HAMMEREVO: LLM-DRIVEN EVOLUTION OF ROWHAMMER PERFORMANCE-ISOLATION DEFENSES

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern DRAM is vulnerable to RowHammer: repeatedly activating a row can flip bits in nearby rows without ever writing them. Safety mitigations stop the flips, but their preventive work—migrations, copies, refreshes—occupies the shared memory-controller service path, and an adversary can trigger that work repeatedly to deny service to benign co-runners. This performance denial-of-service channel is severe yet largely unaddressed: BreakHammer, the existing countermeasure, was evaluated on fixed hammer patterns, not camouflaged ones; real attacks are diverse; and designing a defense demands deep knowledge of each mitigation's mechanism. We present HammerEVO, which applies expert-guided LLM evolution to this problem. The search must cope with a moving adversary, the limited visibility each mitigation exposes, and candidates judgeable only by measurement: domain experts approve hardware extensions and adjudicate measurements, while an LLM proposes attack strategies, probes, and policy code graded by a deterministic, fail-closed simulator campaign. Since evaluating a defense requires real attacks, the campaign evolves both sides. It discovered patterns that degrade benign cores by 13.8–49.1% mean IPC loss (75.6–90.5% worst-core) across multiple mitigation families, which BreakHammer fails to recover, and produced a working performance-isolation policy for each of the eight families: mean benign impact at or below 2.5% (worst-case 7.0%) with near-zero pure-benign overhead, outperforming BreakHammer wherever it engages (worst-core 32–99% reduced to at most 7%) and holding against six published attack families and eight 40-round adaptive red campaigns—for an estimated $50 of API spend and one workstation-week. We release the policies, per-iteration artifacts, and evaluation traces at `https://anonymous.4open.science/r/hammerevo-artifact-EB58`.

1 INTRODUCTION

DRAM stores data as electrical charge in rows of cells. Repeatedly activating one row can disturb neighboring rows and flip their bits—without the attacker ever writing them—a phenomenon known as RowHammer (Kim et al., 2014; Seaborn & Dullien, 2015). Safety mitigations, which we denote D0—spanning activation-tracking, hot-row migration, probabilistic, and refresh-management designs (Saxena et al., 2022; Park et al., 2020; Canpolat et al., 2024b)—detect and defuse aggressor rows before bits flip. But the mitigation's preventive work itself occupies the shared memory-controller service path, and an adversary who triggers it repeatedly denies service to benign co-runners without flipping a single bit (Canpolat et al., 2024a). This performance denial-of-service channel is not a corner case: an unmitigated tenant running a carousel (rotating) hammer pattern degrades benign co-runners by 13.8–49.1% mean IPC loss (75.6–90.5% worst-core) across the baseline D0 defenses we study (§4).

Defending this channel in practice is an open and expertise-bound problem. The prior countermeasure, BreakHammer, observes preventive activity, attributes it to a source, and throttles that source (Canpolat et al., 2024a)—but its published evaluation uses fixed hammer patterns, and in our measurements it either never engages (the attack produces too little attributable activity) or misfires on benign tenants (false throttling up to 99% worst-core). Real attacks are diverse, and each D0 family exposes different observable signals and windows, so a practical performance-isolation defense (D1) must be derived per family, must cooperate with its D0 without weakening it, and must not hurt

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

normal programs. Meeting all three requirements by hand demands scarce hardware knowledge and expert iteration—which is why, years after the channel was named, most families still ship no such defense.

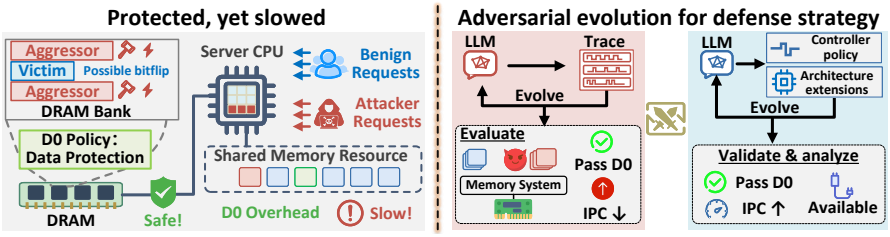

Figure 1: **HammerEVO overview.** *Left:* RowHammer mitigations keep data safe, but their preventive work slows benign co-runners. *Right:* expert-guided evolution of attacks and defenses against this side effect.

At the same time, LLMs have matured into instruments for scientific and algorithmic discovery—AlphaEvolve, FunSearch, and related program-search systems (Novikov et al., 2025; Romera-Paredes et al., 2024; Sharma, 2025; Fu et al., 2026; Lange et al., 2025).

We present HammerEVO, a framework that applies expert-guided LLM evolution to RowHammer performance defense (Figure 1). The goal is defense designs—but evaluating a defense requires real attacks, so the campaign evolves both: attacks that expose each family's window, and defenses that close it. Domain experts set the research direction, approve hardware extensions, and decide from measurements whether a candidate is adopted, revised, or abandoned; the LLM proposes attack skeletons, diagnoses, probes, and policy code; a deterministic, fail-closed grader evaluates every candidate against frozen gates (§3).

The campaign produced three results. First, evolved attacks show the channel is real and severe: benign co-runners lose 13.8–49.1% mean IPC (75.6–90.5% worst-core) across the families—and BreakHammer fails to recover (§4.3). Second, and central, blue-side evolution produced a working D1 for each of the eight mitigation families: mean benign impact at or below 2.5%, worst-case 7.0%, with near-zero pure-benign overhead—better than BreakHammer wherever it engages (worst-core 32–99% reduced to at most 7%), and holding against six published attack families (§4). Third, the defenses survive further scrutiny: eight 40-round adaptive red campaigns found no qualifying breach, and the whole campaign cost an estimated $50 of API spend and one workstation-week.

Our contributions:

- To our knowledge, the **first LLM-driven framework that evolves both the attack and the defense side of RowHammer performance isolation** (§3), where prior work attacks or defends a single mitigation by hand.
- An **attack–defense co-evolution design** (§3): red-side LLM skeleton proposals with genetic instance search under legality and witness constraints; blue-side LLM policy revisions directed by domain experts, with a deterministic, fail-closed simulator gate alone deciding acceptance.
- **Experiments showing both effect and defense** (§4): evolved attacks break all eight representative D0 families and the published BreakHammer policy (13.8–49.1% mean, 75.6–90.5% worst-core benign IPC loss); the evolved D1 policies reduce mean benign impact from up to 49.1% to at most 2.5% (worst-case 7.0%) at near-zero pure-benign overhead, and hold against six state-of-the-art attack families and eight 40-round adaptive red campaigns with no qualifying breach.
- **Release** of the evolved policies, prompts, per-iteration artifacts, and evaluation traces.

2 Background and Related Work

Shared servers and the adversary. Different tenants on a shared server can have isolated address spaces while competing for cache capacity, outstanding-request slots, memory queues, channels, and DRAM service. We assume an adversary co-resident with benign tenants that controls only its own workload (fixture in §4)—a controlled abstraction of a shared server, not a claim that arbitrary cloud rentals guarantee co-location.

From physical disturbance to performance interference. Repeated DRAM row activations can disturb nearby cells, motivating RowHammer safety mitigations (Kim et al., 2014; Seaborn & Dullien, 2015; van der Veen et al., 2016; Frigo et al., 2020; Jattke et al., 2022; Kogler et al., 2022; Meyer et al., 2026). The configured mitigation (D0) may track activations and perform preventive refreshes or row migrations; those actions consume shared resources, so a benign program can lose performance even when data integrity is protected. The performance-isolation layer (D1) observes exposed events and controls admission, budgets, or service priority while preserving D0; BreakHammer is the prior example (Canpolat et al., 2024a). Our attack gate requires at least half of benign cores at half IPC and a 25% mean loss; the defense gate requires mean loss at most 5% and worst-core 10%, plus pure-benign and liveness checks (Appendix A.6–A.7, with a self-contained walkthrough in Appendix A).

Program search and research agents. LLMs synthesize code (Chen et al., 2021), refine outputs from execution feedback (Madaan et al., 2023; Yao et al., 2023), and power program-search systems (Romera-Paredes et al., 2024; Sharma, 2025; Lange et al., 2025; Jiang et al., 2025) and research agents (Boiko et al., 2023; Lu et al., 2024; Yamada et al., 2025; Gottweis et al., 2026). Benchmarks already address tool access, cost, and executable verification (Bragg et al., 2026; Wang et al., 2025): an external executable referee is not our novelty—our contribution is the resulting defense designs and supporting evidence.

Memory-system scope. AQUA, Graphene, TWiCe, Hydra, ReGA, PRAC/RFM, BlockHammer, randomized row swapping, and DAPPER span tracking, in-DRAM action, throttling, and performance-attack-aware designs (Saxena et al., 2022; Park et al., 2020; Lee et al., 2019; Qureshi et al., 2022; Marazzi et al., 2023; Canpolat et al., 2024b; Yağlıkçı et al., 2021; Saileshwar et al., 2022; Woo & Nair, 2025; JEDEC Solid State Technology Association, 2020). HammerEVO does not simulate bit flips or establish a new physical security guarantee; a D0 trigger observed with slowdown is a task qualification rule, not proof that all slowdown came from preventive work.

3 Analyzing the Arms Race: From Problem Features to the Evolution Framework

This section formalizes the problem before presenting the framework: §3.1 fixes terminology and notation; §3.2 defines the system and the performance loss we measure; §3.3 and §3.4 state the attack and defense problems, their objectives, and their constraints; §3.5 derives the evolution mechanism—the search-space factorization, the feedback and selection structure, and the incentive properties that keep the loop honest. The two resulting loops are shown in Figures 2 and 3.

3.1 Terminology and Notation

Table 1 collects the terminology and notation used throughout the paper. Threshold symbols (θ) are pre-registered protocol constants; their instantiated values are reported with the experimental setup (§4.1).

3.2 System Model and the Loss Measure

Let K be a set of safety-mitigation families and M_k the memory system simulator configured with mitigation $k \in K$, whose safety semantics are fixed and trusted throughout. Principals are a set of benign tenants $\mathcal{B}$ running a frozen workload ω, plus one adversarial principal x, whose only capability is choosing its own request stream as a *trace program* τ; we write $\tau = \perp$ when x is absent. A defense policy d is code installed above the mitigation, with d_0 the empty (no-D1) policy. An execution $E(M_k, d, \tau, \omega)$ produces per-core retired instructions I_i for $i \in \mathcal{B}$ over a fixed cycle horizon. All losses are measured against the matched pure-benign reference execution $E(M_k, d_0, \perp, \omega)$, which yields I_i^0:

$$\ell_i \;=\; 1 - \frac{I_i}{I_i^0}, \qquad \bar{L} \;=\; \tfrac{1}{|B|} \textstyle\sum_{i \in B} \ell_i, \qquad L^* \;=\; \max_{i \in B} \ell_i.$$

The side effect we study can now be stated precisely. The loss decomposes as $\bar{L} = \bar{L}_{\text{cont}} + \bar{L}_{\text{prev}}$: ordinary contention the adversary's requests would cause in any memory system, plus the service

Table 1: Terminology and notation.

