ENTITY-CONDITIONED ADDRESSING: REDUCING CROSS-ENTITY INTERFERENCE IN FIXED-SIZE RECURRENT MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Fixed-size recurrent models compress events from many interleaved entities into a shared state. We identify a cross-entity interference failure mode that is distinct from long-range forgetting: performance can remain stable as sequence length increases yet collapse as the number of concurrently updated entities grows. We introduce Entity-Conditioned Addressing (ECA), which injects an entity-derived key into a recurrent model's write and read addresses. Our main variant, ECA-Hash, generates deterministic random keys from entity identifiers without maintaining a trainable entity table, thereby preserving streaming inference and state size independent of the entity count. On a controlled entity-state benchmark, matched identity codes supplied through the input fail as entity load grows, whereas injecting the same codes into the addresses raises accuracy at 16 tracked entities from at most 0.20 to 0.87 (Mamba-2) and 0.98 (Gated DeltaNet). On a second, independently generated benchmark, the effect transfers across official Mamba-1/2/3 blocks, Naju, and Gated DeltaNet, and approximately follows the predicted dependence on entity load relative to address width. On a real Windows endpoint state probe, address injection improves accuracy over the same model without it from 0.714–0.726 to 0.936–0.984 across official Mamba-1/2/3, Naju, and Gated DeltaNet (official Mamba-2: 0.722 to 0.984). However, it provides no consistent benefit when identity information alone or recent content suffices, and address injection can hurt content-dominated detection. These results characterize ECA as a structural prior for interference-bound entity tracking.

1 INTRODUCTION

Can a fixed-size recurrent memory track many interleaved entities without allocating one state per entity? Long streams in practice are rarely a single narrative: a Windows host interleaves thousands of processes, an authentication log interleaves users and machines, a dialogue interleaves speakers and referents. A linear-recurrent state-space model (SSM) processes such a stream with one shared state bank, so an update to one entity's representation competes with every other entity's. We call the resulting failure mode *cross-entity interference*, and this paper argues three things about it: it is a real, separately measurable failure axis of modern sequence models (a **finding**); the bottleneck is *learning entity routing*—with the same identity codes at the input, standard input-conditioning variants of selective SSMs, delta-rule memories, and ordinary input-conditioned attention fail to learn a reliable routing rule at the tested entity loads, while supplying identity as a *structural routing prior* (recurrence addresses, an attention-logit bias, or memory keys) substantially mitigates it in each architecture family we test (a **principle**); and ECA instantiates that prior for fixed-size recurrences without abandoning streaming inference or constant state (a **method**).

Finding: interference is not forgetting. Long-context evaluation varies how long information must survive but does not independently vary how many entities compete for a shared state. On a four-way security grammar, Mamba-2 stays accurate under $4\times$ length extrapolation with a single writer (Appendix F.1) but drops to near the no-identity floor once content-identical updates from multiple entities interleave (Table 3), so we vary entity load M independently of sequence length.

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.

Identity information is insufficient without a routing prior. We ask whether supplying a persistent identity prior directly to the addresses changes learning and generalization relative to input conditioning with the same identity codes. On the primary eight-way benchmark (es-SET, chance 0.125), every input-conditioning variant we evaluate stays at or near the no-identity floor for $M \geq 16$ on official Mamba-2 and Gated DeltaNet (GDN), while adding the same hash code to the addresses reaches .865 and .977 at $M=16$ (Table 2). On an independently generated four-way security grammar, input conditioning leaves a Naju backbone (Lim & Yoon, 2026) (a native-discrete gated SSM with rank-32 gate projections, Appendix D), the official Mamba-1/2/3 blocks (Gu & Dao, 2023; Dao & Gu, 2024; Lahoti et al., 2026), a Transformer encoder, and GDN at the empirical no-identity floor (0.33–0.36, above the 0.25 chance level because event content and label priors carry some signal); amplifying an input-only injection of the same key code recovers only part of the gap (0.69 at $16\times$), whereas address injection lifts the recurrent backbones to 0.81–0.96 (Table 3). On real endpoint telemetry, supplying the identifier as input moves an entity-state probe on Naju from 0.710 to only 0.714, whereas address injection reaches 0.975 (0.936–0.984 across five backbones). Section 7 and Appendix F report the strongest input variants we evaluated.

Method: Entity-Conditioned Addressing. ECA adds an entity key to the write and read address vectors of a scan-compatible recurrence, $\widetilde{B}_t = B_t + \lambda \mathbf{e}_t$, $\widetilde{C}_t = C_t + \lambda \mathbf{e}_t$. Our main method, ECA-Hash, uses a deterministic Rademacher hash of the entity identifier and therefore requires no trainable key table, supports unseen identifiers, and keeps the recurrent state fixed in the entity count. ECA-Learned, a learned-geometry ablation with one materialized key per entity, replaces the hash with a trainable table whose storage grows with the number of allocated entities. The recurrence, its associative scan, and streaming inference are unchanged; setting $\lambda=0$ recovers the matched input-conditioned backbone. ECA is an address *prior* in a shared state: same-entity reads and writes align, different entities land in near-orthogonal directions, and an idealized superposition model (Section 4) predicts crosstalk energy $O(M/N)$ in the state width N, the load axis of our capacity grid (Figure 2).

This paper contributes:

1. We identify cross-entity interference as an evaluation axis distinct from sequence length and introduce controlled benchmarks, with paired interventions on the security-grammar replication, that isolate entity-specific retrieval (Sections 5, 7.1; Appendix F).
2. We propose ECA, a scan-compatible structural prior that injects a stable entity key into recurrent write and read addresses. ECA-Hash keys are seed-derived, with no trainable table, and extend to unseen identifiers (Section 3).
3. We show that the identity-placement effect transfers across official Mamba-1/2/3, Naju, Gated DeltaNet, and an attention control, characterize its capacity scaling, and show a real-data gain on endpoint telemetry along with the regimes where it does not help (Tables 2 and 3; Figure 2 and Appendix F for the capacity grid; Section 8).

Security telemetry motivates the work, but the bottleneck lies in fixed-size sequence modeling.

2 Related Work

Recall, state tracking, and delta-rule memories. Associative recall organizes efficient-sequence-model research: Zoology (Arora et al., 2023) separates architectures by multi-query recall and selective SSMs (Mamba (Gu & Dao, 2023), Mamba-2 (Dao & Gu, 2024)) close much of the gap. We isolate a different axis—not *whether* a fact survives distance, but whether *many interleaved writers* stay separate in one fixed state; a model can be perfect at the former and floor at the latter. Theory bounds what constant-depth SSMs can track (Merrill et al., 2024), but our failures are finite-budget *learning* failures (of optimization or generalization) rather than expressivity limits: a structural prior that adds no capacity repairs them. Key-read matrix memories date to fast weights (Schmidhuber, 1992), revived as linear attention (Katharopoulos et al., 2020) and delta-rule updates (Schlag et al., 2021; Yang et al., 2024; 2025); these supply exactly the same-key-overwrite mechanics the security-grammar task rewards, which sharpens the finding: Gated DeltaNet with the id as *input* still floors (0.34; Table 3), while the same architecture with the key in its q/k solves it—the bottleneck is the key *source*, not the update rule.

