many times, raising the probability of triggering an embedded Trojan so that a payload or output deviation becomes observable. Path-delay fingerprinting [2] takes a complementary route, characterizing a chip's delay signatures and flagging deviations induced by inserted logic. These methods target the *activation* problem and are largely orthogonal to side-channel detection; notably, they too are challenged by genuinely novel Trojans whose triggers are not exercised by the generated patterns—a difficulty analogous to, but mechanistically distinct from, the generalization limits we study here.

2.2 Classical statistical side-channel analysis

Side-channel analysis (SCA) detects an HT's passive perturbation of a chip's power, current, timing, or electromagnetic (EM) emissions. The seminal approach, IC fingerprinting [3], builds power side-channel fingerprints from a set of trusted (golden) chips using subspace projection and flags devices whose signatures fall outside the trusted manifold. Because a golden reference is often unavailable and process variation can mask small Trojans, subsequent work reduced the reliance on golden references and improved robustness to noise. Multiple-parameter SCA [4, 5] combines several correlated measurements (e.g. transient supply current and path delay) into a joint Trojan signature, improving separation of Trojan effects from process variation. Self-referencing [6] eliminates the golden chip entirely by comparing a region of a chip against *itself* at different times or against structurally identical on-die regions, so that process drift cancels while a localized Trojan does not. Chip-free EM statistical analysis [7] extends this golden-free, statistical philosophy to local EM measurements. These methods are statistical hypothesis tests rather than learned classifiers: they assume either a golden reference or an intra-chip self-reference, and they do not involve a

train/test partition over a corpus of Trojans. Our threat model deliberately assumes neither a golden copy of the device under test nor design access, which places these methods outside our evaluation but motivates the learned, golden-free detectors we examine next.

2.3 Machine-learning side-channel detectors

To remove the golden-reference requirement and learn Trojan signatures directly from data, a growing body of work applies ML to the public Trust-HUB benchmarks [8, 9]. Faezi *et al.* [10] propose a golden-chip-free detector that stacks heterogeneous base learners (random forest, gradient boosting, naïve Bayes, DNN, LSTM, and a graph model) and release the Trojan power/EM side-channel dataset used throughout this paper. Nasr *et al.* [11] introduce a Siamese metric-learning framework over power traces, reporting Siamese-LSTM 86.78%, Siamese-GRU 83.59%, and Siamese-CNN 73.54%. Reinforcement-learning approaches generate discriminative test conditions and report up to 84.2% [12]. Chen et al. [13] further apply an attention-enhanced ResNeXt architecture to power side-channel traces, reporting 97% average classification accuracy across Trust-HUB AES Trojan benchmarks under randomly sampled training and validation traces. Jiang and Ding [14] apply contrastive representation learning to power side-channel HT detection and report improved transfer to unseen Trojan variants compared with supervised learning. These golden-free ML detectors are the direct subject of our study: most prior evaluations emphasize within-benchmark trace-level performance, while recent work has explored transfer to selected unseen Trojan variants. However, systematic Trojan-disjoint and family-disjoint evaluation of power side-channel detectors remains limited.

2.4 Structural (netlist-level) ML detection

A parallel line applies ML to gate- or register-transfer-level representations rather than physical measurements, e.g. graph neural networks over circuit netlists [15] and supervised classifiers over gate-level structural features. While the input modality differs from ours, this literature has reported a two-regime pattern—high accuracy within a circuit family and sharp degradation across families—that parallels the side-channel generalization wall we establish and mechanistically explain in this work. Broad surveys [16, 17] catalogue the full HT threat and defense landscape.

2.5 Scope of this work

Our critique concerns the *golden-free ML side-channel* branch (Sec. II-C)—the methods trained and evaluated on Trust-HUB power traces. The logic-testing and classical-statistical methods above are not evaluated here: the former solve a different (activation) problem, and the latter presuppose a golden or self-reference that our threat model excludes; both lack the train/test partition whose leakage our analysis targets. We do not dispute any prior result within its own protocol. Rather, we show that the pooled, trace-level evaluation shared by the ML branch credits Trojan *familiarity* rather than detection, and we quantify what remains under a leakage-free, Trojan-disjoint protocol.

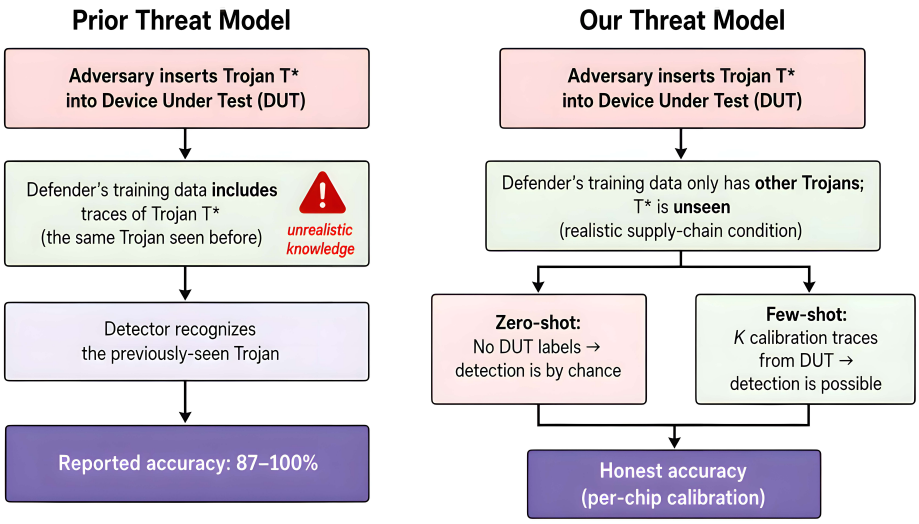


Fig. 1 Threat model. Left: trace-level random-split evaluation implicitly assumes the defender has already characterized the device's own Trojan $T^\star$ ($T^\star \in$ train), so a detector can succeed by recognizing a previously-seen Trojan—an unrealistic capability that inflates reported accuracy. Right: our threat model assumes $T^\star$ (and its family) is unseen at training time ($T^\star \notin$ train). Zero-shot detection then falls to chance; only a small set of K per-chip calibration traces from the device under test restores usable accuracy.

3 Threat Model

We make the adversary and defender assumptions explicit, because the central claim of this paper—that reported accuracies measure Trojan *familiarity* rather than detection—depends entirely on which knowledge the defender is assumed to have at training time.

3.1 Adversary

The adversary inserts a hardware Trojan (HT) into the AES design at design or fabrication time. The HT consists of a *trigger* (a rare activation condition, e.g. a specific plaintext, plaintext sequence, or internal counter value) and a *payload* (the malicious effect, e.g. key leakage through a power/RF side channel, elevated leakage current, or denial of service). The Trojan is designed to be stealthy: its incremental power footprint is small relative to baseline switching activity and remains largely dormant until triggered. Crucially, we assume the adversary is *not* restricted to Trojans the defender has previously characterized: the specific Trojan present on a given device may be novel and need not resemble any Trojan in the defender's training corpus. This is the realistic supply-chain condition and is precisely the assumption that trace-level random-split evaluation silently violates.