Symbol	Meaning
K, M_k	safety-mitigation families; simulator configured with family k
$\mathcal{B}$, x, ω	benign tenants; adversarial principal; frozen benign workload
τ, $\perp$	attack trace program; no attack present
d, d_0	defense policy (D1); the empty (no-D1) policy
$E(M_k, d, \tau, \omega)$	an execution, producing per-core retired instructions I_i
I_i^0, ℓ_i, $\bar{L}$, L^*	reference instructions; per-core / mean / worst benign loss
$A_k(t)$, θ_k, W_k	D0 preventive actions; witness threshold; witness predicate
$\mathcal{Z}$, Φ_z, z, ϕ	skeleton space; a skeleton's parameter space; skeleton; its parameters
Legal, Qual_k	legality predicate; red qualification gate
$d = (O, S, \rho, \chi)$, $b_{\max}$	policy signals/state/rules/actuators; state-bit budget
$\mathcal{D}_k$	legal policy space for family k
$O_k^{\exp}$, O_k^{ext}, G	exposed / extension interfaces; expert-approved subset
Accept_k	frozen blue acceptance gate
θ_Q, θ_A, θ_P	pre-registered red, blue attack-arm, and pure-benign thresholds
f_t, π, $\mathcal{H}_t$	measurement vector; expert selection; compressed history
$\mathcal{A}_z$, $\widehat{\tau}$	one skeleton's trace family; a frozen qualified attack
d_k^*	shipped per-family policy (frozen at acceptance)

consumed by the mitigation's own preventive actions $A_k(t)$ (migrations, copies, refreshes) that τ triggers. The channel is dangerous exactly because $\bar{L}_{\mathrm{prev}}$ scales with the *trigger rate* of A_k, not with the adversary's bandwidth: a legally bounded request stream can occupy a disproportionate share of the shared service path. Every attack evaluation therefore carries a *witness predicate*

$$W_k(\tau) \iff \textstyle\sum_t A_k(t) \geq \theta_k,$$

certifying that the measured loss involves the side-effect channel rather than plain contention; a slow run with $\sum_t A_k(t) < \theta_k$ is recorded as *no evidence*, never as zero damage.

3.3 The Attack Problem

The adversary's search space is a set of trace programs with two structurally different levels:

$$\tau = C(z, \phi), \qquad z \in \mathcal{Z}, \quad \phi \in \Phi_z,$$

where $\mathcal{Z}$ is a discrete, compositional space of *strategy skeletons* (pattern structure: hammer geometry, rotation, spreading, phasing) expressed in a restricted grammar, $\Phi_z \subset \mathbb{R}^n$ the continuous parameter space of one skeleton, and C a deterministic compiler to per-core traces. A statically and dynamically checked predicate $\mathrm{Legal}(\tau)$ restricts the adversary to its own request stream—no access to benign state, no influence on D0 internals; violations fail closed. The attack problem against family k is

$$\widehat{\tau} \approx \arg \max_{\tau:\, \mathrm{Legal}(\tau)} \bar{L}(M_k, d_0, \tau, \omega),$$

and a candidate *qualifies* as a challenge for defense research only under the pre-registered predicate

$$\mathrm{Qual}_k(\tau) \iff \mathrm{Legal}(\tau) \wedge W_k(\tau) \wedge \bar{L} \geq \theta_Q^{\mathrm{mean}} \wedge \left|\{i : \ell_i \geq \theta_Q^{\mathrm{slow}}\}\right| \geq \theta_Q^{\mathrm{frac}} |\mathcal{B}|.$$

The full protocol checks this in both the D0-only and the incumbent-D1 deployment of a family ($\kappa \in \{0, 1\}$); the campaign gates on the D0-only arm, and we write the predicate for that arm (Appendix A.6). Two structural facts shape the search. First, each k exposes a different trigger surface (what A_k reacts to, and what the controller can attribute), so the maximizers are family-specific: the useful unit of variation is the skeleton z, not a global intensity knob—concentrated pressure for some families, dispersed or camouflaged pressure for others. Second, $\mathcal{Z}$ is compositional and semantic while Φ_z is numeric, and neither search method covers both; this factorization assigns skeletons to LLM proposal and instances to genetic search in §3.5.

3.4 The Defense Problem

A defense candidate is a policy program $d = (O, S, \rho, \chi)$: observation signals O, internal state S with a bit budget $|S| \leq b_{\max}$, update and judgment rules $\rho : O^* \times S \to S$, and actuators $\chi : S \to \{\text{admit, delay, demote}\}$ subject to rate and timing bounds. Two constraint sets delimit the legal policy space $\mathcal{D}_k$. *Observation constraints:* $O(d)$ must lie in the exposed interface $O_k^{\exp}$, or in an

extension set O_k^{ext} that becomes available only after explicit expert approval $G \subseteq O_k^{\text{ext}}$, i.e. $O(d) \subseteq O_k^{\text{exp}} \cup G$—new hardware capability is a decision, not a search dimension. *Contract constraints:* a static audit $\text{Contract}(d)$ checks declaration–use consistency, state budgets, port budgets, and timing feasibility before d may compile; infeasible candidates fail closed. Given a frozen qualifying attack $\widehat{\tau}$, the defense problem is

$$\min_{d \in \mathcal{D}_k} \ \bar{L}(M_k, d, \widehat{\tau}, \omega)$$

subject to the frozen acceptance gate

$$\begin{aligned}\text{Accept}_k(d) \iff &\text{Build}(d) \wedge \text{Contract}(d) \wedge \bar{L}_{\text{atk}} \le \theta_A^{\text{mean}} \wedge L^*_{\text{atk}} \le \theta_A^{\text{worst}} \\ &\wedge \bar{L}_{\text{pure}} \le \theta_P^{\text{mean}} \wedge L^*_{\text{pure}} \le \theta_P^{\text{worst}} \wedge \text{Live}(d) \wedge \text{Audit}(d),\end{aligned}$$

where the "pure" conjuncts are measured on the no-attack arm $E(M_k, d, \perp, \omega)$; the scenario is chosen before evaluation (Appendix A.7). The gate encodes two requirements that are easy to conflate and must not be: *protection* (the attack conjuncts) and *absence of collateral harm* (the pure-benign conjuncts)—a policy that stops the attack by starving benign tenants fails even at $\bar{L}_{\text{atk}} = 0$.

Why must $\mathcal{D}_k$ be searched at the level of program text rather than parameters? Tuning the thresholds of one fixed policy program explores only that program's parameter family $\{d_\theta : \theta \in \Theta\} \subsetneq \mathcal{D}_k$. The moves that changed outcomes in our search were structural, not parametric: observing a signal the policy did not previously use (enlarging O), replacing instantaneous judgment with a latch (changing the state structure of ρ), or making an intervention revocable. No re-tuning of constants reaches any of these. And because the observation maps O_k^{exp} differ across families, no single program is even feasible—let alone acceptable—for all k: the defense problem is a *family of problems* $\{\min_{d \in \mathcal{D}_k}\}_{k \in K}$, solved per family. When a design does transfer across families, its constants must be re-established against the target family's own measurements rather than inherited.

3.5 The Two Evolution Frameworks

Figures 2 and 3 show the two frameworks that execute the searches defined above. We walk through each in turn; §3.6 then covers the feedback, selection, and incentive structure they share.

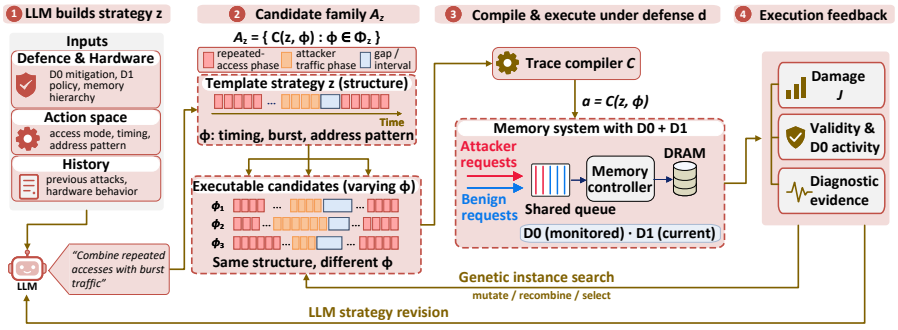

Figure 2: **The red framework: a strategy skeleton becomes a validated attack.** Left: expert-set goal and legal space; middle: LLM skeleton proposal, instance search, and simulation; right: damage, legality and witness feedback driving both search levels.

The red framework (Figure 2). The loop has four stages. *Stage 1 (strategy proposal):* the LLM receives a bounded input pack—the configuration under study (M_k and the incumbent defense d), the legal action space (access modes, timing, address patterns), and a compressed history $\mathcal{H}_t$—and emits a strategy skeleton z, e.g. "combine repeated accesses with burst traffic." *Stage 2 (candidate family):* z is a structural template with free numeric parameters ϕ, defining $\mathcal{A}_z = \{C(z, \phi) : \phi \in \Phi_z\}$; genetic search then mutates, recombines, and selects instances *within* this family—the LLM never chooses individual ϕ. *Stage 3 (compilation and execution):* a deterministic compiler lowers each instance $\tau = C(z, \phi)$ to per-core traces, executed with D0 and the current defense in place. *Stage 4 (feedback):* each execution returns three evidence channels—damage J, legality plus the witness W_k of §3.2, and diagnostics—driving both return edges: the genetic population and the LLM's next skeleton revision when a family saturates. A candidate that clears Qual_k is frozen into the content-addressed attack portfolio as the challenge for blue-side evolution.

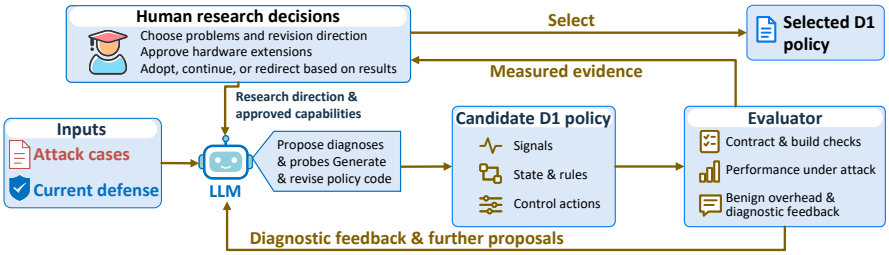

Figure 3: **The blue framework: expert decisions, LLM proposals, external evaluation.** Left: attack cases and the current defense; middle: each candidate declares signals, state and control actions; right: external measurement returns evidence to the experts, who select the direction and approve hardware extensions.

The blue framework (Figure 3). The loop has five stations. *Inputs:* the frozen attack cases and the current defense version. *LLM proposal:* given the experts' current direction and approved capabilities, the LLM proposes diagnoses of the latest measurements, diagnostic probes, and new or revised policy code. *Candidate card:* every candidate is a typed artifact $d = (O, S, \rho, \chi)$ (§3.4), so what the policy observes and how it may act are explicit before any simulation. *Evaluation:* the evaluator runs the contract and build checks ($\mathrm{Contract}(d)$, $\mathrm{Build}(d)$), then measures performance under attack and benign overhead, producing the measurement vector f_t (§3.6). *Expert research decisions:* the measured evidence returns to the domain experts, who choose the direction, approve or reject hardware extensions (G), and decide—from measurements, never from the LLM's self-report—whether to adopt, continue, or redirect; a second, smaller edge returns diagnostics to the LLM for automated repair. The output of the loop is a *selected* D1 policy d_k^*, frozen as an immutable snapshot at acceptance, with its complete version history and evidence trail—never an automatically promoted one.

3.6 Feedback, Selection, and Incentive Properties

The two frameworks share one evaluation substrate and one feedback contract. On the red side, the LLM acts as a proposal operator over skeletons, $z_{t+1} \sim P_{\mathrm{LLM}}(\cdot \mid \mathcal{H}_t)$; on the blue side, it proposes the next policy as a function of the current artifact and its measurements, $d_{t+1} \sim P_{\mathrm{LLM}}(\cdot \mid d_t, f_t, \mathcal{H}_t)$.