Binding and explicit memories. Random-vector binding is classical—tensor products (Smolensky, 1990), holographic representations (Plate, 1995), hyperdimensional computing (Kanerva, 2009); ECA keeps the recurrence's outer-product write and *additively biases its addresses*; the binding primitive is classical, and our contribution is showing that *where* the same identity code enters—input or addresses—decides whether existing recurrent models learn to track many interleaved entities. Slot memories (NTM (Graves et al., 2014), DNC (Graves et al., 2016), EntNet (Henaff et al., 2017), key–value nets (Miller et al., 2016)) hold a fixed slot count with soft addressing; entity-tracking restructures *inputs* around mentions (Gupta & Durrett, 2019); order-p subspace carving (Pence et al., 2026) and per-entity/graph memories (Entity-GRU, TGN-style Kairos (Cheng et al., 2024)) allocate $O(M)$ state with no crosstalk, and HOLA (Cui, 2026) pairs a delta-rule state with a bounded exact KV cache that keeps selected associations *outside* the state; ECA instead supplies an external identifier as an *address prior* inside the $O(1)$ state, trading a crosstalk floor for constant state. Security detectors from flow classifiers to provenance graphs model short windows or engineered features; none we surveyed keys the recurrent state to a semantic entity (Appendix G).

3 Entity-Conditioned Addressing

Recurrent memory interface. Let S_t denote the recurrent state, v_t the content to be written, and k_t, q_t the write and read addresses of a fixed-size recurrent memory,

$$S_t = \mathcal{U}_\theta(S_{t-1}, v_t, k_t, g_t), \qquad y_t = \mathcal{R}_\theta(S_t, q_t, v_t), \tag{1}$$

where g_t collects the backbone's update controls (gates, decay, step sizes). ECA modifies only the addresses, using the event's entity identifier z_t (an integer id assigned during preprocessing):

$$e_t = \phi(z_t; s), \qquad \widetilde{k}_t = \mathcal{I}_k(k_t, \lambda_k e_t), \qquad \widetilde{q}_t = \mathcal{I}_q(q_t, \lambda_q e_t). \tag{2}$$

Here ϕ and its public seed s are defined in Eq. 3; the injection maps $\mathcal{I}$ specify addition and any normalization the backbone requires, and recover the baseline addresses at zero injection. ECA does not add an entity-indexed state bank: it preserves the state shape and recurrence family but changes the per-token update through the modified address inputs. For delta-rule models, changing the write key also changes the erase term. Appendix D lists the concrete k_t/q_t correspondence and injection site for each evaluated backbone, and Figure 1 summarizes the data flow.

Algorithm (streaming step). One ECA-Hash step at time t (stock computation elsewhere):

1: $\mathbf{e}_t \leftarrow \phi(z_t; s) \in \{\pm 1/\sqrt{N}\}^N$ ▷ Philox Rademacher; cached lookup; pad id $\mapsto \mathbf{0}$
2: $k_t, q_t, v_t, g_t \leftarrow$ backbone projections of h_t
3: $\widetilde{k}_t \leftarrow \mathcal{I}_k(k_t, \lambda_k \mathbf{e}_t)$; $\widetilde{q}_t \leftarrow \mathcal{I}_q(q_t, \lambda_q \mathbf{e}_t)$ ▷ per-backbone site (App. D)
4: $S_t \leftarrow \mathcal{U}_\theta(S_{t-1}, v_t, \widetilde{k}_t, g_t)$; $y_t \leftarrow \mathcal{R}_\theta(S_t, \widetilde{q}_t, v_t)$ ▷ stock update/readout

On the controlled benchmarks the scale is fixed per backbone from pilot sweeps (λ=4 at B/C sites with N=64, a $\lambda \propto \sqrt{N}$ heuristic; λ=1 at GDN's post-norm q/k). Validation grids include zero where a λ=0 arm is trained.

Instantiation in selective SSMs. For selective SSMs, the input-dependent B_t and C_t vectors act as write and read coordinates over the recurrent state dimension. ECA replaces them with

$$\widetilde{B}_t = B_t + \lambda_B \, \mathbf{e}(z_t), \qquad \widetilde{C}_t = C_t + \lambda_C \, \mathbf{e}(z_t),$$

without changing the state shape or scan dependency; two events land on overlapping state coordinates unless their addresses separate them, and without injection nothing ties an address to the entity an event is about. Appendix D gives the Naju update equation and the address correspondence and injection site for every evaluated backbone.

Why address injection helps. The write address $B_t \in \mathbb{R}^N$ determines how much of an event's write lands on each state coordinate, and the read address C_t which coordinates a query reads; events with similar addresses share coordinates and interfere. Without ECA these addresses are learned functions of the layer input (B_t=$W_B h_t$ with W_B a learned projection), which carries the identity code in the entity-input controls, so the model must *learn* to make same-entity events collide and

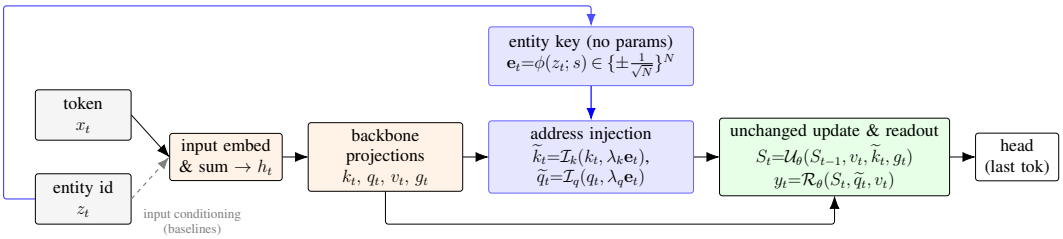

Figure 1: Entity-Conditioned Addressing. Matched baselines provide the entity code through the input path (gray dashed); ECA additionally injects the same stable code into the model's write and read addresses (blue). State shape, streaming execution, and the underlying update-rule family are unchanged.

different entities separate. ECA makes part of the address a function of the known entity identifier: same-entity events receive an aligned address component, whereas independent entity keys have zero expected inner product. ECA therefore increases same-entity signal relative to cross-entity crosstalk: in trained Naju checkpoints the written address has cosine 0.93–0.95 with its own entity key versus -0.01 with other entities' keys (Appendix F.2), without partitioning the state (Appendix I).

Main method: hash-derived keys. **ECA-Hash** derives the key deterministically from the identifier with a fixed public seed s,

$$\mathbf{e}_t = \phi(z_t; s) \in \left\{-\tfrac{1}{\sqrt{N}}, +\tfrac{1}{\sqrt{N}}\right\}^N, \qquad \mathbb{E}\left[\mathbf{e}_a^\top \mathbf{e}_b\right] = 0, \quad \mathrm{Var}\left(\mathbf{e}_a^\top \mathbf{e}_b\right) = \tfrac{1}{N} \quad (a \neq b), \tag{3}$$

where ϕ is a Philox (Salmon et al., 2011) Rademacher draw keyed by (s, z_t); the padding id maps to the zero key. These moments hold for distinct z_t; raw ids mapped to one z_t share its key (App. D). Each z_t keeps one address prior across sequences, unseen entities need no retraining, and key generation adds no trainable parameters.