3.2 Defender

The defender performs post-fabrication, non-invasive side-channel analysis (SCA): they can acquire power traces from a device under test (DUT) during AES encryption but cannot decap, probe, or read back the netlist. The defender's goal is a binary decision—HT-active vs. clean—from these traces. We assume the defender possesses:

- a labeled training corpus of side-channel traces from a *set of known* Trojan-infected and Trojan-free designs (as in the Trust-HUB dataset); and
- optionally, a small number (K) of labeled reference traces from the DUT itself, obtained through a calibration step (per-chip calibration).

The defender does *not* have access to a golden (Trojan-free) copy of the exact DUT, nor to the DUT's design files; detection must rely on the side channel alone.

3.3 Evaluation assumption under test

Prior evaluations implicitly grant the defender a stronger form of knowledge: by pooling all traces and splitting randomly, they allow traces of the *same* Trojan to appear in both training and test, i.e. they assume the DUT's Trojan (or a near-identical measurement of it) was already characterized during training. We regard this as an unrealistic capability and remove it: in our threat model the DUT's Trojan is unseen (LOTO), and in the strictest setting no Trojan of the same family has been seen (LOGO). The per-chip calibration labels (K traces from the DUT) are the only DUT-specific information the defender is permitted, reflecting a realistic in-field calibration rather than prior knowledge of the attacker's design.

3.4 Scope and out-of-scope

In scope: detection of an active HT's presence from power side-channel traces of an AES core under the above knowledge constraints. Out of scope: Trojan *localization* or netlist identification; invasive or optical inspection; defenses that assume a trusted golden reference of the exact DUT; and adaptive adversaries that deliberately shape the side channel to evade a *known* detector (we assume the adversary is stealthy but not detector-aware). These restrictions match the capabilities exercised by the datasets and prior methods we re-evaluate, ensuring a like-for-like comparison.

4 Leakage-Free Evaluation Methodology

4.1 The leakage in trace-level evaluation

Each Trojan in the dataset contributes many power traces that are near-duplicates of one another—the same circuit measured repeatedly, differing only by acquisition noise. The prevailing practice pools all traces across all Trojans and draws a random train/test split. As illustrated in Fig. 2 (left), this places traces of the *same* Trojan on both sides of the split. A model can then classify a test trace by matching it to a near-identical training trace of the same Trojan, i.e. by recognizing a Trojan it has effectively already seen. This inflates reported accuracy but does not measure the

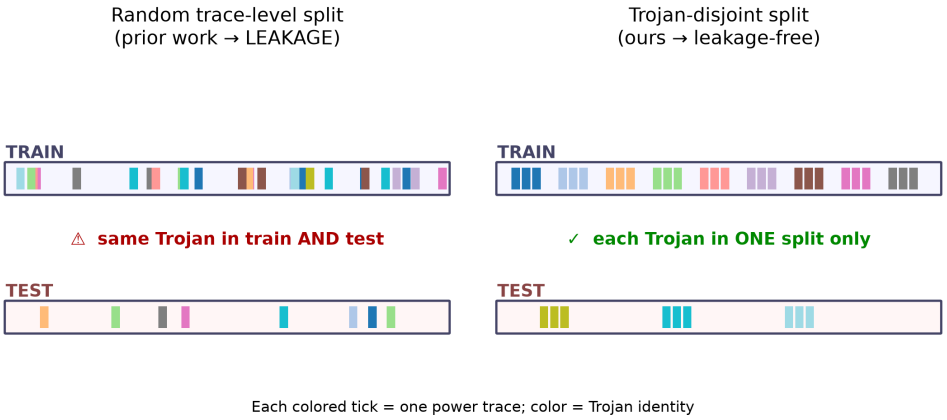


Fig. 2 Trace-level vs. Trojan-level splitting. Each tick is one power trace, colored by Trojan identity. A random trace-level split (left) puts the same Trojan in train and test, so recognition of a near-duplicate inflates accuracy. Our Trojan-disjoint split (right) assigns every Trojan to a single partition, removing this leakage.

deployment task, in which the Trojan on the device under test is, by assumption, absent from training.

4.2 Trojan-level partitioning

Our protocol makes the *Trojan* the atomic unit of the split: all traces of a given Trojan are assigned entirely to one partition (Fig. 2, right). No Trojan appears in both training and test. All fitted preprocessing—feature standardization in particular—is estimated on training Trojans only and then applied to held-out Trojans, so no information crosses the split. We use no synthetic oversampling; class balance is handled by construction of the evaluation episodes.

4.3 Evaluation regimes

On this leakage-free foundation we define three regimes of increasing strictness (Fig. 3):

- **Within-benchmark ceiling.** For each Trojan in isolation, a 5-fold cross-validated ROC-AUC measures how detectable that Trojan is from its *own* traces—an upper bound on what any cross-Trojan method could achieve for it.
- **Leave-one-Trojan-out (LOTO).** The model is trained on 11 Trojans and tested on the held-out one. In the few-shot variant the held-out Trojan supplies K labeled support traces (per-chip calibration); in the zero-shot variant it supplies none.
- **Leave-one-family-out (LOGO).** An entire Trojan *family* is held out, so the test Trojans have no close relative in training. This is evaluated both zero-shot and few-shot and is the strictest, most deployment-realistic setting.

4.4 Transfer structure and family definitions

To characterize *why* generalization succeeds or fails, we compute a Trojan-to-Trojan *transfer matrix*: entry (i, j) is the ROC-AUC of a detector trained on Trojan i and tested on Trojan j. To guard against circularity, we define Trojan families two independent ways: (a) by hierarchical clustering of the transfer matrix (data-driven), and (b) by the Trust-HUB payload taxonomy—power side-channel leak (PSC), radio-frequency leak (RF), leakage current (LC), and denial-of-service (DoS)—which is defined entirely from Trojan function and is independent of any measured transfer. Reporting the wall under both definitions separates a genuine effect from a clustering artifact.

4.5 Detectors and metrics

Except where a specific architecture is under test (Sec. 6.6), the detector is a standardized linear/nearest-centroid classifier over the raw trace, chosen because the leakage signal is diffuse and approximately linear; results are robust to substituting stronger backbones. We report ROC-AUC and PR-AUC as threshold-free primary metrics, plus F1 and true-positive rate at 1% false-positive rate, each as a per-Trojan mean with 95% confidence intervals. Traces are subsampled to 300 per state per input scheme; conclusions are stable at 2000 (Sec. 9).