Feedback and selection: a vector, not a scalar fitness. Standard evolutionary program search attaches a scalar fitness value to each candidate and lets that value alone select the next generation; we deliberately deviate on the blue side. Each evaluation returns a measurement vector $f_t = (\bar{L}_{\mathrm{atk}}, L^*_{\mathrm{atk}}, \bar{L}_{\mathrm{pure}}, L^*_{\mathrm{pure}}, \sum_t A_k,$ witness and audit diagnostics$)$, and no scalarization of it drives the loop. Two distinct consumers read it. The *development loop* uses f_t as information: the LLM proposes the next modification, and the domain experts choose the direction—a selection $\pi(f_t, \mathcal{H}_t) \in \{\text{adopt, continue, redirect, stop}\}$ that also gates the hardware-approval set G. The *acceptance gate* Accept_k is frozen and external: it alone decides validity, independent of the LLM's or the experts' judgment. Trajectories may be non-monotone—a version that adds an observation may regress before the control logic catches up—and only artifacts that pass the gate are recorded as results.

Incentive properties. Four properties keep the loop honest. (i) *External scoring:* the submission cannot set its own score; missing or invalid evidence defaults to rejection (fail-closed). (ii) *Replay verifiability:* every accepted artifact replays bit-exactly under a build whose simulator hash is embedded in the verdict, so "proposed" and "accepted" are different events; this voided five apparent successes that concurrent builds had manufactured. (iii) *Asymmetric certificates:* red qualifies by causing witnessed damage (Qual_k), blue is accepted only by preventing it without collateral (Accept_k). (iv) *Frozen artifacts:* each accepted candidate is frozen at its acceptance and never quietly revised; attacks enter a content-addressed portfolio so later defenses face current and retained opponents.

4 EXPERIMENTS

4.1 SETUP

We study eight representative safety-mitigation families spanning tracking-, migration-, probability-, and refresh-based designs (AQUA, Graphene, Hydra, PARA, PRAC, ReGA, RFM, TWiCe (Saxena et al., 2022; Park et al., 2020; Qureshi et al., 2022; Lee et al., 2019; Marazzi et al., 2023; Canpolat et al., 2024b; JEDEC Solid State Technology Association, 2020; Kim et al., 2014)). All main results are measured in the Ramulator-2.0-based cycle-accurate DRAM simulator released with BreakHammer (Luo et al., 2024; Canpolat et al., 2024a), configured with DDR5-VRR timing and that work's host and memory geometry, so our measurements are directly comparable to the baseline. Benign workloads are SPEC CPU2017 (Standard Performance Evaluation Corporation, 2017) and YCSB mixes (Cooper et al., 2010) (plus synthetic streams); the frozen *carousel* acceptance attack, mixes, seeds, and horizons are in the archived configurations. The red and blue loops run GLM-5.3-family models (Zeng et al., 2024) (other deployments in Appendix D). "Mean"/"worst" are the equal-weight mean and maximum per-core benign IPC loss relative to the pure-benign D0-only reference arm; the gates of §3 are instantiated as $\theta_Q^{\text{mean}} = 25\%$, $\theta_Q^{\text{slow}} = \theta_Q^{\text{frac}} = \frac{1}{2}$ (red) and $\theta_A^{\text{mean}} = 5\%$, $\theta_A^{\text{worst}} = 10\%$, $\theta_P^{\text{mean}} = 1\%$, $\theta_P^{\text{worst}} = 3\%$ (blue). Two checks sit outside this predicate: the AQUA development contract also caps the 95th-percentile pure-benign overhead at 2%, and the grouped benign-pressure stress cases relax the pure-benign bounds to 3% mean and 5% worst (Appendix A.7). Per-family policy mechanisms are explained in Appendix B; per-cell values, configurations, and replay instructions accompany this draft.

4.2 RED-SIDE DISCOVERY: A WORKLOAD-AGNOSTIC ATTACK

The archived single-channel discovery campaign illustrates the red loop of §3.5 end to end; the eight-channel adaptive campaigns against the final defenses appear in §4.5.

The red LLM's first outer round proposed five legal mechanism templates; template-local genetic search evaluated 86 instances against a frozen baseline (17, 18, 17, 16, 18 per template; Table 2), with the numerical parameters of inner mutations not chosen by the LLM. The best retained internal objective rose across the four generations (gen 0–3) from 0.786 (T1) to 1.757 (T2, the selected template), the gain occurring across templates rather than within any single LLM hypothesis; T4 (1.310) was the runner-up, and T3/T5 plateaued at 0.872/0.640.

Table 2: **The five legal attack-mechanism templates of the first outer round.** "Objective" is the best retained internal search coordinate per template—neither benign IPC loss nor a pass label.

Template	Mechanism hypothesis	Instances	Best objective
T1	rotating pressure	17	0.786
T2	**dual-rate bank sweep (selected)**	18	**1.757**
T3	occupancy shrew	17	0.872
T4	single-core max pressure	16	1.310
T5	collective sync hammer	18	0.640

The selected candidate combined rotating pressure and RowHammer-like request phases—the skeleton §3.3 predicts: dual-rate hammering to keep the D0 firing, rotation to defeat per-core attribution. Validated across all eight registered D0 configurations, it drove AQUA's D0-only reference to **54.2% mean and 76.8% worst-core benign IPC loss** (13 of 16 benign cores at half speed or less, clearing the pre-registered qualification gate); across the eight configurations the mean loss fell in a 53–54% band (75–77% worst-core), with per-family endpoints archived in the data package. Its harm spectrum is workload-agnostic: the same trace degrades every benign application mix we deployed.

4.3 MAIN RESULT: EIGHT FAMILIES, THREE DEFENSES, TWO WORLDS

Table 3 is the campaign in one view. Without a D1, the acceptance attack costs benign tenants 13.8–49.1% mean and 75.6–90.5% worst-core IPC across the eight families; with the evolved D1, every family sits at or below 2.5% mean and 7.0% worst. With no attacker present, the evolved policies cost nothing measurable, while BreakHammer's throttler misfires on benign phases—worst-

Table 3: **Master results matrix: 8 D0 families × 6 conditions.** Cells are mean/worst-core benign IPC loss (%) over two worlds (pure-benign vs. frozen carousel attack) × three defense states (no D1; BreakHammer; evolved D1 at its shipped version: AQUA v6 (FASTWIRE), Graphene v2, Hydra v8, PARA v32, PRAC v6, ReGA v2, RFM v12, TWiCe v7; Appendix B). Pure-benign + no-D1 is the reference (zero by construction). ReGA and TWiCe share identical no-D1 cells: TWiCe's D0 never triggers under the carousel attack, so both measure the same benign contention (confirmed by re-run).

		Pure benign (no attack)			Under acceptance attack		
D0	D1 ver.	no D1	BreakHammer	evolved D1	no D1	BreakHammer	evolved D1
AQUA	v6	0.0/0.0	0.0/0.0	0.0/0.0	46.3/88.9	46.3/88.9	−0.1/0.5
Graphene	v2	0.0/0.0	0.0/0.0	0.0/0.0	20.0/78.1	20.0/78.1	0.9/2.1
Hydra	v8	0.0/0.0	0.0/0.0	0.0/0.0	20.3/78.4	20.3/78.4	0.5/2.9
PARA	v32	0.0/0.0	2.0/74.7	0.0/0.0	18.8/77.6	9.5/75.5	1.1/5.1
PRAC	v6	0.0/0.0	0.0/0.0	0.0/0.0	13.8/75.6	13.8/75.6	0.5/2.7
ReGA	v2	0.0/0.0	47.6/99.1	0.0/0.0	49.1/90.5	30.6/99.1	1.6/4.0
RFM	v12	0.0/0.0	−0.4/28.3	0.0/0.0	19.7/77.9	6.8/32.1	2.5/7.0
TWiCe	v7	0.0/0.0	0.0/0.0	0.0/0.0	49.1/90.5	49.1/90.5	1.2/3.6

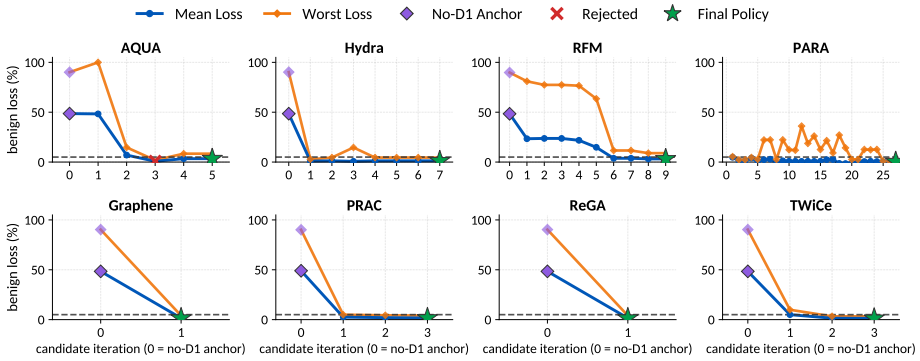

Figure 4: **Defense evolution across the eight families.** Mean (slate) and worst-core (orange) benign IPC loss per candidate; the horizontal axis counts evaluated candidates chronologically (0 = no-D1 anchor). Diamond: no-D1 damage; cross: rejected; star: shipped; dashed: 5% mean bound. Anchors use the evolution campaign's acceptance attack and are not comparable to Table 3's carousel cells; PARA is shown on the literature battery. TWiCe's third candidate was rejected for a $+41.6\%$ pure-benign regression (axis not shown).

core false throttles of $+28.3\%$ (RFM), $+74.7\%$ (PARA), and $+99.1\%$ on ReGA, whose constant preventive activity keeps its source attribution firing with zero attacker present. Three observations.

First, the comparison against BreakHammer clarifies *why* per-family derivation matters. Under the carousel acceptance attack it never engages on five families—too little attributable preventive activity reaches its source-attribution threshold (PRAC's attribution wiring is vestigial; TWiCe takes no preventive action at 8 channels). Where it does engage (RFM, PARA, ReGA), it leaves 32–99% worst-core loss under attack (mean 6.8–30.6%)—on ReGA it nearly halts a benign core even in peace time. The evolved defenses match or beat BreakHammer wherever the comparison is measurable.

Second, the defenses are *semantically distinct* across families—an admission tripwire (AQUA), revocable probation over a row-set sketch (PARA), a relative-miss latch (RFM)—even though the submission skeleton is shared. Knowledge transferred where valid: the Graphene, PRAC, and ReGA policies share the occupancy-latch skeleton first derived for Hydra, re-calibrated against each family's own census.

Third, acceptance is not the end of validation: §4.4 traces each policy's evolution, §4.5 sweeps the final policies against six literature attack families, and §4.5 challenges them with adaptive red search.

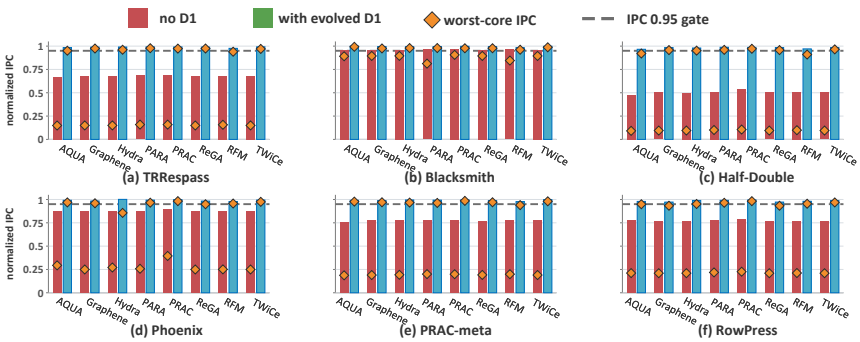

Figure 5: **Literature-attack battery: six attack families × eight D0 families, with and without the evolved D1.** Bars: normalized benign IPC (1.0 = no loss), no D1 (red) vs. evolved D1 (green); diamonds: worst core; dashed: 5% mean bound. The marked Hydra×Phoenix cell is the single exception on seed 123 (§4.5).