Learned keys (learned-geometry ablation). **ECA-Learned** materializes one trainable key row per allocated entity ($E \in \mathbb{R}^{|\mathcal{E}|\times N}$). Unseen identifiers can receive newly initialized rows, but doing so requires an allocation and storage policy, and the result depends strongly on initialization geometry (Appendix F). We therefore use it as a learned-geometry ablation; ECA-Hash is the deployment-oriented variant because it derives equivalent persistent random geometry from a seed, with no trainable per-entity table.

Relation to associative memories. Fast-weight, linear-attention, and delta-rule recurrences (Schmidhuber, 1992; Katharopoulos et al., 2020; Yang et al., 2024; 2025) derive their keys from content; ECA supplies them from a known identifier, placing a classical binding primitive inside the per-timestep addresses of a scan-compatible recurrence. We separately ablate identity input, address injection, key construction, and address scale (Appendices B, F).

4 An Explanatory Crosstalk Model

ECA stores every entity in one state bank by *superposition*. We analyze an idealized memory whose keys are independent of the values and decay; learned address and gate functions need not satisfy this, so the analysis gives reference predictions for the full model.

Idealized superposition memory. Suppose M active entities have been written into a state $S = \sum_{i=1}^{M} v_i\, e_i^\top$, where $e_i \in \mathbb{R}^N$ is entity i's address key and v_i its current value vector. Reading entity q by its key gives

$$S e_q \;=\; v_q\,(e_q^\top e_q) \;+\; \sum_{i\neq q} v_i\,(e_i^\top e_q) \;=\; v_q \;+\; \underbrace{\sum_{i\neq q} v_i\,(e_i^\top e_q)}_{\text{crosstalk}}, \tag{4}$$

for unit-norm keys. The second term is cross-entity crosstalk: every other entity leaks into the read in proportion to its key's overlap with e_q.

Expected interference under random keys. For independent Rademacher keys $e_i \in \{\pm 1/\sqrt{N}\}^N$ (our hash construction) and values that do not depend on the keys, taking the expectation *over the keys* makes every cross term vanish exactly, and

$$\mathbb{E}_e \left\| S e_q - v_q \right\|^2 = \frac{1}{N} \sum_{i \neq q} \|v_i\|^2 = O\Big(\frac{M}{N}\Big) \quad \text{for uniformly bounded values,} \tag{5}$$

i.e. crosstalk amplitude $O(\sqrt{M/N})$, with no assumption on correlations *among* the values: *average* interference grows linearly in M/N (doubling active entities costs as much as halving the state width). A pairwise worst-case bound (max overlap $O(\sqrt{\log M/N})$ by sub-Gaussian concentration) and its caveats are in Appendix H; we stress-test high-overlap key pairs directly in a key-collision ablation (Appendix F) rather than argue from the tail. This classical random-superposition result predicts that interference grows with M at fixed N, shrinks with N at fixed M, and becomes visible as $M/N \to 1$ with no universal threshold at $M{=}N$—trends Fig. 2 bears out along both axes. Static-vs-learned caveats, the λ-scaling heuristic, and the remaining predictions are in Appendix H. ECA thus occupies a different operating point from explicit per-entity memory (no crosstalk, $O(M)$ state): constant-state multiplexed memory trading a crosstalk floor for $O(1)$ scaling in M.

5 Controlled Entity-State Benchmark

Our primary benchmark, **es-SET**, contains M interleaved entities, each holding one of $K{=}8$ mutable values. Target and distractor entities follow the same update-count distribution ($\max(1, \text{Poisson}(4))$) and use content-identical update tokens; only the entity identifier distinguishes which state is updated. A query selects one active entity uniformly and asks for its latest value (balanced labels, chance 0.125; the empirical no-identity floor sits above chance). We vary entity load M independently of sequence length. Additional generator families extend the same event structure to multi-attribute, compositional-counter, and lifecycle variants (Section 7.1).

On the security-grammar replication (Section 7.2) we additionally construct paired query-swap and distractor-perturbation interventions (results in Appendix F). For both benchmarks an independent replay oracle derives labels, and on the security grammar it also verifies that perturbations preserve the intended target state; the es-SET generator, the four-way security-grammar task suite with its mechanism-specific error metrics, and the shortcut and leakage controls are specified in Appendix C.

6 Experimental Setup

Notation. M is the *total* number of concurrently tracked entities (target plus distractors), N the key/address dimension of one memory head (equal N need not mean equal total state; Table A3), and T the stream length in events (security-grammar lengths are in tokens, four per event).

Models. Table 1 lists the primary-benchmark models (versions, state sizes, and injection sites in Appendix D). Within a backbone, input-only and ECA arms receive matched identity inputs; the ECA-Hash keys use no per-entity table and add one cached key lookup and two vector additions per step (Table A16). The security-grammar replication (Section 7.2) hosts its mechanism ablations on Naju (Lim & Yoon, 2026) (settings: Appendix C).

7 Results

7.1 Primary controlled comparison: identity placement on es-SET

We first report the study's primary comparison on es-SET (Section 5; $T{=}512$ events, M = total tracked entities).

Table 1: Primary-benchmark (es-SET) models; hash-ID-input and ECA-Hash parameters.

Backbone	depth	width	params
Mamba-2 (official)	4	256	1.84M
Gated DeltaNet	4	128	1.10M
Transformer (and +ID-equality bias)	4	256	3.78M
keyed delta memory	1	128	0.06M
per-entity GRU dictionary	1	128	0.14M

Table 2: **Identity placement under matched codes:** eight-way entity-state tracking (es-SET, chance 0.125) across two recurrent backbones. Within each backbone, the ECA-Hash arm and the hash-ID input arm share the input encoder and differ only in the address injection. Mean±sd; five seeds (Mamba-2), three (Gated DeltaNet, GDN).

Backbone	arm	M=4	M=16	M=64
Mamba-2	no ID	.324 ± .018	.166 ± .012	.130 ± .007
	learned-ID input	.322 ± .022	.143 ± .016	.122 ± .008
	hash-ID input	.332 ± .007	.175 ± .013	.130 ± .006
	concat-MLP input	.385 ± .099	.175 ± .018	.128 ± .005
	FiLM input	.408 ± .106	.201 ± .036	.131 ± .008
	per-layer hash input	.349 ± .058	.189 ± .009	.126 ± .002
	ECA-Hash	**.968 ± .008**	**.865 ± .034**	**.527 ± .008**
GDN	no ID	.334 ± .012	.197 ± .006	.134 ± .008
	learned-ID input	.329 ± .009	.192 ± .006	.128 ± .002
	hash-ID input	.336 ± .005	.197 ± .006	.135 ± .004
	concat-MLP input	.793 ± .133	.203 ± .006	.134 ± .002
	FiLM input	.471 ± .211	.199 ± .004	.133 ± .007
	ECA-Hash	**.999 ± .001**	**.977 ± .002**	**.706 ± .011**

Table 2 is the paper's central measurement. Every input arm—learned table, hash code through a projection, concat-MLP, FiLM, and per-layer re-injection (Mamba-2 only)—sits at or near the no-identity floor for $M \geq 16$ on both backbones. The only above-floor input readings are at the easiest load (M=4) and carry large seed variance (FiLM .408 ± .106 on Mamba-2; concat-MLP .793 ± .133 on GDN): unstable partial fits rather than a routing program that holds as entities accumulate. The matched ECA-Hash and hash-input arms differ only in the additional address injection within each backbone, so the per-seed paired deltas isolate the effect of injection site: +.637±.013/ + .690±.036/ + .397±.012 on Mamba-2 and +.663/ + .780/ + .571 on GDN. At M=N=64 keyed test accuracy falls (.527/.706) although final-epoch training accuracy stays ≥.96: a load-dependent generalization gap at the load where the crosstalk model predicts interference to become visible.