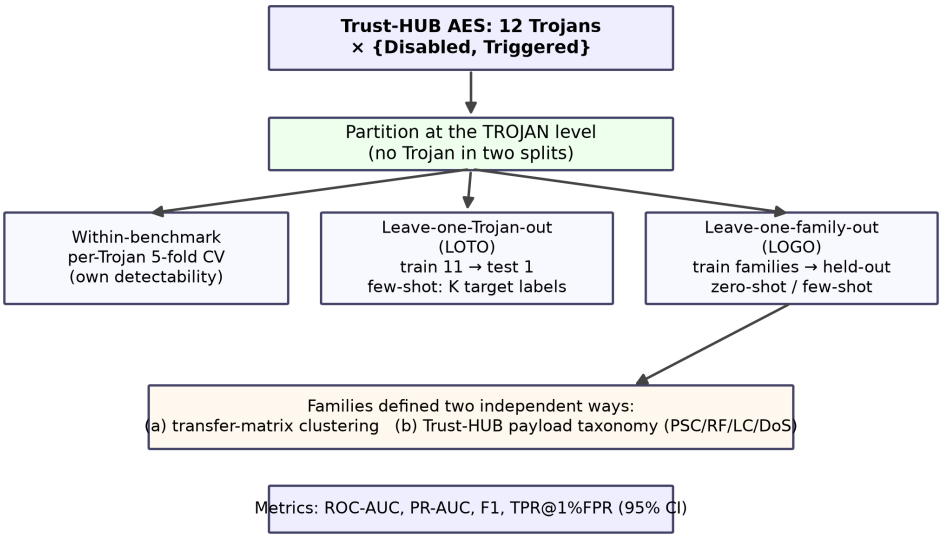


Fig. 3 Leakage-free evaluation protocol. The dataset is partitioned at the Trojan level, then evaluated under three regimes of increasing strictness (within-benchmark, LOTO, LOGO), with families defined two independent ways to rule out circularity.

5 Experimental Setup

This section documents the dataset, preprocessing, detectors, protocol implementation, hyperparameters, and compute environment in full, so that every reported number is reproducible from the released code.

5.1 Dataset

We use the Trust-HUB AES-128 power side-channel dataset [10, 18], distributed via IEEE-DataPort . It comprises twelve Trojan-infected AES-128 variants—AES-T400, T500, T600, T700, T800, T1000, T1100, T1300, T1400, T1600, T1800, and T2000. Each variant is captured in three operating states: *Disabled* (the Trojan is present but switched off; the golden-free clean reference), *Enabled* (present but inactive), and *Triggered* (actively executing its payload). Each state is recorded under two input schemes (denoted `_1` and `_2`), and every acquisition is a single power trace of 2500 samples measured at 25°C. There are approximately 10,000 traces per (Trojan, state, scheme) group . The detection task throughout is the binary discrimination of *Triggered* vs. *Disabled* traces (active-HT detection); the *Enabled* state is retained in the corpus but not used as a class label. No electromagnetic traces, alternate temperatures, or additional circuits are available in this release, which bounds the study to a single modality, temperature, and circuit (Sec. 9).

5.2 Preprocessing and trace representation

Each trace is used in its raw 2500-sample form; we apply no filtering, alignment, or spectral transform, so that the evaluation reflects the signal as acquired. Feature standardization (per-sample z-scoring) is the only fitted preprocessing: its mean and standard deviation are estimated *on the training Trojans only* and then applied to held-out Trojans, ensuring no statistic crosses the split (Sec. 4). To bound memory and runtime on commodity hardware, each (Trojan, state, scheme) group is randomly subsampled to 300 traces; we verified that the qualitative conclusions and headline numbers are stable when this cap is raised to 2000 (Sec. 9). We apply no synthetic oversampling or class re-balancing; class balance is enforced by the construction of evaluation episodes (below). A per-run cache of the assembled trace tensor is stored to disk so that repeated evaluations reuse an identical sample.

5.3 Detectors and architectures

Except where an architecture is itself under test (Sec. 6.6), the default detector is a standardized linear / nearest-centroid classifier over the raw trace, chosen because the leakage signal is diffuse and approximately linear (Sec. V-A). For the architecture-independence study we additionally evaluate a family of backbones behind a common training and evaluation harness:

- **Logistic regression** on standardized traces (`max_iter`= 2000).
- **MLP**: a standardized, non-pooling multilayer perceptron over the flat trace, producing a 128-dim embedding.

- **Convolution–transformer (SCAFormer)**: a 1-D convolutional stem followed by self-attention.
- **Hybrid CNN-attention**: a standardized linear path fused with a multi-scale convolutional path using statistical (mean/std/max) and attention pooling.
- **Selective state-space (Mamba-style)**: a strided-convolution stem, an input-dependent selective state-space block (plain-PyTorch selective scan; no fused CUDA kernel), and statistical pooling.
- **Prototypical network** for few-shot episodic evaluation.
- **Self-supervised contrastive encoder**: SimCLR-style pretraining with an NT-Xent objective, used as a frozen feature extractor.

For the leakage-ladder analysis we additionally use k-nearest-neighbors and a random forest as representative flexible classifiers. All embedding backbones emit a 128-dimensional representation and receive standardized input.

5.4 Protocol implementation

The three regimes of Sec. 4 are implemented as follows. The *within-benchmark* ceiling is a stratified 5-fold cross-validated ROC-AUC per Trojan. *Leave-one-Trojan-out* (LOTO) and *leave-one-family-out* (LOGO) iterate the held-out unit over all Trojans / families. Few-shot evaluation is episodic: each episode draws a K-shot support set and a Q-query set (with Q=15 per class) from the held-out Trojan, builds class prototypes from the support, and scores the queries; we average ROC-AUC over 100–200 episodes per fold. Zero-shot evaluation supplies no target labels: a detector trained on the source Trojans/families is applied directly to the held-out unit. The transfer matrix trains a detector on each single Trojan and evaluates it on every other. Family definitions are (a) average-linkage hierarchical clustering of the symmetrized transfer matrix, and (b) the fixed Trust-HUB payload taxonomy (PSC/RF/LC/DoS).

5.5 Hyperparameters

Table 1 lists the hyperparameters used across experiments. They were not tuned for peak accuracy—the study's conclusions concern the *gap* between regimes, which is insensitive to fine tuning—so we deliberately use modest, uniform settings across backbones for a like-for-like comparison.

5.6 Metrics and statistical methodology

We report ROC-AUC and PR-AUC as threshold-free primary metrics, since the operating threshold does not transfer across Trojans (Sec. V); we additionally report F_1 and the true-positive rate at a 1% false-positive rate (TPR@1%FPR) as operating-point summaries, noting that the latter is high-variance in the cross-Trojan regime. Aggregate figures are per-Trojan (or per-fold) means with 95% confidence intervals; because the results are bimodal across Trojans, we report per-Trojan values rather than relying on the mean alone. Episodic metrics are averaged over the episode count above with a fixed seed for repeatability.