4.4 Defense Evolution Across the Eight Families

Figure 4 shows the eight trajectories; they are research processes, not monotone improvements, and three recurring lessons explain the shapes (Appendix B). *Observation precedes action* (Hydra): every pre-observation candidate failed the gate; only an expert-approved cross-controller occupancy witness let the chain converge, and the skeleton then transferred to Graphene, PRAC, and ReGA with per-family recalibration. *References contaminate* (RFM): early candidates misjudged benign bursts under a pressure-polluted baseline; anchoring the reference to benign-only measurements (v12) removed the channel. *Intervention must be cheap to be wrong about* (PARA): v14's permanent probation reached 4/6 defended families, then regressed from v15 to v25, bottoming at 0/6; revocable probation (v30) and pressure gating (v31–v32) recovered to 6/6.

4.5 Validation: Literature Battery and Adaptive Red Campaigns

Each family's final policy was swept against traces adapted from six published attack families (TRRespass, Blacksmith, Half-Double, Phoenix, PRAC-meta (the PRAC availability / DAPPER row of Table 4), RowPress; Appendix C)—a development and regression set, not a held-out test. Figure 5 gives per-cell outcomes. Under a uniform gate (mean $\leq 5\%$, worst $\leq 10\%$), **47 of 48 family–attack cells are defended**; the exception is Hydra×Phoenix (worst-core 14.4% on seed 123, shipped v8), adjudicated defended under the pre-declared multi-seed protocol (frozen single-seed verdict 47/48, and 46/48 under each line's own gate; adjudicated 48/48). We then challenged each final policy with a 40-round adaptive red campaign per family.

The adaptive red campaigns found no qualifying breach: the best measured mean loss is 19.0% (AQUA), and all other families are at 13.9% or below. Caveats: "no qualifying win" is not "no residual pressure" (the best AQUA and Hydra attacks still inflict 55.4%/55.2% worst-core loss), and the Graphene and TWiCe lines produced no witnessed rounds—failed rounds are missing measurements, never zero damage.

Scope of the method evidence. On three pre-registered tasks (a paired pilot, two arms per task, six sessions), a generic adapter and HammerEVO each passed numerically in two of three sessions—the PARA passes do not survive the underlying checker and the RFM arms are not paired—so we claim no advantage over a simpler organization of the search (distinct from the pre-registered ablation protocol of Appendix L). Replay arbitration voided five phantom successes, and the AQUA held-out seed sweep bounds what the gate alone certifies (Appendix L).

5 Conclusion

We presented HammerEVO, an expert-guided LLM evolution campaign for the RowHammer side-effect arms race: researchers set the direction and judged the evidence, an LLM proposed attacks, probes, and policy code, and a fail-closed simulator grader decided what survived—yielding a validated performance-isolation defense for each of eight mitigation families.

Statement on the Use of Artificial Intelligence

The core code of the evaluation framework and the main text of this paper were written by the authors. Language models were used in three supporting roles: (i) to complete parts of the non-core code (glue scripts and utilities); (ii) as the research instrument of the reported experiments, where LLMs generated attack-strategy proposals and defense-policy candidates that were then graded by the deterministic simulator; and (iii) for language polishing during manuscript preparation. Domain experts supplied the domain knowledge, designed and maintained the arena, reviewed all hardware-extension proposals, selected the protocols, and take full responsibility for the claims and artifacts. The archived discovery used a tool-enabled research session. The current evidence audit identified reporting and plotting errors, which are disclosed with their corrections in Appendix H.

Reproducibility Statement

We retain candidate code, probe outputs, validation records, and per-run ledgers. Figure data and extraction scripts accompany this draft. Appendix H distinguishes historical records, invalid measurements, and outstanding revalidation. The supplementary artifact—the eight evolved D1 policies, the frozen attack programs, all result data, the simulator and evaluation harness with rebuild instructions, and an immutable file manifest—is available at `https://anonymous.4open.science/r/hammerevo-artifact-EB58`.

References

Daniil A. Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. In *Nature*, volume 624, pp. 570–578, 2023.

Jonathan Bragg, Mike D'Arcy, Nishant Balepur, et al. AstaBench: Rigorous benchmarking of AI agents with a scientific research suite. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2510.21652.

Oğuzhan Canpolat, A. Giray Yağlıkçı, Ataberk Olgun, Ismail Emir Yuksel, Yahya Can Tuğrul, Konstantinos Kanellopoulos, Oğuz Ergin, and Onur Mutlu. Breakhammer: Enhancing rowhammer mitigations by carefully throttling suspect threads. In *Proceedings of the 2024 57th IEEE/ACM International Symposium on Microarchitecture*, MICRO '24, pp. 915–934. IEEE Press, 2024a. doi: 10.1109/MICRO61859.2024.00072. URL `https://doi.org/10.1109/MICRO61859.2024.00072`.

Oğuzhan Canpolat, A. Giray Yağlıkçı, Geraldo F. Oliveira, Ataberk Olgun, Oğuz Ergin, and Onur Mutlu. Understanding the security benefits and overheads of emerging industry solutions to DRAM read disturbance. *arXiv preprint arXiv:2406.19094*, 2024b.

Mark Chen, Jerry Tworek, Heewoo Jun, et al. Evaluating large language models trained on code. In *arXiv preprint arXiv:2107.03374*, 2021.

Brian F. Cooper, Adam Silberstein, Erwin Tam, Raghu Ramakrishnan, and Russell Sears. Benchmarking cloud serving systems with YCSB. In *Proceedings of the 1st ACM Symposium on Cloud Computing (SoCC)*, pp. 143–154, 2010.

Finn de Ridder, Pietro Frigo, Emanuele Vannacci, Herbert Bos, Cristiano Giuffrida, and Kaveh Razavi. SMASH: Synchronized many-sided rowhammer attacks from javascript. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 1001–1018, 2021.

Pietro Frigo, Emanuele Vannacci, Hasan Hassan, Victor van der Veen, Onur Mutlu, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi. Trrespass: Exploiting the many sides of target row refresh, 2020. URL `https://arxiv.org/abs/2004.01807`.

Rongliang Fu, Yi Liu, Qiang Xu, and Tsung-Yi Ho. Mappingevolve: Llm-driven code evolution for technology mapping. In *ACM/IEEE Design Automation Conference (DAC)*, 2026.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Petar Sirkovic, Artiom Myaskovsky, Grzegorz Glowaty, Felix Weissenberger, Alessio Orlandi, Dan Popovici, Anil Palepu, Keran Rong, Ryutaro Tanno, Khaled Saab, Fan Zhang, Jacob Blum, Andrew Carroll, Kavita Kulkarni, Nenad Tomašev, Dina Zverinski, Ivor Rendulic, Elahe Vedadi, Florian Hasler, Luka Rimanic, Marina Boia, Ivan Budiselic, Ben Feinstein, Mathias Bellaiche, Tom Sheffer, Jan Freyberg, Jeremy Ratcliff, Ottavia Bertolli, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Yuan Guan, Vikram Dhillon, Eeshit Dhaval Vaishnav, Byron Lee, Tiago R. D. Costa, José R. Penadés, Gary Peltz, Yossi Matias, James Manyika, Demis Hassabis, Yunhan Xu, Pushmeet Kohli, Annalisa Pawlosky, Alan Karthikesalingam, and Vivek Natarajan. Accelerating scientific discovery with co-scientist. *Nature*, 655(8122):487–496, May 2026. ISSN 1476-4687. doi: 10.1038/s41586-026-10644-y. URL http://dx.doi.org/10.1038/s41586-026-10644-y.

Patrick Jattke, Victor van der Veen, Pietro Frigo, Stijn Gunter, and Kaveh Razavi. BLACKSMITH: Scalable rowhammering in the frequency domain. In *IEEE Symposium on Security and Privacy (S&P)*, pp. 716–734, 2022.

JEDEC Solid State Technology Association. *DDR5 SDRAM Standard*. JEDEC, 2020.

JEDEC Solid State Technology Association. *DDR5 SDRAM Standard*. JEDEC, 2024.

Zhengyao Jiang, Dominik Schmidt, Dhruv Srikanth, Dixing Xu, Ian Kaplan, Deniss Jacenko, and Yuxiang Wu. Aide: Ai-driven exploration in the space of code. *arXiv preprint arXiv:2502.13138*, 2025.

Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji-Hye Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In *International Symposium on Computer Architecture (ISCA)*, 2014.

Andreas Kogler, Jonas Juffinger, Salman Qazi, Yoongu Kim, Moritz Lipp, Nicolas Boichat, Eric Shiu, Mattias Nissler, and Daniel Gruss. Half-Double: Hammering from the next row over. In *31st USENIX Security Symposium (USENIX Security 22)*, pp. 3807–3824, 2022.

Robert Tjarko Lange, Yuki Imajuku, and Edoardo Cetin. Shinkaevolve: Towards open-ended and sample-efficient program evolution, 2025. URL https://arxiv.org/abs/2509.19349.

Eojin Lee, Ingab Kang, Sukhan Lee, G. Edward Suh, and Jung Ho Ahn. TWiCe: Preventing rowhammering by exploiting time window counters. In *International Symposium on Computer Architecture (ISCA)*, pp. 385–396, 2019.

Chris Lu, Cong Lu, Robert T. Lange, et al. The ai scientist: Towards fully automated open-ended scientific discovery. In *arXiv preprint arXiv:2408.06292*, 2024.

Haocong Luo, Ataberk Olgun, A. Giray Yağlıkçı, Yahya Can Tuğrul, Steve Rhyner, Meryem Banu Cavlak, Joël Lindegger, Mohammad Sadrosadati, and Onur Mutlu. RowPress: Amplifying read disturbance in modern DRAM chips. In *International Symposium on Computer Architecture (ISCA)*, 2023. doi: 10.1145/3579371.3589063.

Haocong Luo, Yahya Can Tuğrul, F. Nisa Bostancı, Ataberk Olgun, A. Giray Yağlıkçı, and Onur Mutlu. Ramulator 2.0: A modern, modular, and extensible DRAM simulator. *IEEE Computer Architecture Letters*, 2024.

Aman Madaan, Niket Tandon, Prakhar Gupta, et al. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Michele Marazzi, Flavien Solt, Patrick Jattke, Takashi Kubo, and Kaveh Razavi. REGA: Scalable rowhammer mitigation with refresh-generating activations. In *IEEE Symposium on Security and Privacy (S&P)*, 2023.

Diego Meyer, Patrick Jattke, Michele Marazzi, Salman Qazi, Daniel Moghimi, and Kaveh Razavi. Phoenix: Rowhammer attacks on ddr5 with self-correcting synchronization. In *IEEE Symposium on Security and Privacy (S&P)*, 2026. Distinguished Paper Award; ETH Zurich ComSec and Google.

Jayadev Misra and David Gries. Finding repeated elements. *Science of Computer Programming*, 2 (2):143–152, 1982.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. Alphaevolve: A coding agent for scientific and algorithmic discovery, 2025. URL https://arxiv.org/abs/2506.13131.

Yeonhong Park, Woosuk Kwon, Jung Ho Ahn, Eojin Lee, Jae W. Lee, and Tae Jun Ham. Graphene: Strong yet lightweight row hammer protection. In *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2020.

Moinuddin K. Qureshi, Aditya Rohan, Gururaj Saileshwar, and Prashant J. Nair. Hydra: Enabling low-overhead mitigation of row-hammer at ultra-low thresholds via hybrid tracking. In *International Symposium on Computer Architecture (ISCA)*, pp. 699–710, 2022.

Kaveh Razavi, Ben Gras, Erik Bosman, Bart Preneel, Cristiano Giuffrida, and Herbert Bos. Flip feng shui: Hammering a needle in the software stack. In *25th USENIX Security Symposium (USENIX Security 16)*, 2016. URL https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/razavi.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, et al. Mathematical discoveries from program search with large language models. *Nature*, 625:468–475, 2024. doi: 10.1038/s41586-023-06924-6.