References and controls. These references and controls position ECA against alternatives (Table A1, Appendix B). A full-context Transformer given the same hash-code input fits small training sets without generalizing and stays near chance as M grows (.166/.130/.125), whereas a separate Transformer with a learned ID-equality attention bias reaches .999/.999/.997, and removing that bias alone from the same model drops it to chance (.131/.125 at M=16/64): identity as a routing prior also succeeds in attention. A dedicated keyed delta-rule memory using the same hash keys is stronger than the full backbones on single-attribute tracking (1.00/.99/.77) but not on the real ACME probe (.932 at 0.10M vs. .936–.984 for 1.3–3.3M keyed backbones), and an explicit per-entity dictionary with $O(M)$ state is exact at every load. Key-consistency controls isolate the mechanism: resampling keys per event or mismatching write and read keys removes the effect (−.70/−.72 paired), whereas a fixed permutation of the key-to-entity assignment does not (−.019 ± .041). The placement gap reproduces on the multi-attribute (.436 → .749) and lifecycle (+.535 paired) generator families, and at reduced magnitude on compositional counters (.234 → .441), which no arm solves.

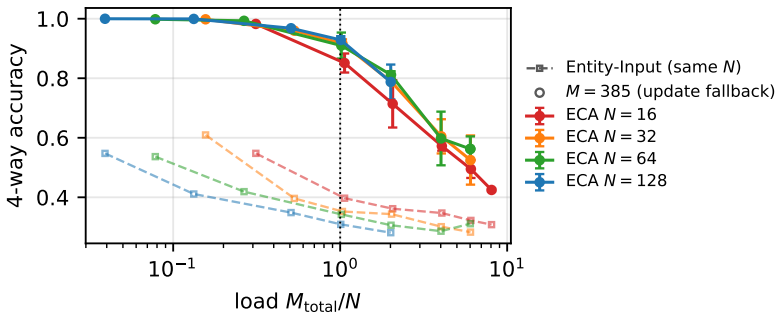

Figure 2: Four-way accuracy of ECA-Hash on the Naju host, security-grammar benchmark, as a function of load M/N, for $M \in \{5, 17, 33, 65, 97, 129, 193, 257, 385\}$ and $N \in \{16, 32, 64, 128\}$ (2048 tokens, $\lambda = 4\sqrt{N/64}$, three seeds per ECA cell; 23 cells spanning $M/N = 0.04$–8). The per-lifecycle update cap is 1 throughout, with mean target update counts of 1.12–1.14 for $M \leq 257$; at $M=385$ the event-budget fallback also lowers the lifecycle probability, so that cell (open marker) is a stress test rather than an update-controlled point. Appendix F details it and a companion sweep with the default update cap (Figure A1). Dashed lines are the entity-input controls at the same N (three seeds).

Table 3: Portability across recurrent backbones on the security-grammar benchmark ($M=17$, 1024 tokens). Within each backbone, the entity-input and ECA arms use the same model, data, and training budget; ECA additionally injects the deterministic entity key into the model's write/read addresses. Mean±sd over five seeds for the Mamba and Naju variants and three seeds for Gated DeltaNet. Implementation details (versions, widths, injection sites, size-matching) in Appendix A.

Backbone	params	no entity	entity input	ECA keyed	Δ vs input
Mamba-1 (official)	2.2M	.348 ± .012	.356 ± .013	**.813 ± .025**	+.46
Mamba-2 (official)	1.8–1.9M	.350 ± .008	.348 ± .009	**.932 ± .011**	+.58
Mamba-3 (official)	2.0–2.1M	.348 ± .011	.354 ± .004	**.857 ± .048**	+.50
Naju	0.7M	.351 ± .015	.364 ± .012	**.951 ± .006**	+.59
Gated DeltaNet	1.1M	.337 ± .010	.339 ± .006	**.956 ± .004**	+.62

7.2 Independent replication on a four-way security grammar

We replicate the placement effect on an independently generated controlled benchmark, separate from es-SET: a security event grammar with four-way current-state labels (chance 0.25) in which a target and $M-1$ distractors receive content-identical privilege updates (Appendix C; length alone does not expose interference, Appendix F.1, Table A10).

At $M=17$ (Table 3), identity as input stays at the floor on every backbone (.339–.364), while routing it through the backbone's own addresses reaches .813–.956. Figure 2 shows accuracy rising with N at fixed M and falling with M at fixed N across the update-controlled cells ($M \leq 257$), organized approximately by M/N (.71–.81 at $M/N \approx 2$, .50–.53 at 6), while the entity-input controls stay at .28–.61; $M=385$ is a separate budget-limited stress cell. With the generator's default update cap, which falls with M, the two largest N instead rebound at $M=129$ (Appendix F). Key scale matters (official Mamba-1 at $N=64$: .358 at $\lambda=1$, .813 at $\lambda=4$; Appendix A). Interventions, ablations, key-collision stress, and cost are in Appendix F.

8 Real-Data Evaluation

8.1 Wintap ACME 4: endpoint telemetry (interference-bound)

The public Wintap ACME 4 capture (Lawrence Livermore National Laboratory, 2026) (five weeks; host and process-instance counts in Appendix E, Table A7) is merged into one chronological per-host stream with the process instance as the entity; processes first observed on or after 2024-09-10 form the test split (split and leakage audit in Appendix E).

Table 4: Real corpora and hypothesis status. *No-entity arms at chance; input identity suffices.

Dataset	Task property	Hypothesis status	ECA outcome
ACME probe	routing-bound	confirmatory	positive (+21 to +26pt)
CTU-13	content/recency	confirmatory boundary	no clear gain; keys reduce recall
GCT	identity sufficient*	exploratory diagnosis	no consistent gain

Table 5: ACME live/terminated state probe with a nominal 16-event gap (exceptions and filtered evaluation: Appendix E). Temporal split by target first-event date; three seeds; balanced probes (chance 0.5). Within each row the entity-input and ECA-Hash arms are the same model with identical training settings (for the official Mamba blocks and Gated DeltaNet, the keyed model at λ=0). ECA-Hash uses λ=4 for Naju ($\lambda \propto \sqrt{N}$ heuristic) and for the official Mamba blocks (selected on validation PR-AUC from $\{0, 1, 4\}$), and post-norm λ=1 for Gated DeltaNet. No-entity arms of the official Mamba blocks are the library blocks (Mamba-2 at N=64 and head dimension 32, width-matched rather than at its library defaults); library-default input arms are in Appendix E.