Table 1 Hyperparameters. Settings are uniform across backbones for a like-for-like comparison and were not tuned for peak accuracy.

Parameter	Value
Trace length	2500 samples
Traces per group (subsample)	300 (stable at 2000)
Embedding dimension	128
Few-shot shots K	20 (swept $\{1, 5, 10, 20, 50\}$)
Query per class Q	15
Episodes per fold	100–200
Cross-validation (within)	5-fold stratified
Optimizer	AdamW
Learning rate	1×10^{-3}
Weight decay	1×10^{-4}
Batch size	128
ProtoNet training steps	$\sim$300
Contrastive pretraining steps	300–800
Contrastive temperature τ	0.2
Contrastive augmentations	jitter, scaling, shift, masking
Logistic regression `max_iter`	2000
Random forest trees	200
k-NN neighbors	5
Random seed	fixed per run [VERIFY]

5.7 Implementation, compute, and reproducibility

All experiments run on a single commodity CPU—a 14-core Intel Core Ultra 7 165U (15.1 GB RAM), with no GPU acceleration—using Python 3.12.3 with PyTorch 2.13.0 (CPU execution), scikit-learn 1.9.0, NumPy 2.5.0, and SciPy 1.18.0. The selective state-space backbone uses a plain-PyTorch selective scan rather than a fused CUDA kernel, so it runs without a GPU at the cost of speed. All randomness is controlled by a fixed seed: the deterministic components of the study—the per-Trojan within-benchmark ceilings, the cross-Trojan transfer matrix, and the zero-shot leave-one-Trojan-out and leave-one-family-out analyses—regenerate bit-identically across repeated runs, while the episodic few-shot metrics regenerate within the reported 95% confidence intervals. We verify these guarantees with an included reproducibility script that reports the execution environment, confirms seed-controlled determinism of the NumPy, PyTorch, and scikit-learn components, and re-runs the headline analyses to check that the regenerated values match those reported here. We release the complete implementation—data loading, leakage-free splitting, all backbones, the transfer and few-shot analyses, and the figure- and table-generation scripts—together with the fixed seeds and the cached trace tensor, so that every number and figure in this paper can be regenerated end-to-end.

6 Results

6.1 Within a benchmark, activity is detectable but diffuse

A standardized linear model separates Triggered from Disabled on 11/12 Trojans (mean ROC-AUC **0.836**; AES-T1100 an outlier at 0.593). The largest single-sample class-mean difference is $d \approx 0.15$, yet aggregating all 2500 samples reaches ≈ 0.81: the

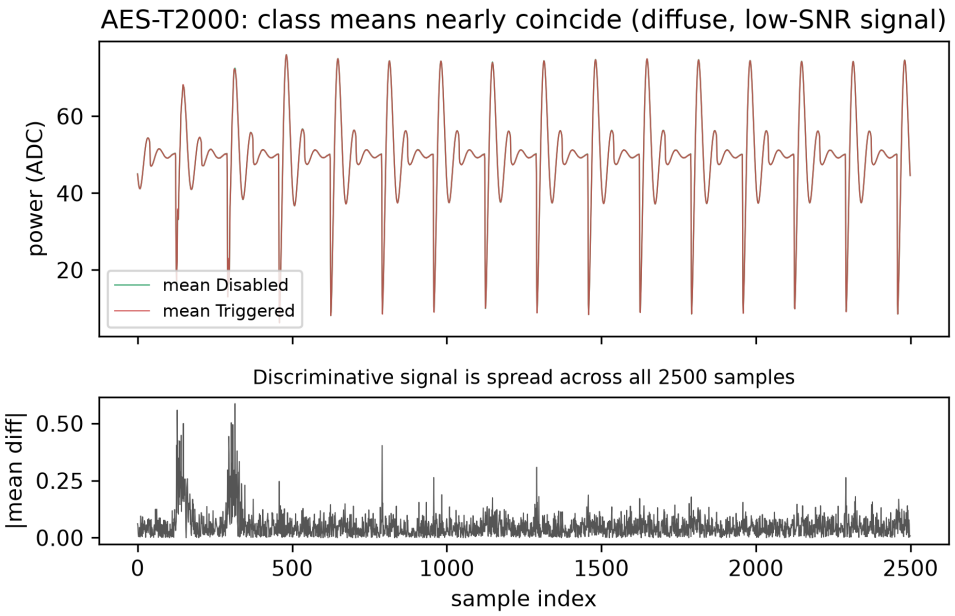


Fig. 4 Mean Disabled vs. mean Triggered trace for AES-T2000 (top; nearly coincident) and per-sample absolute difference (bottom). The discriminative signal is diffuse and low-SNR, spread across all 2500 samples, so global pooling averages it away.

Table 2 Backbone comparison across regimes (ROC-AUC). Capacity does not cross the wall.

Backbone	Within-T2000	Few-shot	Zero-shot
Global-avg-pool CNN/Transformer	∼0.50	—	—
Standardized MLP (no pooling)	∼0.81	0.584	0.398
Hybrid CNN-attention	∼0.81	0.592	0.399
Self-supervised contrastive	—	0.514	0.405
Selective state-space (Mamba)	—	0.502	0.417

signal is diffuse. Global-average-pooling CNN/Transformer backbones score ≈ 0.50 (a mixed-sign pattern averages to zero under pooling); a standardized non-pooling MLP recovers ≈ 0.81 (Table 2).

6.2 Per-Trojan few-shot detection

Table 3 reports the full per-Trojan breakdown under leave-one-Trojan-out with K=20 calibration labels. The mean ROC-AUC of 0.648 conceals a strongly *bimodal* distribution: AES-T700 and AES-T800 are essentially solved (1.000 and 0.989), four Trojans are partial (AES-T500 0.736, AES-T1300 0.667, AES-T1400 0.633, AES-T400 0.622), and the remaining six sit at or near chance (0.50–0.55). The threshold-dependent

Table 3 Per-Trojan few-shot leave-one-Trojan-out detection (K=20; standardized MLP with prototypical head). Values are means over episodes with 95% CIs. Threshold-dependent columns (F_1, TPR@1%) are high-variance across Trojans, as the operating point does not transfer; ROC-AUC and PR-AUC are the reliable summaries.