Gururaj Saileshwar, Bolin Wang, Moinuddin K. Qureshi, and Prashant J. Nair. Randomized row-swap: Mitigating row hammer by breaking spatial correlation between aggressor and victim rows. In *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pp. 1056–1069, 2022.

Anish Saxena, Gururaj Saileshwar, Prashant J. Nair, and Moinuddin K. Qureshi. Aqua: Scalable rowhammer mitigation by quarantining aggressor rows at runtime. In *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2022.

Mark Seaborn and Thomas Dullien. Exploiting the dram rowhammer bug to gain kernel privileges. In *Black Hat USA*, 2015.

Asankhaya Sharma. Openevolve: An open-source implementation of alphaevolve. https://github.com/algorithmicsuperintelligence/openevolve, 2025.

Standard Performance Evaluation Corporation. SPEC CPU2017 Benchmarks. https://www.spec.org/cpu2017/, 2017.

Victor van der Veen, Yanick Fratantonio, Martina Lindorfer, Daniel Gruss, Clémentine Maurice, Giovanni Vigna, Herbert Bos, Kaveh Razavi, and Cristiano Giuffrida. Drammer: Deterministic rowhammer attacks on mobile platforms. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 1675–1689, 2016.

Zhun Wang, Tianneng Shi, Jingxuan He, Matthew Cai, Jialin Zhang, and Dawn Song. CyberGym: Evaluating AI agents' real-world cybersecurity capabilities at scale. *arXiv preprint arXiv:2506.02548*, 2025.

Jeonghyun Woo and Prashant J. Nair. DAPPER: A performance-attack-resilient tracker for rowhammer defense. In *International Symposium on High-Performance Computer Architecture (HPCA)*, 2025.

Jeonghyun Woo, Shaopeng Lin, Prashant J. Nair, Aamer Jaleel, and Gururaj Saileshwar. QPRAC: Towards secure and practical PRAC-based rowhammer mitigation using priority queues. In *International Symposium on Computer Architecture (ISCA)*, 2025.

Yutaro Yamada, Robert T. Lange, Cong Lu, et al. The ai scientist-v2: Workshop-level automated scientific discovery via agentic tree search. *arXiv preprint arXiv:2504.08066*, 2025. Sakana AI.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

A. Giray Yağlıkçı, Minesh Patel, Jeremie S. Kim, Roknoddin Azizibarzoki, Ataberk Olgun, Lois Orosa, Hasan Hassan, Jisung Park, Konstantinos Kanellopoulos, Taha Shahroodi, Saugata Ghose, and Onur Mutlu. BlockHammer: Preventing rowhammer at low cost by blacklisting rapidly-accessed DRAM rows. In *International Symposium on High-Performance Computer Architecture (HPCA)*, pp. 345–358, 2021.

Aohan Zeng et al. ChatGLM: A family of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

A Background and Experimental Walkthrough

This appendix introduces the memory-system concepts and experimental conventions needed to interpret HammerEVO. It separates the physical RowHammer phenomenon, the deployed system motivating our task, and the controlled simulator experiments actually performed. Readers familiar with DRAM can proceed to the attack and defense gates in Sections A.6 and A.7.

A.1 Shared Servers, Tenants, and Protection Domains

What a tenant controls. A tenant rents computing capacity and runs its own application within a protection domain, for example a virtual machine. The provider's trusted software manages placement, memory mappings, and resource accounting. In a shared deployment, tenants can execute on different cores but still use common cache or memory resources. Address-space isolation prevents one tenant from directly accessing another tenant's private memory; it does not automatically provide performance isolation for a shared queue, channel, or bank.

Our adversary controls the memory-request behavior of one co-resident tenant. It may coordinate its own worker cores and change its own access patterns. It may not modify the hypervisor, the benign programs, D0, the validation rules, or another tenant's identity. We assume relevant resource sharing; we do not implement a cloud-placement attack that guarantees co-location with a chosen customer. Historical cross-VM RowHammer work illustrates why physical sharing is security-relevant (Razavi et al., 2016), but its bit-flip exploitation conditions are not the performance threat model evaluated here.

Tenant, principal, and core are different objects. A principal is an opaque protection-domain identifier carried or derived from trusted request metadata. It may abstract a tenant, VM, or another resource-accounting domain. Several cores can belong to one principal, and several benign principals can coexist with an attacker. The policy may use an exposed, trusted core-to-principal mapping, but it cannot interpret a particular numerical ID as inherently malicious. Identity permutation and multi-core benign domains are therefore relevant tests.

The formal fixture has 32 modeled cores, with one attacker tenant using 1–16 cores. In the archived AQUA example, 16 attacker cores share one principal and 16 cores execute benign traces. The g4 and g8 variants group benign cores into larger principals; they are not four-core or eight-core machines. Replacing the single attacker with many separately identified malicious tenants changes the threat model and belongs to a separately labelled stress test. Likewise, a trace-language implementation limit is not a substitute for the smaller formal core-allocation constraint.

Real vCPUs can be scheduled or migrated and need not correspond to dedicated physical cores. HammerEVO fixes its simulated geometry and principal map for a case rather than modeling every cloud-management operation.

A.2 DRAM Requests and the RowHammer Phenomenon

From a load to a DRAM command. DRAM is organized into banks containing rows. Activating a row exposes its contents to the bank's row buffer; precharging prepares the bank for a subsequent activation. Normal refresh operations preserve stored charge. The path from a software load to these commands includes caches, outstanding-miss tracking, controller queues, and scheduling. Cache hits need not access DRAM, multiple requests can merge behind one outstanding miss, and accesses to an already open row need not cause a new activation. Thus software accesses, LLC admissions, and DRAM ACT commands are not interchangeable counts.

Physical disturbance. Repeatedly activating aggressor rows can disturb nearby cells and, under vulnerable device and access conditions, cause data errors before charge is restored. This is the RowHammer phenomenon (Kim et al., 2014). A neighboring-row illustration is useful for introducing the concept, but is not a complete description of every attack variant or device behavior. The physical victim row in that illustration must not be confused with a tenant whose program merely slows down because memory service is congested.

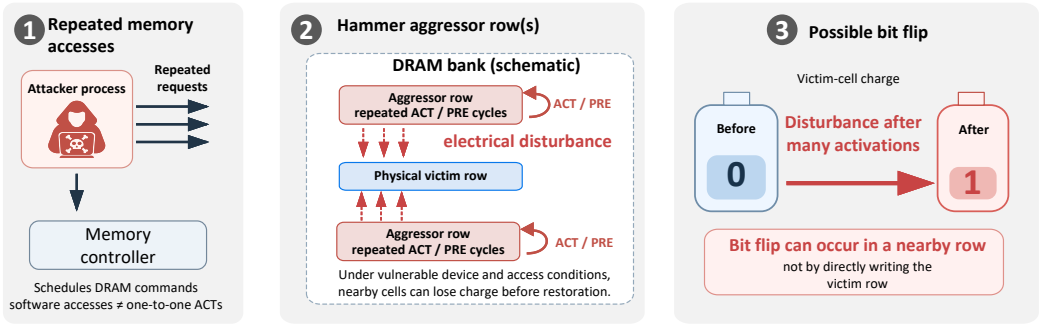

Figure 6: The RowHammer phenomenon in three steps (schematic, following Kim et al. (2014)). *Panel 1:* software requests become DRAM activations, but caching, miss merging, and row-buffer state break the one-to-one mapping. *Panel 2:* repeated ACT/PRE cycles on aggressor rows disturb cells of a nearby victim row. *Panel 3:* the disturbance can flip a bit without the attacker ever writing that row. Concept aid, not a device model: HammerEVO does not simulate bit flips.

Reading Figure 6 left to right makes two points this work depends on. First, the attacker never touches the victim's data: the damage is a side effect of legitimate activations, which is why mitigations must act on *activation patterns* rather than on data accesses. Second, every stage of the path—request, command, activation, disturbance—has its own counting semantics, so a "hammer count" observed anywhere above the DRAM is already an approximation.

What is and is not simulated. HammerEVO models request execution and mitigation-related activity. It does not model device-specific bit flips, physical cell susceptibility, temperature dependence, or proprietary physical-row adjacency. Its address-pattern names do not expose these quantities to the researcher. In particular, a virtual-address stride or a DSL field named `row_distance` does not establish physical adjacency after address translation. The safety of the selected D0 is assumed within its stated configuration; this work measures performance rather than independently certifying physical memory integrity.

Several thresholds have different meanings. A physical disturbance threshold describes device vulnerability; a D0 trigger threshold determines when a mitigation acts; a D1 detector threshold controls performance isolation; and an experimental success threshold is a rule for judging measured outcomes. The configured `nrh`=4096, a modeled D0 count such as 2048, the FASTWIRE (the evolved AQUA D1 policy) admission tripwire 256, and a 5% IPC-loss gate therefore cannot be compared as though they were one quantity. Each must be reported with its units, counting event, time interval, and configuration.

A.3 Baseline RowHammer Mitigations: The D0 Role

Tracking and preventive actions. We use D0 to name the underlying safety role. A mitigation can monitor activation activity using counters, compact tracking structures, or probabilistic decisions, and perform preventive work such as refreshing potentially affected rows or moving data. Tracking representation and preventive action are separate design dimensions; counting, refresh, and migration should not be treated as three mutually exclusive classes. Graphene applies compact heavy-hitter-style activation tracking (Misra & Gries, 1982; Park et al., 2020), while AQUA uses quarantining and migration (Saxena et al., 2022).

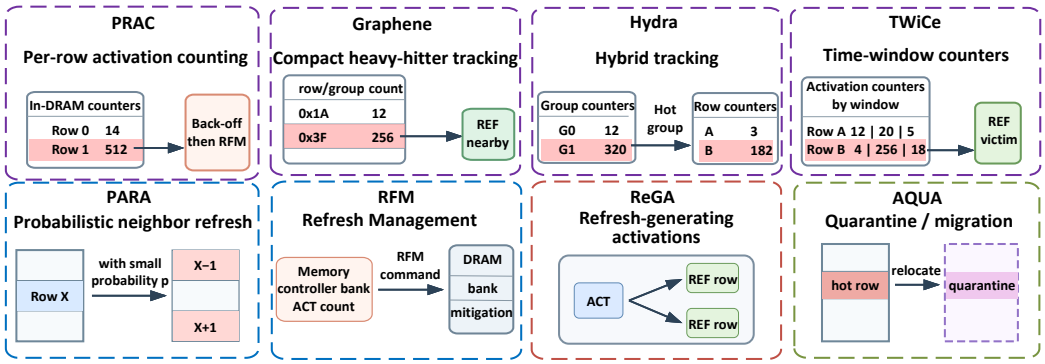

Figure 7: The eight safety-mitigation (D0) families studied in this work, drawn along the two design dimensions that matter for the performance side effect: *what is tracked* and *what preventive work is triggered*. Schematic logical roles, not hardware block diagrams.

Figure 7 walks through the eight families. **PRAC** keeps per-row activation counters inside the DRAM device; when a row's count crosses the configured threshold the device backs off ordinary requests and performs refresh management (Canpolat et al., 2024b). **Graphene** compresses activation tracking into a compact heavy-hitter structure over rows and row groups and refreshes the neighbors of identified aggressors (Park et al., 2020). **Hydra** tracks activations hybridly: cheap group-level counters flag a hot group, and only then are per-row counters consulted, so its observable activity mixes counter-metadata traffic with victim refreshes (Qureshi et al., 2022). **TWiCe** counts activations within a time window per row and refreshes the victim rows of whichever rows exceed the windowed rate (Lee et al., 2019). **PARA** (Kim et al., 2014) keeps no counters at all: on each activation it refreshes the neighboring rows with a small probability p, so its preventive work is stochastic and spread thinly over all traffic. **RFM** is the DDR5 refresh-management command path: the controller counts bank activations and, past a threshold, issues an RFM command under which the bank performs its mitigation (JEDEC Solid State Technology Association, 2020; 2024). **ReGA** moves the action into the device: a refresh-generating activation both serves the request and refreshes a potential victim row (Marazzi et al., 2023). **AQUA** relocates hot rows into a quarantine region, paying migration and bookkeeping work instead of refreshes (Saxena et al., 2022).