Backbone	no entity	entity input	ECA-Hash
Mamba-1 (official)	0.720 ± .003	0.726 ± .003	**0.936 ± .011**
Mamba-2 (official)	0.713 ± .006	0.722 ± .001	**0.984 ± .000**
Mamba-3 (official)	0.723 ± .003	0.716 ± .003	**0.973 ± .004**
Naju	0.710 ± .006	0.714 ± .010	**0.975 ± .001**
Gated DeltaNet	0.713 ± .006	0.719 ± .002	**0.979 ± .002**

State probe (interference-bound). To test the interference-bound regime directly, we construct an entity-state probe on processes with an observed `PROC_END`: we query a process's live/terminated state at least 16 events after its last prior event, or after `PROC_END` for terminated probes. Trailing registry events break that gap in 11% of terminated probes; excluding those probes lowers seed-1 accuracy by .001 for ECA and .012 for entity input, slightly widening ECA's advantage (Appendix E). Labels come from `PROC_END`, never from red-team, and are balanced within host before the temporal split, so the pooled held-out split is approximately balanced (47,522 probes, 51.1/48.9; Table 5). Entity identifiers supplied as input leave every backbone near the no-entity floor (0.714–0.745 across all input arms). Against the same model at λ=0, injecting the same identifiers as address keys adds +0.210 ± .010, +0.262 ± .001, and +0.257 ± .006 on official Mamba-1/2/3 (mean ± s.d. of the paired difference over three seeds; per-seed values in Appendix E). On Naju, whose size-matched input arm is a separate model, the keys add 27 points (pooled-probe accuracy difference for one checkpoint pair, host-cluster bootstrap: ECA−input = +0.267, 95% CI [0.260, 0.276]), and on Gated DeltaNet +0.260 ± .003. These within-model comparisons, identical in size and training settings, attribute the gain to address injection rather than to the availability of the identifier.

Interference gap. Resampling probes with a larger minimum event gap before the query (no events are inserted; elapsed time also grows) lowers accuracy (Appendix E, Table A8): on Naju, models without address injection fall toward chance (0.77 → 0.64) while ECA stays near ceiling (0.99 → 0.93), so the accuracy gap widens (+22 → +29 points). A host-held-out re-split over two disjoint machines gives Naju ECA 0.975 ± .003 vs. input 0.722 ± .008 (Appendix E).

Attack-process detection. On ACME process-level attack detection, entity addressing provides no consistent benefit (Naju; PR-AUC, mean over three seeds: no entity 0.47, entity input 0.43, and ECA with λ=4 0.43). Reducing the key scale to λ=1 raises the observed ECA mean to 0.49, but all differences are small relative to the substantial seed-to-seed variation. A plausible explanation is that the query targets the most recent process and is therefore largely content- and recency-decodable (further detection numbers in Appendix E).

8.2 CTU-13: NETWORK FLOWS (A BOUNDARY CASE)

The standard CTU-13 (García et al., 2014) flow windows carry no per-entity channel, so we build an entity edition of the identical scenario-held-out windows (entity = destination endpoint; tokens, labels, and splits unchanged). Entity information provides no clear additional benefit (Naju: entity

input 0.942 vs. no entity 0.932 accuracy; botnet recall $0.731 \pm .171$ vs. $0.667 \pm .296$, with substantial seed variation). All arms are effectively indistinguishable on the in-distribution validation split—the chronologically last 10% of windows within each training scenario—where every arm saturates (0.998–1.000, and within each backbone the arms differ by at most 0.001), so its numerical argmax selects keyed configurations that reduce botnet recall on the scenario-held-out test set for all three evaluated backbones (Naju $0.731 \rightarrow 0.551$; Mamba-2 $0.845 \rightarrow 0.491$; Mamba-1 $0.797 \rightarrow 0.759$). In a post-hoc diagnostic that instead withholds whole scenarios for validation, the arms become distinguishable: λ=4 ranks last for both Mamba-2 and Naju, validation selects the entity-input arm on both, and the validation ranking agrees with the test-recall ranking within each backbone. This suggests that address scale must be selected on a validation split that reflects the anticipated deployment shift (Appendix E.1).

8.3 GCT: THE GOOGLE 2011 CLUSTER TRACE (NON-SECURITY)

On the Google 2011 cluster trace (Reiss et al., 2011) we probe a task's scheduler state under at least 16 intervening other-task events. No-entity arms sit at chance, so identity is necessary; entity-input arms on four backbones already reach .82–.84, overlapping the keyed arms' .80–.86 with no consistent keyed-versus-input advantage. This is the boundary the routing account predicts: ECA helps when identity is available but the backbone fails to turn it into a reliable routing rule, whereas on GCT the backbone already routes identity from the input. A keyed model at λ=0, sharing the keyed arms' batch size (the input arms use 64), stays at that level ($.832/.819/.828$), which rules out the batch-size difference as the explanation. On the Naju host the ACME–GCT contrast also survives the alternative identity encodings we tested: with persistent hash ids the GCT arms still overlap ($.801$ vs. $.782$), while with window-local bucket ids the ACME probe retains its $+.26$ gap (Appendix E). Within those encodings the boundary tracks the task's routing requirement rather than the identity encoding.

9 LIMITATIONS

Scope: routing-bound tasks. On tasks where recency and content suffice (attack-process detection, CTU-13 flow windows) or identity input alone suffices (the GCT probes; Table 4), ECA is neutral to harmful (on CTU-13, keyed configurations reduce botnet recall by up to 53 points, and by 18 at the scale that in-distribution validation actually selects on Naju, which detected neither loss); the mechanism helps consistently across backbones only when persistent entities interleave and the query must retrieve a specific entity's state from under more-recent unrelated events. Our principal real-data improvement is on an entity-state probe built from real process lifecycles rather than on operational attack detection. **Requires reliable entity identifiers.** ECA consumes parser-provided ids; noisy resolution (merges, splits, reuse) degrades the prior and needs explicit robustness measurement. **No isolation guarantee.** Keys can collide, retention gates act globally, and additive writes do not implement exact overwrite (Appendix I). **Scale and implementation scope.** Evidence comes from synthetic probes and three real corpora at small model scale (under 5M parameters); the host-held-out test split spans only two machines, and the keyed official Mamba and Gated DeltaNet layers run non-fused reference paths.

10 CONCLUSION

Fixed-size sequence models can retain information over long horizons yet fail when many entities update similar mutable state in one shared memory. On the controlled benchmarks, an entity identifier supplied as input leaves every tested recurrent backbone near the no-identity floor as entity load grows; supplied by ECA to the recurrent write and read addresses, the same identifier substantially improves entity-specific retrieval across five SSM and delta-rule backbones, while keeping state size independent of the number of entities. Real-data results show the same benefit on an interference-bound endpoint-state probe, but no consistent gain when identity or recent content alone suffices, and measurable harm on content-dominated detection. One account covers both: what changes is where the identifier enters, not whether it is available. ECA is therefore a structural prior for entity-multiplexed recurrence, for tasks in which a query must retrieve one entity's state from under interleaved updates by others.

Reproducibility statement

The mechanism is fully specified in Section 3 (deterministic Philox Rademacher keys; additive address injection; setting $\lambda{=}0$ recovers the matched input-conditioned backbone without address injection, to floating-point precision). Benchmark generators, configurations, and seeds are described in Appendices D, C, and E, and anonymized code and dataset-generation scripts accompany the submission. Training runs used single NVIDIA RTX 4090, RTX 5090, and H200 GPUs; the throughput and state-size measurements in Table A16 were all taken on one RTX 4090.