Trojan	Acc	Prec	Recall	F_1	ROC-AUC	PR-AUC	TPR@1%
AES-T400	0.622	0.572	1.000	0.727	0.622	0.572	1.000
AES-T500	0.700	0.995	0.402	0.562	0.736	0.783	0.611
AES-T600	0.509	0.507	0.525	0.510	0.530	0.573	0.109
AES-T700	1.000	1.000	1.000	1.000	1.000	1.000	1.000
AES-T800	0.911	0.864	1.000	0.923	0.989	0.981	0.966
AES-T1000	0.502	0.500	0.507	0.497	0.507	0.530	0.295
AES-T1100	0.500	0.498	0.512	0.500	0.502	0.551	0.064
AES-T1300	0.629	0.619	0.689	0.646	0.667	0.650	0.086
AES-T1400	0.590	0.586	0.630	0.600	0.633	0.654	0.126
AES-T1600	0.529	0.533	0.528	0.524	0.539	0.579	0.103
AES-T1800	0.536	0.575	0.414	0.453	0.550	0.608	0.119
AES-T2000	0.496	0.498	0.489	0.488	0.500	0.551	0.063
mean	0.627	0.646	0.641	0.619	**0.648**	0.669	0.379

columns expose why aggregate accuracy is misleading: AES-T400 attains recall 1.000 at precision 0.572 (it predicts almost everything positive), whereas AES-T500 attains precision 0.995 at recall 0.402 (it predicts almost nothing positive)—opposite failure modes produced by a decision threshold that does not transfer across Trojans. We therefore treat ROC-AUC and PR-AUC as the reliable summaries and report per-Trojan values throughout.

6.3 Zero-shot cross-family detection collapses below chance

Under LOGO with no target labels, detectability falls from 0.836 to 0.408 (clustering families)—a gap of 0.427, below chance. AES-T700 and AES-T800 invert from within-ROC 1.000 to **0.000** (perfect anti-correlation).

6.4 The collapse is a controlled donor effect

AES-T700 scores **0.971** when held out alone (donor AES-T1400 remains in training) and **0.000** when its family is held out (donor removed). The single controlled variable is the donor's presence. Table 4 quantifies this across all twelve Trojans: full-data leave-one-out transfer (LOO-full) averages only 0.600 against a 0.836 within-benchmark ceiling—a mean gap of 0.236—and falls *below* chance for AES-T400 (0.351) and AES-T1800 (0.423). The best-donor column reveals concentrated structure: AES-T1400 is a hub that transfers strongly to AES-T700, AES-T800, AES-T500, and AES-T600, while AES-T1000↔AES-T1300 form a mutually-transferring pair (Fig. 5). Strong *single* donors coexist with weak *pooled* transfer—e.g. AES-T500 has a 0.85 donor yet a 0.485 pooled LOO-full—because averaging all eleven sources dilutes the one good donor with uninformative and anti-aligned ones.

6.5 Robustness to an independent taxonomy

Repeating the zero-shot test with Trust-HUB payload families: within 0.836 → zero-shot 0.534, gap **0.302** (Table 5). We report 0.302 as the conservative, non-circular

Table 4 Donor structure. *Within*: per-Trojan ceiling. *LOO-full*: full-data zero-shot transfer with all *other* Trojans in training. *Best donor*: the single training Trojan that transfers best, with its transfer ROC-AUC.

Trojan	Within	LOO-full	Gap	Best donor (ROC)
AES-T400	0.704	0.351	0.353	T1600 (0.64)
AES-T500	0.902	0.485	0.417	T1400 (0.85)
AES-T600	0.737	0.589	0.148	T1400 (0.73)
AES-T700	1.000	0.971	0.029	T1400 (1.00)
AES-T800	1.000	0.893	0.107	T1400 (1.00)
AES-T1000	0.824	0.591	0.233	T1300 (0.77)
AES-T1100	0.593	0.487	0.105	T1600 (0.54)
AES-T1300	0.960	0.645	0.315	T1000 (0.91)
AES-T1400	0.806	0.618	0.188	T600 (0.72)
AES-T1600	0.715	0.526	0.189	T1100 (0.58)
AES-T1800	0.949	0.423	0.525	T2000 (0.64)
AES-T2000	0.838	0.621	0.217	T600 (0.65)
mean	0.836	0.600	0.236	—

Table 5 Per-Trojan within-benchmark ceiling vs. zero-shot leave-one-family-out (payload taxonomy). ROC-AUC.

Trojan	Payload	Within	Zero-shot	Gap
AES-T400	RF	0.704	0.366	0.338
AES-T500	DoS	0.902	0.519	0.383
AES-T600	LC	0.737	0.589	0.148
AES-T700	PSC	1.000	0.654	0.346
AES-T800	PSC	1.000	0.482	0.518
AES-T1000	PSC	0.824	0.488	0.336
AES-T1100	PSC	0.593	0.503	0.090
AES-T1300	PSC	0.960	0.540	0.421
AES-T1400	PSC	0.806	0.625	0.181
AES-T1600	RF	0.715	0.538	0.177
AES-T1800	DoS	0.949	0.471	0.477
AES-T2000	PSC	0.838	0.629	0.209
mean		**0.836**	**0.534**	**0.302**

Clustering-defined families give mean zero-shot 0.408 (gap 0.427).

estimate and 0.427 as the upper bound. Fig. 6 shows the per-Trojan breakdown: every Trojan is detectable within its own benchmark, yet most fall to chance when the family is unseen.

6.6 The wall is architecture- and representation-independent

A hybrid CNN-attention backbone matched a linear/MLP baseline at both few-shot (0.592 vs. 0.584) and zero-shot (0.399 vs. 0.398; Table 2). An in-domain self-supervised contrastive encoder did not help (0.405 zero-shot; 0.514 few-shot). A selective state-space (Mamba-style) backbone likewise matched the baseline (0.417

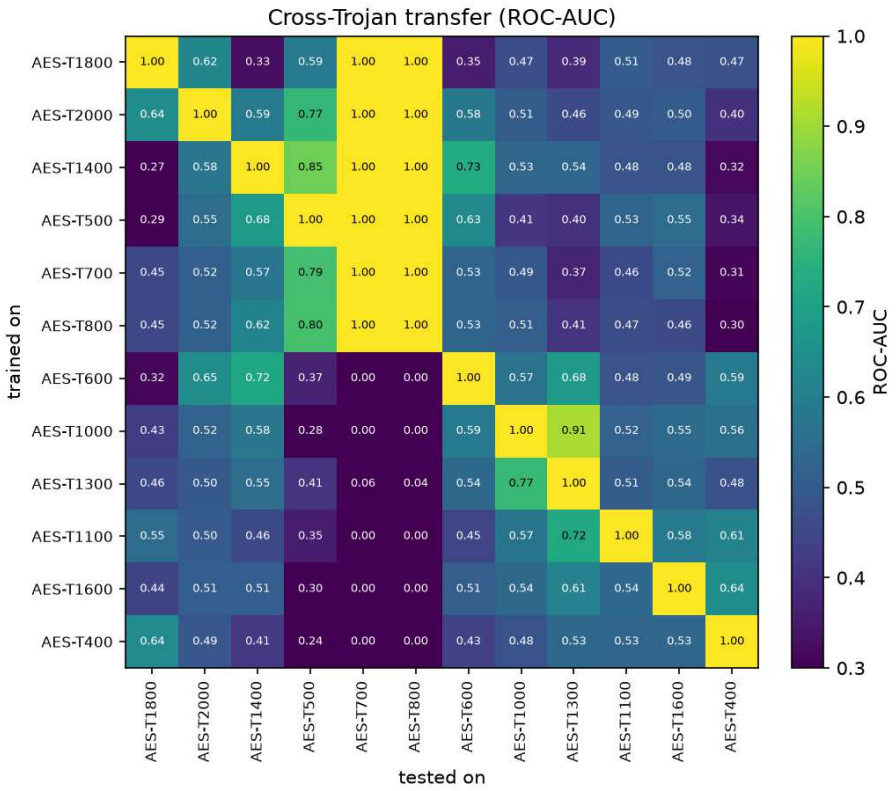


Fig. 5 Cross-Trojan transfer (ROC-AUC), reordered by clustering. Bright blocks are transferable families; off-block pairs fail. Detection transfers only to a seen relative.