Two observations connect this figure to the rest of the paper. First, the *trigger surfaces* differ as much as the actions: a per-row threshold (PRAC, TWiCe), a compact-structure identification (Graphene, Hydra), a bank-level count (RFM), a coin flip (PARA), a hot-row migration decision (AQUA). A pressure pattern that maximizes preventive work in one family can be nearly invisible to another—the formal reason attack search must be per-family (§3.3). Second, every family places its preventive work on a shared service path used by benign tenants: in-DRAM actions occupy banks, refreshes occupy command slots, metadata and migration occupy controller and data bandwidth. That occupancy, not the bit flips the D0 prevents, is the side effect a D1 must isolate.

AQUA as a running example. AQUA relocates an aggressor row into a quarantine region to disrupt the physical exposure associated with repeated accesses. Tracking and mapping state identify when and where to move data. The migration entails memory operations and bookkeeping, so preserving safety need not be free in terms of service time. The experimental implementation exports a migration counter; this counter establishes that modeled preventive activity occurred, but its value alone does not measure the resulting benign slowdown.

Hydra and metadata activity. Hydra illustrates a different tracking and traffic structure, combining group-level and fine-grained row counting with stored counter metadata. Metadata operations and victim-row refresh activity can interact with ordinary memory traffic. The relevant observation for a D1 may therefore differ from the burst-rate signal useful in the archived AQUA case. The current Hydra experiments use explicit simulator counters and a versioned pressure-sideband extension; they should not be interpreted as a claim that every hardware deployment already exports this interface. The original Hydra design motivates hybrid tracking at low thresholds (Qureshi et al., 2022).

Physical location versus logical role. D0 is our logical safety layer, not a claim that all mechanisms reside inside commodity DRAM. Real mechanisms can involve controller-side and device-side facilities. The simulator realizes mechanisms through its own component interfaces; those interfaces do not establish that a commercial chip exposes the same observations. Our registered set (Figure 7) contains AQUA (Saxena et al., 2022), Graphene (Park et al., 2020), Hydra (Qureshi et al., 2022), PARA (Kim et al., 2014), PRAC (Canpolat et al., 2024b; Woo et al., 2025), ReGA (Marazzi et al., 2023), RFM (JEDEC Solid State Technology Association, 2020; 2024), and TWiCe (Lee et al., 2019). These mechanisms differ in tracking, triggering, preventive work and implementation location; their configuration semantics must not be treated as interchangeable.

A.4 Why Safety Mitigation Needs Performance Isolation

The performance problem. Three effects must be distinguished: physical disturbance threatens data integrity; D0 preventive work consumes service resources; and demand traffic itself competes for those resources. A tenant can impose enough pressure to degrade co-runners even when D0 continues to protect data. BreakHammer motivates suspect-thread throttling as a response to the performance costs surrounding RowHammer mitigation (Canpolat et al., 2024a). Our task asks how to discover a bounded isolation policy in such a memory system.

A performance victim need not own a row adjacent to an aggressor. Its requests can wait behind other requests, metadata accesses, or preventive operations elsewhere on a shared path. Conversely, simultaneous D0 activity and slowdown do not prove that all the slowdown is caused by that activity.

What a D1 observes. D1 receives the events and bounded state permitted by its interface: request source, principal mapping where exposed, selected cache/miss and admission events, and registered controller telemetry or sidebands. It does not receive a ground-truth maliciousness label. In particular, an attacker-specific simulator counter is a grading aid, not an admissible detector input. Physical-neighbor information and hidden validator answers are likewise not provided as policy observations.

What a D1 changes. A D1 can shape admission of new requests, impose bounded resource budgets, or change service priority through permitted actuators. It cannot disable D0, fabricate its preventive counters, or discard already accepted requests to claim a speedup. The control path can span LLC admission and memory-controller scheduling; the policy's logical role is not limited to a single hardware box. A policy must preserve the specified progress conditions, including for a principal whose requests are being restricted.

A concrete policy. FASTWIRE (shipped as FASTWIRE-256H) counts admissions in reset-based 512-cycle epochs and latches a principal when its count reaches 256. An already exposed APS (activation-pressure sideband, a read-only export of D0 pressure events) witness provides a second arming path. The latch changes admission and service-class settings; the final candidate uses a token period and denial-bound parameter of 32,768 cycles. These constants describe this policy and fixture, not D1 in general. The three calibration cases informed its threshold, and its measured behavior must not be generalized to arbitrary benign workloads without further testing.

Hardware extension is a separate stage. A new observer changes the research interface. It requires explicit implementation, resource accounting, validation, and a new frozen arena version before its results can be compared. This is distinct from a candidate using capabilities already present in a contest. C++ execution and model-level state/timing audits do not replace RTL implementation or silicon validation.

A.5 How Attacks and Defenses Enter the Experiment

From attacker behavior to traces. The red researcher proposes a structured request program or a strategy template. Its parameters describe allowed workload behavior, such as phase structure, working-set patterns, request spacing and cooperating cores. A bounded search may instantiate numerical parameters, after which a deterministic compiler emits per-core trace records. The frontend interprets these records, address translation maps requests into the modeled memory system, and

normal simulated execution determines cache misses, admissions, DRAM commands and mitigation actions. The researcher does not submit a desired slowdown value or directly set measured counters.

The variable-length trace-program path uses virtual-address patterns. Names such as `stream`, `page_cycle` and `row_hammer` describe the generator's intent rather than guaranteeing physical row placement. This abstraction enables bounded, reproducible experiments but leaves a gap to a complete implementation inside a real tenant VM.

The combined workload. Generated attack traces run alongside fixed benign traces in the same simulated machine. Each evaluation freezes its core allocation, traces, translation, D0 configuration, seed and horizon. In a pure reference, memory-silent padding occupies the absent attacker's slots so that the host geometry does not change. Padding cores and attacker cores are excluded from benign-core statistics. Identical trace files on different benign cores remain separate per-core observations.

Freezing the opponent. During one blue design task, the corresponding red attack is frozen. The blue researcher submits an observation probe or candidate policy, which passes contract audit and compilation before matched simulation. The red side can subsequently search against a frozen incumbent defense. An attack portfolio can contain different attacks covering different D0 configurations; one attack need not cover all eight. The original multi-core wide-pressure attack and the later single-core low-rate attack are separate artifacts and should not be silently substituted in a comparison.

A.6 Attack Qualification: The 50/50/25 Rule

Matched reference for each deployment. Let $\kappa = 0$ denote D0-only and $\kappa = 1$ denote D0 plus the incumbent D1. For each benign core i, define

$$r_i^{(\kappa)} = \frac{IPC_{\text{attack},i}^{(\kappa)}}{IPC_{\text{pure},i}^{(\kappa)}}, \quad \ell_i^{(\kappa)} = 1 - r_i^{(\kappa)}, \quad m^{(\kappa)} = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \ell_i^{(\kappa)}.$$

The pure denominator belongs to the same deployment κ. This measures attack-induced degradation separately from that deployment's own no-attack overhead. The broad rule requires, for both $\kappa = 0$ and $\kappa = 1$,

$$\frac{\#\{i : r_i^{(\kappa)} \le 0.50\}}{|\mathcal{B}|} \ge 0.50 \quad \text{and} \quad m^{(\kappa)} \ge 0.25.$$

These are task-defined severity thresholds, not universal RowHammer constants or thresholds taken from the original BreakHammer paper.

For 16 benign cores, at least eight must run at half speed or less. Eight cores retaining 50% and eight retaining 100% meet both gates exactly. If the latter eight instead retain 120%, coverage still passes but average loss is only 15%, so the attack fails the broad rule. If all cores retain 70%, mean loss is 30% but none is at half speed, so coverage fails. Speedups remain negative losses; values are not clipped or rounded before threshold comparison. Core-equal-weight loss is not tenant-equal-weight loss or the ratio of aggregate IPC totals.

Activity and validity gates. The current red checker additionally requires D0 preventive activity in the D0-only attack arm, complete trace-matched benign-core records, fixed-horizon consistency, and the safety/evidence checks. Activity floors are per-case or command parameters and must be reported for the run in question. A zero-attack IPC can represent complete loss; a missing metric, nonfinite value, invalid pure denominator, or mismatched core cannot. An untriggered-D0 diagnostic run does not establish the same claim as a formal trigger-qualified attack.

Local qualification is a different mode. Local mode reduces the affected-core requirement to at least one benign core at half speed. The current implementation also sets the average-loss threshold to zero and still checks $m^{(\kappa)} \ge 0$. This differs slightly from a literal absence of a mean constraint when other cores speed up.

A.7 Defense Qualification: Performance, Overhead, and Progress

Six deployments. For each case, the blue validator runs D0-only, D0 plus incumbent D1, and D0 plus candidate D1, each with attack traffic and with a pure-benign reference. Matching the traces alone is insufficient: host geometry, mapping, seed, configuration, executable identities and horizon must also match. These six arms distinguish the attack's harm, the candidate's recovery, and the candidate's no-attack cost.

Absolute residual impact. Let P_i be core i's IPC in D0-only pure, X_i in candidate attack, and Z_i in candidate pure. Define

$$\iota_i = 1 - \frac{X_i}{P_i}, \qquad o_i = 1 - \frac{Z_i}{P_i}.$$

The attack-period performance gates are

$$\frac{1}{|\mathcal{B}|} \sum_i \iota_i \le 0.05 \quad \text{and} \quad \max_i \iota_i \le 0.10.$$

The normal-speed anchor is D0-only pure, not candidate pure or an incumbent that may already be slow. Using only X_i/Z_i would allow a policy that slows both its attack and pure arms to appear protective.

Pure-benign overhead. In the frozen AQUA contract-compliant development case, the mean, 95th-percentile and worst o_i must be at most 1%, 2% and 3%, respectively, together with the applicable transparency check. In the predeclared g4/g8 benign-pressure stress cases, mean overhead must be at most 3% and worst overhead at most 5%; p95 is reported but is not a separate hard gate. Stress cases require bounded progress rather than an unconditional absence of all resource-control actions. The scenario type is fixed before evaluation, not chosen after seeing results.

Misclassification and liveness. In the current attack-arm security checks, benign admission denials and benign QoS changes must be zero, and benign throttling may not exceed the incumbent's count. For enabled principals with demand, admission telemetry must show progress, maximum denial no greater than the declared bound, and a horizon long enough to observe that bound. Relevant quota, service-class, probation and scheduler checks also apply. These are resource-progress requirements, not a guarantee that every individual memory request completes within the denial-bound interval.

D0 activity and benefit checks. The frozen AQUA blue cases require at least three migrations and at least one migration per million cycles in the D0-only attack arm. The archived record reports four migrations in three million cycles. The candidate arm is not required to retain that trigger count: suppressing the offending activity can be the intended effect of a defense. Per-core recovery relative to D0-only attack must also meet the case's minimum, zero in these frozen cases. The configured recovery/cost target of three is diagnostic when its enforcement flag is false.

Fields named `min_attack_improvement` and `max_pure_benign_regression` remain in legacy criteria records, but current code marks them as telemetry-only. They should not be presented as additional hard gates. Final success requires every selected case to pass all active checks; passing one development case does not establish full validation or unseen-workload generalization.

Attack failure is not defense success. A measured outcome can lie between the red severity threshold and the blue recovery threshold. For example, an outcome with 15% mean benign loss may fail the broad red average-loss gate while still exceeding the blue 5% gate. Invalid deployment evidence is a third outcome, not a measured performance failure and not a success for either side.