AI use statement

We used generative AI tools for coding and writing assistance, including implementation scaffolding, refactoring, feedback on experimental design and claim calibration, result interpretation, and LaTeX/prose editing. All AI-assisted code and analyses were independently reviewed and verified by the authors; all reported experiments were run by the authors and all numerical claims were checked against the resulting artifacts. We take responsibility for the final content of this work, including all text, claims, and artifacts.

References

Abdulellah Alsaheel, Yuhong Nan, Shiqing Ma, Le Yu, Gregory Walkup, Z. Berkay Celik, Xiangyu Zhang, and Dongyan Xu. ATLAS: A sequence-based learning approach for attack investigation. In *USENIX Security*, 2021.

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. Zoology: Measuring and improving recall in efficient language models. *arXiv preprint arXiv:2312.04927*, 2023.

Le Cheng, Peican Zhu, Yangming Guo, Chao Gao, Zhen Wang, and Keke Tang. SourceDetMamba: A graph-aware state space model for source detection in sequential hypergraphs. In *IJCAI*, pp. 2749–2757, 2025. arXiv:2505.12910.

Zijun Cheng, Qiujian Lv, Jinyuan Liang, Yan Wang, Degang Sun, Thomas Pasquier, and Xueyuan Han. Kairos: Practical intrusion detection and investigation using whole-system provenance. In *IEEE S&P*, 2024. arXiv:2308.05034.

Andrew Chu, Xi Jiang, Shinan Liu, Arjun Nitin Bhagoji, Francesco Bronzino, Paul Schmitt, and Nick Feamster. Feasibility of state space models for network traffic generation. In *ACM SIGCOMM Workshop on Networks for AI Computing (NAIC)*, pp. 9–17, 2024. arXiv:2406.02784.

Wanyun Cui. A hippocampus for linear attention: An exact memory for what the recurrent state forgets. *arXiv preprint arXiv:2607.02303*, 2026.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *ICML*, 2024. arXiv:2405.21060.

Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. DeepLog: Anomaly detection and diagnosis from system logs through deep learning. In *ACM CCS*, 2017.

Mohamed Elbasheer and Adewale Akinfaderin. User-based sequential modeling with transformer encoders for insider threat detection. *arXiv:2506.23446*, 2025.

Sebastián García, Martin Grill, Jan Stiborek, and Alejandro Zunino. An empirical comparison of botnet detection methods. *Computers & Security*, 45:100–123, 2014.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv:1410.5401*, 2014.

Alex Graves, Greg Wayne, Malcolm Reynolds, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476, 2016.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv:2312.00752*, 2023.

Haixuan Guo, Shuhan Yuan, and Xintao Wu. LogBERT: Log anomaly detection via bert. In *IJCNN*, 2021. arXiv:2103.04475.

Aditya Gupta and Greg Durrett. Effective use of transformer networks for entity tracking. In *Proceedings of EMNLP-IJCNLP*, 2019.

Xueyuan Han, Thomas Pasquier, Adam Bates, James Mickens, and Margo Seltzer. Unicorn: Runtime provenance-based detector for advanced persistent threats. In *NDSS*, 2020. arXiv:2001.01525.

Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. Tracking the world state with recurrent entity networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Md Nahid Hossain, Sadegh M. Milajerdi, Junao Wang, Birhanu Eshete, Rigel Gjomemo, R. Sekar, Scott D. Stoller, and V. N. Venkatakrishnan. SLEUTH: Real-time attack scenario reconstruction from COTS audit data. In *USENIX Security*, 2017.

Pentti Kanerva. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive Computation*, 1(2):139–159, 2009.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *ICML*, 2020. arXiv:2006.16236.

Kaichuan Kong, Dongjie Liu, Xiaobo Jin, Zhiying Li, and Guanggang Geng. Log2Sig: Frequency-aware insider threat detection via multivariate behavioral signal decomposition. *arXiv:2508.05696*, 2025a.

Kaichuan Kong, Dongjie Liu, Xiaobo Jin, Zhiying Li, Guanggang Geng, and Jian Weng. MambaITD: An efficient cross-modal mamba network for insider threat detection. *arXiv:2508.05695*, 2025b.

Gayan K. Kulatilleke, Siamak Layeghy, Mahsa Baktashmotlagh, and Marius Portmann. MambaNetBurst: Direct byte-level network traffic classification without tokenization or pretraining. *arXiv:2605.11034*, 2026.

Aakash Lahoti, Kevin Y. Li, Berlin Chen, Caitlin Wang, Aviv Bick, J. Zico Kolter, Tri Dao, and Albert Gu. Mamba-3: Improved sequence modeling using state space principles. In *International Conference on Learning Representations*, 2026. arXiv:2603.15569.

Lawrence Livermore National Laboratory. Wintap: An extensible host-based agent for Windows, and the ACME datasets. https://github.com/llnl/Wintap, 2026. Datasets at https://gdo-wintap.llnl.gov/; accessed September 2026.

Hyuk Lim and Seunghyun Yoon. Naju: A native discrete state-space model with independent retention and writing for long-sequence memory, 2026. URL https://arxiv.org/abs/2607.21000.

Xinjie Lin, Gang Xiong, Gaopeng Gou, Zhen Li, Junzheng Shi, and Jing Yu. ET-BERT: A contextualized datagram representation with pre-training transformers for encrypted traffic classification. In *The Web Conference (WWW)*, 2022. arXiv:2202.06335.

Qiqi Liu, Runhan Song, Lei Cui, Heng Zhang, Yuyan Sun, and Limin Sun. MIDS: Detecting stealthy masquerade and tampering attacks on CAN bus via bidirectional Mamba. *arXiv:2606.18599*, 2026.

Liam Daly Manocchio, Siamak Layeghy, Wai Weng Lo, Gayan K. Kulatilleke, Mohanad Sarhan, and Marius Portmann. FlowTransformer: A transformer framework for flow-based network intrusion detection systems. *Expert Systems with Applications*, 241, 2024. arXiv:2304.14746.

Xuying Meng, Chungang Lin, Yequan Wang, and Yujun Zhang. NetGPT: Generative pretrained transformer for network traffic. *arXiv:2304.09513*, 2023.

William Merrill, Jackson Petty, and Ashish Sabharwal. The illusion of state in state-space models. In *International Conference on Machine Learning (ICML)*, 2024.

Sadegh M. Milajerdi, Rigel Gjomemo, Birhanu Eshete, R. Sekar, and V. N. Venkatakrishnan. HOLMES: Real-time APT detection through correlation of suspicious information flows. In *IEEE S&P*, 2019.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents. In *EMNLP*, 2016. arXiv:1606.03126.

Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. Kitsune: An ensemble of autoencoders for online network intrusion detection. In *NDSS*, 2018.

Travis Pence, Daisuke Yamada, and Vikas Singh. Recursive binding on a budget: Subspace carving in order-p tensor memories. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2606.11391.

Tony A. Plate. Holographic reduced representations. *IEEE Transactions on Neural Networks*, 6(3): 623–641, 1995.

Charles Reiss, John Wilkes, and Joseph L. Hellerstein. Google cluster-usage traces: format + schema. Technical report, Google Inc., 2011. Revised 2014; https://github.com/google/cluster-data.