zero-shot; 0.502 few-shot). Across nine model families—linear, MLP, convolution-transformer, hybrid CNN-attention, prototypical few-shot networks, donor-selected transfer, self-supervised contrastive pretraining, and selective state-space—zero-shot cross-family detection converges near or below chance (Fig. 7). Cross-family failure is a property of the data, not of the model.

6.7 Functional taxonomy does not govern transferability

Across all 132 directional Trojan pairs, sharing a payload type yields an insignificant transfer difference (Δ=+0.037, Mann–Whitney p=0.13) and trigger type none (Δ=−0.006, p=0.52). The rule transfer $\sim$ same_payload + same_trigger has R^2<0.01 and predicts a successful transfer at AUC≈0.51 (Table 6). The transfer structure that exists is electrical, not functional.

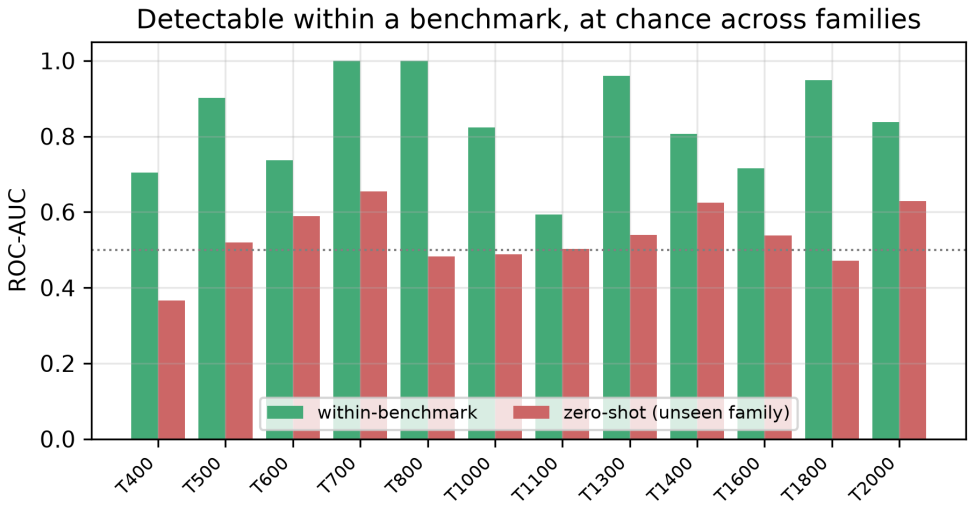


Fig. 6 Per-Trojan within-benchmark ceiling (green) vs. zero-shot leave-one-family-out (red). Every Trojan is detectable within its own benchmark; most fall to chance when the family is unseen, and AES-T700/T800 invert below chance.

Table 6 Transferability rule: does a shared property predict transfer? (132 directional pairs.)

Property	same	diff	gap	p
Payload type	0.530	0.492	+0.037	0.13
Trigger type	0.502	0.508	−0.006	0.52

Rule R^2=0.006; predicting transfer>0.6 at AUC≈0.51.

6.8 A few own-labels beat any transfer

With K target labels, standardized nearest-centroid on the target's own support exceeds pooled transfer at every $K \geq 5$ (K=20: **0.655** vs. 0.605) and rises with K (0.610 → 0.665 for K=1 → 50), while pooled transfer stays flat (∼0.60; Table 7, Fig. 8). Donor-selection does not beat this baseline (0.637; oracle 0.724). The ceiling is Trojan-dependent.

6.9 Few-shot under whole-family hold-out

Table 8 reports few-shot detection when an entire family is held out. Even with K=20 target labels the mean ROC-AUC is only 0.608, and three of four families sit near 0.56. The exception, fam1 (0.757), is buoyed by AES-T700/T800, whose intrinsic within-Trojan signal is strong enough that a handful of labels separate them regardless of the meta-training set. This contrast—strong target labels rescue intrinsically-detectable Trojans but not others—indicates that few-shot success is bounded by each Trojan's

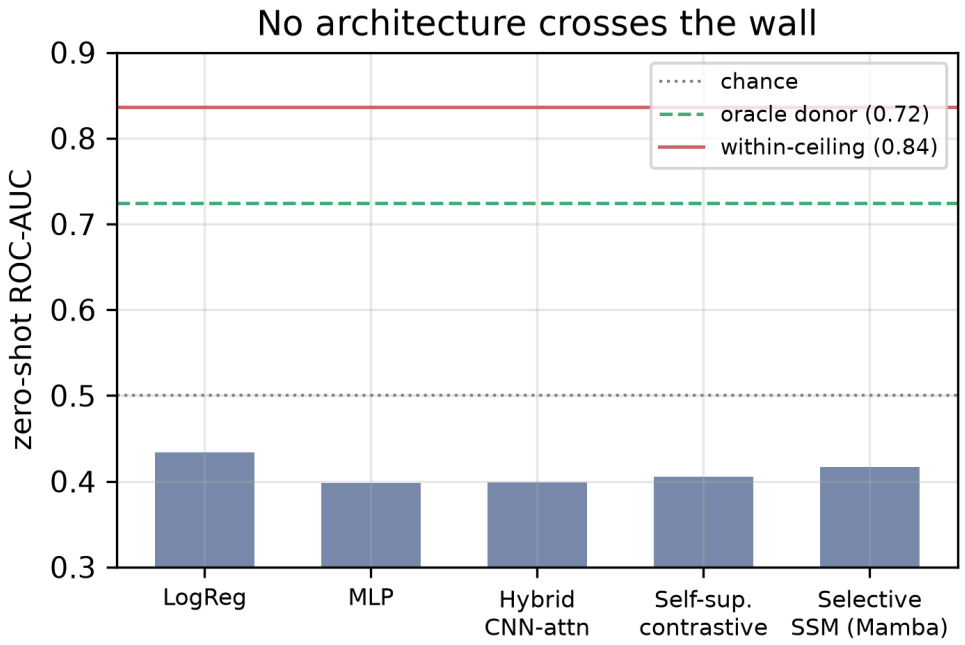


Fig. 7 Zero-shot cross-family ROC-AUC for representative backbones. All cluster near chance, far below the within-benchmark ceiling (0.84) and even the oracle-donor upper bound (0.72): added capacity does not cross the wall.