B Evolved D1 Mechanisms and Design Analysis

This appendix explains the mechanism of each evolved D1 policy referenced in the main text, the design lessons their trajectories produced, and the conditional limits of any formal defense claim.

B.1 Per-Family Mechanism Notes

AQUA — D1 v6 (six versions, two passing revisions). A per-principal admission tripwire keyed to AQUA's concentration threshold: once a principal's admissions in a window cross $B{=}256$, further requests are delayed rather than served. The threshold sits between the recorded benign window peak (124) and the attack peak (366), with benign-side and attacker-side margins of 2.06 and 1.43 (§B.2).

Hydra — D1 v8 (seven accepted versions, v2–v8). The line that established observation precedes action. Early candidates judged pressure from controller-local counters and were circumvented by dispersed patterns that stayed under every local threshold while still triggering preventive work. The expert-approved hardware extension—cross-controller occupancy witnesses—made the pressure visible; the final policy latches on accumulated occupancy evidence rather than instantaneous counts. The same skeleton transferred to Graphene, PRAC, and ReGA (below).

Graphene / PRAC / ReGA — D1 v2, v6, v2. Knowledge transfer done honestly: the occupancy-latch skeleton was ported, then re-calibrated against each family's own census rather than copied with Hydra's thresholds. ReGA's final calibration (tiers 64/256/4096) came from its own ladder census after a first attempt exposed that tier-1 was lazy under the runtime's enqueue rule; PRAC's calibration similarly adjusted to PRAC's per-bank action cadence.

PARA — D1 v32 (32 accepted versions). The longest chain and the clearest record of the loop correcting an over-strong design (Figure 4). Version v14 added a row-set sketch with permanent probation and immediately reached 4/6 defended literature families; v15–v25 regressed, bottoming at 0/6 (v18, v21, v22), because instantaneous evidence is ambiguous under dense benign patterns and permanent punishment turned every misjudgment into lasting collateral damage. Recovery came from making the intervention revocable—probation that decays unless re-confirmed (v26, back to 4/6)—and gating it on pressure evidence (v31–v32, 6/6).

RFM — D1 v12 (ten recorded versions). Early versions scored principals against a running baseline of normal behavior; sustained attacker pressure shifted that baseline, so benign bursts began to be misclassified as aggressive—the defense taxed the victims. v12 anchors the reference to benign-only measurements and derives suspicion from relative-miss deficits rather than absolute rates.

TWiCe — D1 v7 (seven accepted versions). The final version is a dual-latch design over inter-arrival timing tiers (16/64/256, calibrated from TWiCe's own census), combining a fast latch for bursty pressure with a slow latch for sustained residence.

B.2 Scope of a Formal Defense Claim

For a high-value tripwire, let b_{ben} be the largest observed benign feature value and a_{atk} the smallest target-attack feature value across the specified calibration cases. Empirical separation requires $b_{\mathrm{ben}} < B \leq a_{\mathrm{atk}}$, and a two-sided multiplicative margin $\gamma > 1$ requires $a_{\mathrm{atk}}/b_{\mathrm{ben}} \geq \gamma^2$. For the recorded AQUA window peaks, $366/124 = 2.95 < 4$, so a two-sided $2\times$ margin is impossible on those values; $B = 256$ instead has benign-side and attacker-side margins 2.06 and 1.43. This is an empirical condition, not a distributional guarantee on unseen workloads, and overlap in one feature does not prove that joint or temporal detectors cannot work. The held-out 12.0% worst-core failure in the AQUA seed sweep (Appendix L) is a concrete counterexample to extending the calibrated performance gate to all seeds.

Likewise, the conditional no-trigger bound of Appendix O states only that *if* attacker-attributable activations before latching, in-flight contributions, and post-latch service are bounded as assumed, then no allowed trace can cause the specified D0 trigger within the horizon. The current data do not certify those premises for the full runtime, and a no-trigger statement would not imply benign performance even if certified: ordinary traffic can still occupy shared resources. We therefore make no unconditional "unbreakable" claim; the defenses are empirical artifacts with measured boundaries, reported in §4.5 and the appendices.

C Literature Battery: Configuration and Adaptation

This appendix documents how the six literature attack families used in §4.5 were adapted into the simulator, and what each adapted trace can and cannot certify. The cited papers do not evaluate our defenses, and their physical bit-flip outcomes cannot be imported as our simulator measurements: each family was re-expressed as a legal trace program preserving a specific stress dimension, and every case archives the generated trace, address mapping, timing/configuration, source binary, and six-arm verdict. The battery is complete: all 48 family–attack cells have been measured (§4.5), with per-cell values in the data package.

Table 4: Literature attack families and what the adaptation preserves. "Partial" means the trace shape is expressed but the original physical semantics are not reproduced: a trace-shape stress test, not evidence about the original attack's device behavior.

Source family	Preserved stress dimension	Required check before scoring
TRRespass (Frigo et al., 2020)	Many-sided aggressor schedule	Map virtual addresses to banks/rows; audit D0 trigger.
BLACKSMITH, SMASH (Jattke et al., 2022; de Ridder et al., 2021)	Nonuniform phase and synchronized bursts	Check trace timing and refresh-clock representation.
Half-Double (Kogler et al., 2022)	Multiple aggressor distances	Physical adjacency and disturbance effects are not modeled; trace-shape test only.
Phoenix (Meyer et al., 2026)	Long refresh-aware sequence	Refresh synchronization semantics added before scoring.
PRAC availability, DAPPER (Canpolat et al., 2024b; Woo & Nair, 2025)	Preventive-action/metadata pressure	Measure D0 action, controller occupancy and source attribution.
RowPress (Luo et al., 2023)	Extended row-open residence	Requires a modeled row-open interval and a separate threat contract; partial.

The closest directly relevant comparison is mitigation overhead and shared-tracker pressure, not an assertion that preventing a D0 trigger prevents every memory denial of service. A cell is called defended only when the case is valid and the stated performance, overhead, and progress gates all pass under the uniform gate (θ_A^{mean}, θ_A^{worst} of §3.4); failures remain visible in the battery rather than being removed—the Hydra Phoenix cell (§4.5) is the standing example.

D Artifact and Protocol Details

The arena registers AQUA, Graphene, Hydra, PARA, PRAC, ReGA, RFM, and TWiCe configurations. These are different D0 mechanisms and are not assumed to have identical trigger semantics. The formal AQUA fixture uses 32 modeled cores, one attacker principal with up to 16 cores, a fixed 3,000,000-cycle horizon, and matched benign traces. Host CPU allocation controls simulation scheduling; it does not define the modeled workload's core count.

The policy entry point is `blue_policy_logic(BluePolicyCtx& ctx)`. Exposed operations include read-only event and telemetry access, per-principal admission control, and service-class control. State and capability declarations are audited. Timing and storage accounting are model-level constraints; this work does not include a completed RTL or silicon implementation.

The public task package contains the attack JSON, a policy seed, contract schema, compiler and grader entry points, and feedback format. Later comparison runs used GLM-5.3 via a gateway, with Opus-labelled relay runs for selected comparisons. The archived discovery instead used ZCode's GLM-5.3-family coding-plan deployment. The equivalence of these model instances and sampling configurations is unverified.

E METRIC REFERENCES AND GATE SEMANTICS

Mean and worst-core losses are defined in Section 3. The core-equal-weight mean is distinct from aggregate throughput loss, $1 - \sum_i a_i / \sum_i p_i$. For a candidate's attack impact, the reference is the D0-only pure-benign arm prescribed by the validator; pure overhead is checked separately against the corresponding pure reference. Different denominator choices must not be mixed in a comparison.

The broad qualification gate combines the affected-core fraction, mean loss, D0 triggering, and the remaining evidence/progress checks. The local gate has a different affected-core condition. Nonzero impact alone is neither broad nor local qualification. A no-trigger result excludes the protocol-defined class even if other memory interference remains.

The historical partial fitness is

$$s = 0.1 + 0.9 \left[\tfrac{1}{2} \max(0, 1 - m/0.05) + \tfrac{1}{2} \max(0, 1 - w/0.10) \right]$$

for a buildable, measured candidate. This score is flat while both performance gates are violated. It is therefore unsuitable as evidence that an evolutionary method received a useful ranking over all partially effective policies. Invalid metrics must be represented as unavailable; a scheduling failure score is not a measured performance value.

F CENSUS AND CANDIDATE HISTORY

The following measurements come from the archived P0/P1 census (the two observation probes submitted during the AQUA design session: P0 logged admission peaks and witness timing, P1 logged threshold crossing times), not from generated or interpolated curves. Peaks are successful admissions within the probe's 512-cycle counting convention.

Table 5: Comparable census observations. *dev* is the development seed; g4/g8 are the predeclared benign-pressure stress cases with benign cores grouped into larger principals (Appendix A). No independent-replication error bars are implied.

Case	Pure benign peak	Attacker peak	Attacker / benign
dev	38	366	9.63
g4	79	366	4.63
g8	124	366	2.95

The attacker's 26,834 requests are a horizon total. The g8 benign principal's reported horizon total is 47,953, so totals do not yield the same separation. The policy threshold $B = 256$ is 2.06 times the worst measured benign peak; the attacker peak is 1.43 times B. There is no two-sided factor-of-two margin. The dev and g4 rows show the same monotone trend (38 peaks at 9.63× and 79 at 4.63×, respectively, versus 124 at 2.95× for g8).

P1 observed the attacker crossing 128 at cycle 3 and 248 at cycle 6; no benign principal crossed 128 in the inspected cases. The first APS tier-2 witness in the wide-envelope probe arrived around cycle 102,314, motivating an early rate wire and a later witness backstop. All-zero telemetry from an undeclared or unexercised observer cannot demonstrate that the corresponding behavioral feature is absent.

Table 6: Research actions and recorded outcomes in the archived session. Defense-candidate counts exclude the two probes and internal model/tool calls.

Action	Measurement or change	Recorded outcome
P0	Admission peaks and APS witness timing	Observation only
P1	Threshold crossing time and per-core peaks	Observation only
v1	$B = 256$, latch and bounded admission	pass
v2	Align denial bound and token period at 32768	pass

The v1/v2 labels are internal session revisions and cannot be mapped line-by-line to the archived validation records; the authoritative numbers are those of the shipped policy's archived evidence (3.4% development mean / 8.4% grouped worst loss).

The historical loop's TWIL record reports 14.16% mean and 26.05% worst-core loss, with no full pass. It belongs to the preceding research trajectory and is not included in the later framework-ledger aggregates. FASTWIRE's two passing revisions are from one research session, not two independent successful discoveries. The archive reports approximately 59 minutes for the successful session, but its model-runtime usage and the loop's token accounting are not normalized. We make no proposal-count-to-compute speedup claim.

G Implementation Semantics of FASTWIRE

FASTWIRE stores an epoch anchor, a counter, and a latch in a packed per-principal word. If the current cycle is at least 512 cycles after the anchor, the policy resets the count to one and moves the anchor; otherwise it increments the count. This is an event-anchored reset window, not an exact arbitrary sliding window. The latch fires at 256. Cross-boundary bursts require separate analysis.

The final policy sets both the token period and maximum-denial parameter to 32,768 and clamps stored tokens on arming. It polls APS from the LLC-new-miss event branch. The guarantee therefore depends on when this branch executes, how queued events are consumed, and what work was admitted before the latch. Historical APS and topology extensions are part of the supplied substrate; zero additional units for this candidate does not mean zero hardware cost for the system.

H Deployment and Reporting Audit

The September 22 audit found two concrete reporting problems. First, the previous census plots used hard-coded sequences that combined a 124-request window peak with the attacker's 26,834-request horizon total and labelled the ratio as $216\times$. Those curves and claims have been removed; Figure 4 uses comparable peak counts. Second, the no-telemetry candidate `20260915-200837-da056b` has null mean/worst metrics and a shared-library hash mismatch in its validation record. The adapter's fallback converted null to 1.0, which was incorrectly plotted as 100% loss. This record is now marked invalid. The same fallback also misinterprets a legitimate zero and must be repaired before new performance aggregation.