John K. Salmon, Mark A. Moraes, Ron O. Dror, and David E. Shaw. Parallel random numbers: as easy as 1, 2, 3. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, 2011.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *ICML*, 2021. arXiv:2102.11174.

Jürgen Schmidhuber. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139, 1992.

Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *ICISSP*, 2018.

Paul Smolensky. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*, 46(1–2):159–216, 1990.

Aaron Tuor, Ryan Baerwolf, Nicolas Knowles, Brian Hutchinson, Nicole Nichols, and Robert Jasper. Recurrent neural network language models for open vocabulary event-level cyber anomaly detection. *AAAI Workshops*, 2018. arXiv:1712.00557.

Su Wang, Zhiliang Wang, Tao Zhou, Hongbin Sun, Xia Yin, Dongqi Han, Han Zhang, Xingang Shi, and Jiahai Yang. ThreaTrace: Detecting and tracing host-based threats in node level through provenance graph learning. *IEEE TIFS*, 2022. arXiv:2111.04333.

Tongze Wang, Xiaohui Xie, Wenduo Wang, Chuyi Wang, Youjian Zhao, and Yong Cui. NetMamba: Efficient network traffic classification via pre-training unidirectional mamba. In *IEEE ICNP*, 2024. arXiv:2405.11449.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. In *NeurIPS*, 2024. arXiv:2406.06484.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. In *ICLR*, 2025. arXiv:2412.06464.

Ruijie Zhao, Mingwei Zhan, Xianwen Deng, Yanhao Wang, Yijun Wang, Guan Gui, and Zhi Xue. Yet another traffic classifier: A masked autoencoder based traffic transformer with multi-level flow representation. In *AAAI*, 2023.

Table A1: References and diagnostics (es-SET, three seeds, unless the row states otherwise): a full-context Transformer with the same hash-code input; a separately implemented causal Transformer of the same depth and width with one learned ID-equality attention-bias scalar per layer (the hash-input Transformer differs in implementation: it is bidirectional, uses batch 16 rather than 8, and has attention-projection biases and FFN-internal dropout; the matched contrast is the same bias model with the bias removed, which falls to .131 ± .008 at M=16 and .125 ± .002 at M=64); a dedicated keyed delta-rule memory using the same hash keys (0.06M); and an explicit per-entity GRU dictionary ($O(M)$ state). †es-SET, learning rate $\leq 3{\cdot}10^{-4}$, one seed per learning rate.

	M=4	M=16	M=64
Transformer + hash-ID input	.166 ± .009	.130 ± .010	.125 ± .002
+ ID-equality attention bias	**.999 ± .001**	**.999 ± .001**	**.997 ± .000**
keyed delta memory (es-SET)	**1.000 ± .000**	**.992 ± .001**	**.774 ± .001**
per-entity dictionary ($O(M)$ state)	1.000 ± .000	1.000 ± .000	1.000 ± .000
keyed delta memory, multi-attribute (M=16)		.454 ± .004	
keyed delta memory, ACME probe (0.10M)		.932 ± .000	
Transformer, M=2 fit: train / test†		.99–1.00 / .45–.50	

A Portability implementation details

All Table 3 arms share the identical training budget (30 epochs, batch 32, lr $3{\cdot}10^{-4}$) and byte-identical dataset cache. "Entity input" supplies ids as input embeddings (matching the security-suite entity-input arm); "ECA keyed" adds the same deterministic hash key to each backbone's own write/read addresses: per-timestep B_t/C_t for the SSMs (λ=4 by the $\lambda \propto \sqrt{N}$ heuristic at N=64), per-head q/k for Gated DeltaNet (λ=1, post-norm site; pre-norm gives 0.952±.007). Mamba-3 rotates B/C by data-dependent rotary angles, under which a static key written at time s and read at time t would be rotated by their relative angle sum; we therefore inject into the non-rotary half only (official rope fraction 0.5: the last 64 of its N=128 dims), post B/C-norm, with the same N=64 key geometry and λ=4. Mamba-1/2/3 are the official `mamba_ssm` 2.3.2.post1 blocks and GDN the `fla` 0.5.1 layer, wrapped unmodified for the Mamba baseline arms and the GDN no-entity arm (the GDN entity-input arm is the keyed model at λ=0, see below); the keyed arms re-implement only the block's own non-fused forward around its own kernel; at λ=0 they are verified to recover the matched input-conditioned block without address injection, to floating-point precision. Widths: d_{model}=256 (Mamba-1/2/3), 128 (Naju, GDN); N=64 everywhere except Mamba-3's official default N=128 (rotary fraction 0.5); widening Mamba-1's N alone from the default 16 yields no clear change without keys (.353±.004 at N=16 vs. .348±.012 at N=64). Key scale matters: with everything else identical, keyed Mamba-1 at N=64 stays at the floor with λ=1 (.358±.007, three seeds) versus .813±.025 at λ=4 (five seeds). Naju's params reflect its gate-compressed host form (full-rank gates would be 1.1M at the same width, with no clear accuracy difference on the entity-blind baselines we compared). The GDN arms are size-matched at the keyed configuration (head_dim 64, 1.1M): the entity-input arm is the keyed model at λ=0 (byte-identical code path), so within the row only the key injection varies; the 4× larger `fla`-default baseline also floors (.335±.010). The keyed host remains the smallest model in the table.

B Reference models, key-consistency controls, and generator-family coverage

The address prior must be consistent, not any particular code. Three key-consistency controls (two negative, one invariance) on Mamba-2 ECA-Hash (es-SET, M=16, keyed .865 ± .034) separate *consistent addressing* from the specific key values. Drawing a fresh random key per event (**resample**) collapses accuracy to .161 ± .014 (−.704 ± .045 paired); using different write and read keys for the same entity (**mismatch**) gives .145 ± .012 (−.720 ± .042); but a fixed random *permutation* of the key-to-entity assignment (**permute**), which preserves per-entity consistency and write/read alignment, is not clearly different from ECA-Hash (.845 ± .038, −.019 ± .041). The mechanism needs a stable, aligned per-entity address—not the particular hash the seed happens to produce.

Family coverage on the same generator confirms the placement gap is not specific to set-membership. It persists on multi-attribute tracking (Mamba-2 hash-ID input $.436 \pm .001$ vs. ECA-Hash $.749 \pm .100$) and on compositional counter updates at reduced magnitude (hash-ID input $.234 \pm .007$ vs. ECA-Hash $.441 \pm .030$; neither solves the family). It also reproduces on lifecycle tracking with the full paired contrast: ECA-Hash reaches $.852 \pm .023$ vs. hash-ID input $.317 \pm .003$ ($+.535 \pm .025$ paired). Across es-SET, the lifecycle family, and the multi-attribute family, the address prior is the deciding factor wherever the family is learnable at all.