Table 7 Few-shot method comparison at K=20 (cross-Trojan, ROC-AUC).

Method	ROC-AUC
pool (all source Trojans)	0.605
donor-selected	0.637
proto (target's own K labels)	**0.655**
oracle donor (upper bound)	0.724

proto vs. K: 0.610/0.627/0.642/0.656/0.665 at K=1/5/10/20/50; pooled transfer flat ∼0.60.

own detectability rather than by cross-family transfer, and reinforces the per-chip-calibration conclusion.

7 Discussion

Per-benchmark accuracy on this dataset measures Trojan *familiarity*, not detection: absent a relative of the target in training, side-channel detection is at or below chance, under two independent family definitions, and this cannot be repaired by architecture

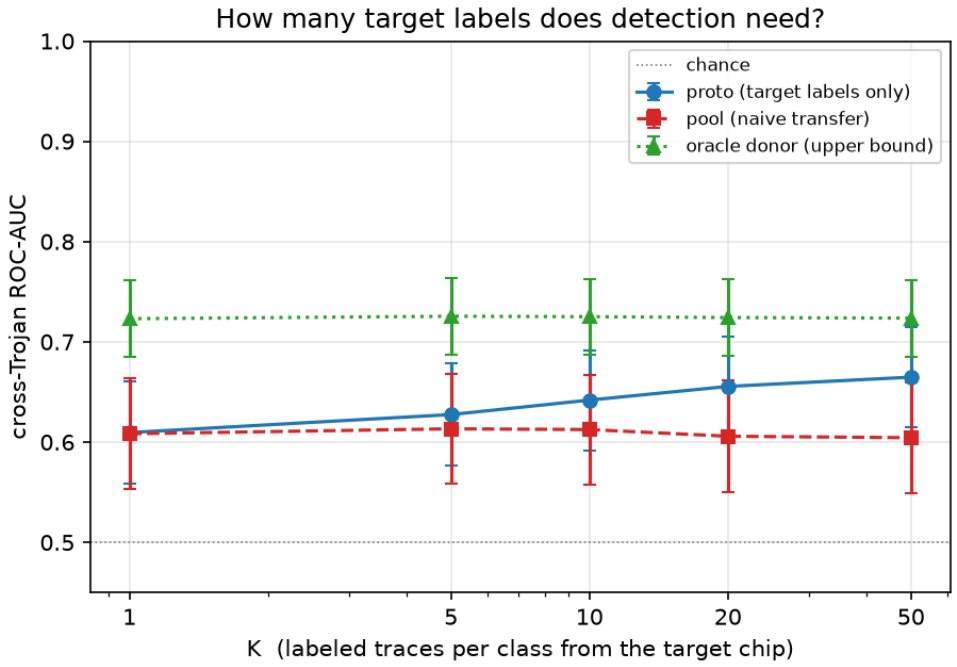


Fig. 8 Cross-Trojan detection vs. number of target calibration labels K. The target's own labels (proto) beat pooled transfer at every $K \geq 5$; pooled transfer is flat.

Table 8 Per-family few-shot leave-one-family-out detection (K=20; clustering-defined families).

Family	Acc	F_1	ROC-AUC	PR-AUC	TPR@1%
fam1	0.748	0.723	0.757	0.770	0.611
fam2	0.537	0.530	0.559	0.607	0.103
fam3	0.533	0.526	0.561	0.592	0.120
fam4	0.550	0.581	0.555	0.561	0.381
mean	0.592	0.590	0.608	0.632	0.304

or predicted by functional taxonomy. An HT side-channel detector therefore cannot ship as a universal pretrained model; it requires per-chip calibration labels, and its ceiling is set by the Trojan's intrinsic detectability. We recommend reporting Trojan-disjoint (and where possible family-disjoint) accuracy alongside any random-split number.

8 Recommendations for Deployment-Faithful Evaluation

Our results imply that the community's default evaluation of ML side-channel hardware-Trojan detectors does not measure the capability it reports. We therefore distill our protocol into a concrete, reusable evaluation standard. We recommend that any study reporting ML-based side-channel HT detection accuracy adopt the following:

1. **Partition at the Trojan level, not the trace level.** Assign all traces of a given Trojan entirely to one split. Report leave-one-Trojan-out (LOTO) accuracy; a pooled trace-level split conflates detection with familiarity and should never be the headline number.
2. **Report a family-disjoint (LOGO) result where families can be defined.** Holding out an entire Trojan family is the strictest and most deployment-realistic setting; report it whenever an independent grouping (e.g. payload or trigger taxonomy) is available.
3. **Report per-Trojan results, not only the mean.** Cross-Trojan detectability is strongly bimodal; a single aggregate can hide that half of the Trojans are at chance. Always publish the per-Trojan breakdown.
4. **Use threshold-free primary metrics.** Report ROC-AUC and PR-AUC as the primary measures. Accuracy, F_1, and TPR at a fixed FPR are informative but must be treated cautiously, because the decision threshold does not transfer across Trojans.
5. **Fit all preprocessing on training Trojans only.** Standardization, normalization, feature selection, and any learned transform must be estimated without the held-out Trojan, or leakage re-enters through the preprocessing.
6. **Separate the zero-shot and few-shot conditions.** Zero-shot (no labels from the device under test) and few-shot (with K per-chip calibration labels) measure different deployment scenarios and must be reported separately, not conflated.
7. **Report a calibration-label budget.** When per-chip calibration is used, report accuracy as a function of the number of labels K, so the deployment cost of the detector is explicit.
8. **Report the within-benchmark ceiling as a reference.** Per-Trojan within-benchmark detectability upper-bounds what any cross-Trojan method can achieve and contextualizes the generalization gap.

Concretely, a reproducible submission should state, per Trojan: the within-benchmark ceiling, the zero-shot LOTO/LOGO ROC-AUC, and—where calibration is used—the few-shot ROC-AUC at the reported label budget. We provide an open-source reference implementation of this protocol (Sec. 5) so that adopting it requires no additional tooling. Reporting a high pooled, trace-level accuracy without these disjoint numbers should be read, in light of our results, as evidence of familiarity rather than of detection capability.

9 Limitations and Future Work

(i) Aggregate means hide bimodality; we report per-Trojan values. (ii) The clustering-family gap (0.427) is partially circular; the taxonomy gap (0.302) is the non-circular estimate. (iii) AES-T2000's payload label is a soft assignment. (iv) The study covers one dataset, one modality (power), one temperature, and one circuit; extending to EM and additional circuits would test whether the wall is universal. (v) Results use a 300-trace subsample; conclusions are stable to subsample and seed. (vi) The electrical driver of transferability (e.g. Trojan area/overhead) is identified but not quantified.