Five apparent attack successes were separately overturned by fixed-binary replay: flash broad round 15, GLM local rounds 19 and 20, Opus-labelled broad round 8, and DeepSeek-labelled broad round 11. The arbitration attributes them to deployments where the intended defense was not exercised. These five replays support rejection of those success claims. They do not constitute a replay audit of every failed candidate in the campaign.

The historical campaign summaries report 57 completed design rounds and 154 attack–round evaluations across four channels, with no retained qualifying attack against the AQUA composite after arbitration. These are descriptive historical totals. Missing or invalid deployment evidence must not be treated as a valid negative security test.

Table 7: Infrastructure defects and required interpretation.

Event	Interpretation / action
Shared build drift	Verify loaded code and library for all matched arms; use isolated candidate builds or one transaction spanning build and validation.
Missing performance	Mark unavailable; never reinterpret a sentinel as measured harm. Replay affected candidates and separately assess corrupted search feedback.
Swallowed evaluator output	Disclose periods when methods did not receive intended feedback; repaired output does not retroactively repair search history.
Task/patch adapter errors	Separate paths, schema and patch failures from valid runtime performance comparisons.

I Observability Census on Hydra

The September 22 census used a different frozen attack, `014628fc`, a single-core, low-rate, high-footprint pattern. It measured admission counters and row-pattern observers under development and grouped benign conditions.

Table 8: Selected Hydra census observations; the named benign case is reported explicitly rather than as a global maximum. These marginal statistics do not establish detector impossibility.

Signal	Benign observation	Attacker
512-cycle admission peak, g4	79	87
Horizon admissions, g8	30,469	20,773
New misses / policy window, g8	4,495	3,185
Alternations, window maximum, g8	427	7
Row-reuse sketch	approximately 64	4
Mean request latency (cycles)	4,734–5,629	14,390

The no-D1 arm also leaves more new misses per policy window at g8 (4,495 vs. 3,185), consistent with the admission-rate and row-reuse differences above.

Latency is not currently the same observation as an admissible per-request decision signal, and may reflect congestion rather than an intrinsic attacker identity. A low marginal value also need not be indistinguishable from benign behavior when other features and history are available. The census rules out simple high-threshold designs with the proposed margin on these cases; it does not rule out the whole D1 program space.

A new Hydra RCT-pressure observation extension was implemented by instrumenting the D0 counter-update site and reusing the existing sideband. The current acceptance records test observability and unchanged D0 behavior; observation-only policies are expected to fail the defense-performance gate. No end-to-end defense success is claimed from those acceptance records.

J Conditional-Certificate Assumptions

Table 9: Model obligations and implementation status. These replace the previous unqualified lossless-sideband assumption.

ID	Obligation
A1	Fix workload, translation, clock domain, horizon and executable identities.
A2	Preserve D0 semantics; distinguish attacker-attributable activity from legitimate background D0 triggers.
A3	Define the trigger-qualified predicate separately from general performance interference and from the defense-performance gate.
A4	Establish the relevant per-row activation threshold T and reset rules.
A5	Bound pre-latch activations, witness-delivery work and residual in-flight requests; do not equate admissions and ACTs without a mapping.
A6	Prove bounded delivery of a relevant witness for every triggering source/row, including overflow and delayed polling. Not yet established.
A7	Derive post-latch service and forced-liveness bounds from the deployed runtime. A parameter value alone is not this derivation.

The sideband audit inspected 63 records, 45 with telemetry. Five showed nonzero drops (368 in total, maximum 228). Delivered tier-2 events in those deployments show that overflow did not suppress every witness; they do not prove that every relevant witness was delivered in time. The prior assumption of a lossless FIFO is not satisfied by these data.

For the idealized values $C_{\mathrm{pre}} = 256$, $Q = 0$, $D = 8192$, $H = 3{,}000{,}000$, the bound is $256 + 367 = 623 < 2048$. The integer sufficient threshold is $C_{\mathrm{pre}} \leq 1680$ under those fixed assumptions. At longer horizons or smaller spacing this certificate may cease to apply. Its failure is not a constructive attack, and it makes no claim about non-triggering occupancy denial of service.

K Parameter Sweep and Calibration

The grid held the discovered policy structure fixed and varied parameters. Development and grouped calibration cases answer different questions. Only tested values are reported; no continuous feasible interval or optimality guarantee is inferred.

The archived tier-2 calibration record tested the APS witness threshold: tier2 values of 32 and 64 were rejected for benign false positives, and tier2=128 was adopted. Only tested values are recorded; no continuous feasible interval is inferred. The tier-2 witness threshold is a different parameter from the admission-count bound B used elsewhere in this paper, and values of one must not be read as calibration evidence for the other.

L Session Battery and Held-Out Seed Protocol

Session battery. Eight arms, three independent sessions each, on one tier (`glm-5.3-flash` through one local gateway). Controls held fixed across arms: model deployment, task version (frozen attack, seed policy, contract schema), budget (20 proposals per session), stopping rule (gate pass or budget exhausted), and raw-feedback access (differences confined to the factor under test). Each session starts from a fresh history and context; candidates within a session are not independent trials. Arms: (i) fixed census data injected (benign peak 124, attacker demand 26,834, APS events 13 vs. 0) with probe authoring forbidden; (ii) a self-designed first probe (the Stage-0 rule); (iii) vanilla loop, the shared control ($\equiv$ no-history); (iv) continuous per-case telemetry feedback; (v) a frozen campaign context (attack profile, census numbers, failure taxonomy; no solution mechanism); (vi)–(viii) the methodology transplant, full, minus census, and minus diagnosis directives. This 24-session battery is pre-registered but has not been executed; we report the protocol only.

Held-out seeds. The frozen $B{=}256$ configuration (code fingerprint `d1a7c85d`) is re-validated in final mode on the three-case gate with seeds $\{234, 345, 456, 567, 678, 789, 890, 1234, 2345, 3456\}$, none used in design, calibration grids, or the frozen gate. Per-seed outcomes: three passes (456, 567, 678: mean 0.76–1.19%, worst core 2.9–8.5%, bracketing the frozen verdict); one measurable failure (789: mean 1.4%, worst core 12.0%); six seeds on which the evidence pipeline fails closed under concurrent shared-binary rebuilds by the live campaigns (per-arm simulator-hash mismatches and incomplete incumbent runs—the build-drift family) so no measurement exists; retried runs under quieter windows are collected separately and never silently merged. We report pass rates over measurable seeds only, disclose the fail-closed count as a grader limitation, and do not extrapolate to unmeasurable seeds.

M Impact and Release Scope

The intended use is performance isolation around trusted RowHammer mitigations. Automated adversarial search can also reduce the cost of finding denial-of-service traffic. Artifacts therefore preserve the D0/D1 boundary and separate performance results from physical memory security claims. We will release sanitized evidence and protocols subject to anonymity and responsible disclosure.

N Outstanding Validation

The defense-centered milestones take priority over additional autoresearch rankings: freeze the eight D0/baseline-D1 configurations; validate their attack counterexamples and new D1 policies; re-attack frozen defenses; evaluate held-out workloads and the four-controller/eight-channel model; and compare specialized versus unified designs with hardware accounting. These milestones map to the figures and tables in Sections 4.2–4. The following historical requirements are also retained; their labels identify evidence obligations rather than a ranking of research priorities.

1. E0: complete immutable-build replay and the invalid-record audit.
2. E1: compare fixed and self-designed measurements using the same model deployment, history, feedback access, and total compute budgets.

3. E2: test the Hydra observation intervention with matched policies.
4. E3: connect the conditional certificate to queues, event delivery, clock domains and forced-liveness behavior.
5. E4: evaluate the frozen policy on genuinely unseen benign workloads (seed-level re-validation in Appendix L; workload-level generalization remains open).
6. E5: assess at least one search baseline with valid continuous feedback outside the pass gate.
7. E6/E7: equal-budget disclosure and causal harm decomposition.

These are planned checks, not results. Historical case-study findings remain labelled retrospective.

O Extended Design Analysis and Historical Certificate

O.1 Observability Is a Testable Explanation

The AQUA case suggests that the choice of measurement matters: horizon-total admissions do not separate the attacker from the heaviest benign principal, whereas the measured 512-cycle peaks do. A separate census on the Hydra task found low attacker admission rate, low alternation, and low row reuse relative to some grouped benign principals (Appendix I). This is evidence against the tested high-threshold detectors. It is not a proof that every detector using the interface must fail: joint features, temporal structure, and low-value predicates remain possible, and a $2\times$ margin is a design heuristic rather than a necessary condition for classification.

Hardware observability is also versioned. FASTWIRE reuses an APS interface added before the frozen task; a later principal-topology extension and a new Hydra pressure-sideband effort change the available design space. The latter has observation-probe acceptance records, but those records are not a passing Hydra defense. Any sideband-on/off comparison must freeze the remaining policy and workload conditions.

O.2 A Conditional Finite-Horizon Certificate

The campaign motivates a counting argument, but implementation evidence must establish the argument's premises. Let T be the number of attacker-attributable activations needed to trigger D0 on a protected row. For each such row, let C_{pre} bound contributing activations before the relevant latch, including activations committed during witness delivery, and let Q bound additional activations from already admitted in-flight work. Let D be a guaranteed minimum spacing of subsequent contributing admissions, assuming at most one relevant activation per such admission.

Proposition 1 (Idealized no-trigger condition)**.** *In a fixed horizon H, suppose attribution and request-to-activation mapping are valid, every potentially triggering row reaches a permanent latch with bounds C_{pre}, Q, and post-latch service obeys the spacing bound D. If*

$$C_{\mathrm{pre}} + Q + \lceil H/D \rceil < T,$$

no attacker-attributable row reaches the D0 trigger threshold within H.

Proof. For any row, partition contributing activations into those before the latch, residual in-flight work, and work admitted afterwards. The premises bound these classes by C_{pre}, Q, and $\lceil H/D \rceil$, respectively. Their sum is below the trigger threshold. The argument is independent of how a request sequence is selected, provided the premises hold for it. □

The earlier idealization substitutes $C_{\mathrm{pre}} = 256$, $Q = 0$, $H = 3{,}000{,}000$, and $D = 8192$, giving $623 < 2048$. This arithmetic is correct, but it is not a verified certificate for the deployed policy. In particular, APS overflow occurred in archived runs; observations that some witnesses were delivered do not establish bounded delivery of every relevant witness. The actual policy sets a 32,768-cycle token period and denial bound, and the conservative spacing bound must be derived from the complete runtime, including forced service. Admission and ACT counts cannot be identified without accounting for queues and event ordering.

Under the idealized substitution, the sufficient region is $C_{\mathrm{pre}} < 1681$ (or integer $C \leq 1680$). Increasing the horizon beyond that region removes this certificate; it does not prove an attack exists.

Even a valid no-trigger certificate excludes only the trigger-qualified attack class. It does not establish absence of ordinary occupancy interference or benign performance loss.

O.3 Calibration Sensitivity and Evidence Fidelity

A parameter sweep over the discovered structure found $B = 64$ better than $B = 256$ on the development case, but $B = 64$ caused up to 99.9% worst-core loss on grouped benign validation cases. The archived $B = 256$ policy passed all three calibration/validation cases; larger tested values failed (Appendix K). A finite grid does not establish a continuous feasible interval, and g4/g8 are not unseen data because they informed the census.

Separately, a reporting defect mislabeled 320 validations across 33 rounds of two runs as configuration errors. Preserved stderr enabled post-hoc relabeling. This incident demonstrates that a rejected candidate's diagnosis can still be wrong. We have not quantified a controlled pre/post improvement in search quality; a causal feedback-fidelity claim requires a matched-checkpoint experiment.