10 Conclusion

Under leakage-free, Trojan-level evaluation, the widely reported 87–100% accuracy of ML side-channel hardware-Trojan detectors does not hold. When the held-out Trojan's family is unseen, detection falls from a 0.84 within-benchmark ceiling to 0.53 ROC-AUC—below chance for several Trojans—and a controlled single-donor knockout confirms the mechanism: one Trojan's score swings from 0.971 to 0.000 depending solely on whether a related Trojan is present in training. The collapse is robust to an independent payload taxonomy (gap 0.30), architecture-independent across nine model families, and unpredictable from functional Trojan categories (R^2<0.01). What prior evaluations measure is Trojan *familiarity*, not detection.

For practitioners, the implication is concrete: a side-channel HT detector cannot ship as a universal pretrained model. Reliable screening requires per-chip calibration labels, and pooling unrelated Trojans actively degrades accuracy. For researchers, we recommend Trojan-disjoint and family-disjoint evaluation as the default, with per-Trojan breakdowns replacing aggregate means. Extending this protocol to electromagnetic traces, additional circuits, and Trojan-overhead correlations are the most direct routes to determining whether the wall is universal and to discovering a positive transferability rule. We release the full protocol and code to support these efforts.

References

[1] Chakraborty, R.S., Wolff, F., Paul, S., Papachristou, C., Bhunia, S.: Mero: A statistical approach for hardware trojan detection. In: International Workshop on Cryptographic Hardware and Embedded Systems, pp. 396–410 (2009). Springer

[2] Jin, Y., Makris, Y.: Hardware trojan detection using path delay fingerprint. In: 2008 IEEE International Workshop on Hardware-oriented Security and Trust, pp. 51–57 (2008). IEEE

[3] Agrawal, D., Baktir, S., Karakoyunlu, D., Rohatgi, P., Sunar, B.: Trojan detection using ic fingerprinting. In: 2007 IEEE Symposium on Security and Privacy (SP'07), pp. 296–310 (2007). IEEE

[4] Narasimhan, S., Du, D., Chakraborty, R.S., Paul, S., Wolff, F.G., Papachristou,

C.A., Roy, K., Bhunia, S.: Hardware trojan detection by multiple-parameter side-channel analysis. IEEE Transactions on computers **62**(11), 2183–2195 (2012)

[5] Narasimhan, S., Du, D., Chakraborty, R.S., Paul, S., Wolff1, F., Papachristou, C., Roy, K., Bhunia, S.: Multiple-parameter side-channel analysis: A non-invasive hardware trojan detection approach. In: 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp. 13–18 (2010). https://doi.org/10.1109/HST.2010.5513122

[6] Du, D., Narasimhan, S., Chakraborty, R.S., Bhunia, S.: Self-referencing: A scalable side-channel approach for hardware trojan detection. In: International Workshop on Cryptographic Hardware and Embedded Systems, pp. 173–187 (2010). Springer

[7] He, J., Zhao, Y., Guo, X., Jin, Y.: Hardware trojan detection through chip-free electromagnetic side-channel statistical analysis. IEEE Transactions on Very Large Scale Integration (VLSI) Systems **25**(10), 2939–2948 (2017)

[8] Salmani, H., Tehranipoor, M., Karri, R.: On design vulnerability analysis and trust benchmarks development. In: 2013 IEEE 31st International Conference on Computer Design (ICCD), pp. 471–474 (2013). IEEE

[9] Shakya, B., He, T., Salmani, H., Forte, D., Bhunia, S., Tehranipoor, M.: Benchmarking of hardware trojans and maliciously affected circuits. Journal of Hardware and Systems Security **1**(1), 85–102 (2017)

[10] Faezi, S., Yasaei, R., Barua, A., Al Faruque, M.A.: Brain-inspired golden chip free hardware trojan detection. IEEE Transactions on Information Forensics and Security **16**, 2697–2708 (2021)

[11] Nasr, A., Mohamed, K., Elshenawy, A., Zaki, M.: A siamese deep learning framework for efficient hardware trojan detection using power side-channel data. Scientific Reports **14**(1), 13013 (2024)

[12] Sarihi, A., Jamieson, P., Patooghy, A., Badawy, A.-H.A.: Multi-criteria hardware trojan detection: A reinforcement learning approach. In: 2023 IEEE 66th International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 1093–1097 (2023). IEEE

[13] Chen, S., Wang, T., Huang, Z., Hou, X.: Detection method of golden chip-free hardware trojan based on the combination of resnext structure and attention mechanism. Computers & Security **134**, 103428 (2023) https://doi.org/10.1016/j.cose.2023.103428

[14] Jiang, Z., Ding, Q.: A framework for hardware trojan detection based on contrastive learning. Scientific Reports **14**(1), 30847 (2024)

[15] Yasaei, R., Yu, S.-Y., Al Faruque, M.A.: Gnn4tj: Graph neural networks for hardware trojan detection at register transfer level. In: 2021 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1504–1509 (2021). IEEE

[16] Bhunia, S., Hsiao, M.S., Banga, M., Narasimhan, S.: Hardware trojan attacks: Threat analysis and countermeasures. Proceedings of the IEEE **102**(8), 1229–1247 (2014)

[17] Xiao, K., Forte, D., Jin, Y., Karri, R., Bhunia, S., Tehranipoor, M.: Hardware trojans: Lessons learned after one decade of research. ACM Transactions on Design Automation of Electronic Systems (TODAES) **22**(1), 1–23 (2016)

[18] Yasaei, R., Faezi, S., Faruque, M.A.A.: Hardware Trojan Power & EM Side-Channel Dataset. https://doi.org/10.21227/9fwb-8978 . https://dx.doi.org/10.21227/9fwb-8978

Declarations

Funding

No funding was received for conducting this study.

Conflicts of interest

F. Amsaad is a Guest Editor of the special collection to which this manuscript is submitted. To ensure impartiality, F. Amsaad was not involved in the editorial handling or peer-review process of this manuscript. All other authors declare that they have no conflict of interest.

Availability of data and material

The Trust-HUB AES-128 power side-channel dataset used in this study is publicly available from IEEE-DataPort at https://ieee-dataport.org/open-access/hardware-trojan-power-em-side-channel-dataset. No new data were generated; all experiments use this existing public dataset.

Code availability

The complete implementation will be made publicly available upon acceptance of this paper.

Authors' contributions

M.R.G. developed the core research concept, designed and implemented the methodology, conducted all experiments, performed the analysis, and wrote the original draft. M.K.P. performed the data processing. S.B. contributed to writing the manuscript.

F.A. and A.R.K. provided advisory supervision and guidance throughout the project. All authors read and approved the final manuscript.

Ethics approval

Not applicable. This study does not involve human participants, human data, or animal subjects.

Consent to participate

Not applicable.

Consent for publication

Not applicable.