Training fit versus test accuracy. The es-SET logs record final-epoch online training accuracy (dropout active, averaged over that epoch's updates; not an evaluation-mode score of the selected checkpoint). At $M{=}16$, the Mamba-2 arms without address injection reach .21–.41 in training (learned-ID input $.68{\pm}.09$) and .14–.20 on test (five seeds), and every GDN arm without it, including no-ID, reaches .73–1.00 in training but .19–.20 on test (three seeds). ECA-Hash reaches $.998 \pm .001$ in training and $.865 \pm .034$ on test on Mamba-2, and 1.000 and $.977 \pm .002$ on GDN. Input arms thus either fit poorly or fit without generalizing; since the no-ID GDN arm also fits ($.79 \pm .09$), training fit alone does not indicate identity-based routing. At $M{=}64$, ECA-Hash still reaches $.978 \pm .013$ (Mamba-2) and 1.000 (GDN) in training but $.527 \pm .008$ and $.706 \pm .011$ on test, so its high-load drop occurs with the training set largely fit.

C Benchmark specification: event grammar, security-grammar task suite, and metrics

Both benchmarks isolate a *generic* computational problem: many persistent entities interleave in one stream, each carries mutable K-way state, and a query must recover a specific entity's current (or retained) state from under more-recent unrelated activity. es-SET (Section 5) uses neutral tokens; the four-way security-grammar suite specified below instantiates the same structure with a security event grammar (principals, privilege levels, attack-seed evidence), the setting where the problem occurs at scale and entity ids are reliably available.

es-SET generator. Each stream contains M entities with $K{=}8$ mutable values over $T{=}512$ events. Per-entity update counts are drawn as $\max(1, \mathrm{Poisson}(4))$ for targets and distractors alike, update tokens are content-identical across entities, and the query selects one active entity uniformly (balanced labels). Raw entity ids are drawn from a pool of 2^{20} and mapped to codes $1, \ldots, 4095$ (first 32 bits of the MD5 digest, modulo 4095, plus one); active sets are redrawn until their codes are distinct, which excludes exact code aliases but not high overlap between distinct keys (cf. T4-C). The multi-attribute, compositional-counter, and lifecycle families reuse the same event structure. An independent replay oracle re-derives every label from the frozen caches.

es-SET itself is close to uniform (largest test class 0.130), but the derived families are not, because a queried attribute that was never set returns a default value and lifecycle adds a deleted class: the constant-predictor rates are 0.437 (multi-attribute), 0.318 (lifecycle) and 0.240 (compositional counters). Arms on those families should be read against those rates, not against $1/K$. In the lifecycle family the per-entity create, update and delete events are interleaved by a global shuffle, so an entity's create can land after its own updates or after its delete; labels are taken from the replay oracle applied to the realized stream rather than from the intended per-entity schedule, which is why its default class is common.

Training protocol. Every run uses AdamW with learning rate 3×10^{-4}, weight decay 10^{-4}, a cosine schedule with 5% warmup, gradient clipping 1.0, dropout 0.1, and bfloat16 autocast; the reported checkpoint is the one with the best validation accuracy, except on the ACME tasks where validation PR-AUC is the selection metric. Only the epoch budget and batch size vary by benchmark: es-SET 20 epochs at batch 16 (8,000/1,000/2,000 train/validation/test streams of 1,539 tokens for $M \in \{4, 16, 64\}$ and the derived families); the security-grammar suite and the portability table 30 epochs at batch 32; the T1–T3 baseline grid and CTU-13 50 epochs at batch 32; the ACME and GCT probes 20 epochs, at batch 64 for the Naju arms, 16 for the official Mamba blocks on ACME, and 32 for Gated DeltaNet and for the keyed official blocks on GCT, whose entity-input and no-entity arms use 64. Seeds are 1–3 unless a table states five.

Event encoding. Each host timeline is a sequence of structured events `[ENTITY_TYPE ACTION OBJECT VALUE]` over a self-contained vocabulary (process/user/file/session entity types; connect/access/set-privilege/ login/logout/delete/create actions; benign object pool; attack-seed and privilege value tokens). A detection request is appended as an explicit query token triple, so the label is a function of the whole history rather than of the last event. T1–T3 instances are trained at sequence length 512 and evaluated with length extrapolation to 1024 and 2048 (longer benign gaps / more overwrites). The T2/T4 interference grid trains at length 1024 (2048 for the capacity grid).

T1–T3 — retention, tracking, and their composition. **T1** (long-horizon retention): an attack-seed event emitted in the first quarter is buried under benign events; the query asks for it (chance 0.25). The answer is fixed at emission, so long-length accuracy measures cue survival; T1 doubles as a *non-degradation* check—ECA is unnecessary here and must not hurt. **T2** (mutable entity-state tracking): a target and $M-1$ distractor entities receive *content-identical* `SETPRIV` updates—only the entity channel distinguishes them—and the query asks the target's current four-way state (three privileges or deleted, with a delete/re-create lifecycle axis). M is varied *independently of sequence length*, making interference a controlled variable; $\mathbf{T2}_0$ denotes the single-writer special case (recency suffices, no binding required), which is what the length-extrapolation grid (Table A10) runs, while $M>1$ ($M-1$ competing writers) adds interference. **T3** (joint decision): a 2×2 label over (early seed severity) $\times$ (current privilege); both the far-away seed and the latest state are required (chance 0.25; measured single-axis ceiling 0.53).

T4 — cross-entity interference stress tests. T4 is designed to *isolate* interference as the binding constraint (paired interventions: Table A13). **T4-A (same content, different entity)** is realized inside T2: distractor updates are token-identical to target updates, so content-only addressing cannot separate them, and sweeping M traces the interference curve. **T4-B (adversarial interleaving)** inserts, immediately *after* the target's last update, a burst of L_b distractor updates whose values are drawn independently of the target's label (constraining them to *differ* from the label would leak the answer by complement—the value absent from the burst—which an entity-blind model can exploit); the *recency-pick rate* (predicting the most recent distractor's value) quantifies the recency pull. **T4-D (unseen entities)** splits the entity-identifier pool so that validation/test identifiers are disjoint from training identifiers, testing hash-key generalization against learned per-entity key tables. **T4-C (key collision)** pairs entities whose hash keys are unusually similar, constructed from the public key generator (results in Appendix F).

Interference metrics (operational definitions). Let y be the label, $\hat{y}$ the prediction, and, per test instance, let $\mathcal{H}$ be the set of values the *target* held earlier (its history), $\mathcal{O}$ the set of *other* entities' current states (including deleted), and r the value of the last distractor update before the query. To avoid counting coincidences, the label is removed from both sets ($\mathcal{H}, \mathcal{O}\setminus\{y\}$), mirroring the stale metric's $\text{prev} \neq y$ condition. We define (Tables A10 and A11 report (i), (iii), and (iv)): (i) *immediate-previous stale rate*: $\Pr[\hat{y} = \text{prev}]$ over instances whose state genuinely changed ($\text{prev} \neq y$); (ii) *any-historical stale rate*: $\Pr[\hat{y} \in \mathcal{H}]$ over instances with $\mathcal{H} \neq \emptyset$; (iii) *cross-entity selection rate*: $\Pr[\hat{y} \in \mathcal{O}]$ over instances with $\mathcal{O} \neq \emptyset$—a *descriptive proxy* for answering with some other entity's state (at large M the set $\mathcal{O}$ covers most wrong classes, so this rate approaches the plain error rate; see the caveat at Table A11); (iv) *recency-pick rate*: $\Pr[\hat{y} = r]$ over instances with $r \neq y$—the rate of copying the most recent value regardless of entity, the shortcut T4-B is built to tempt; (v) *deleted-entity resurrection rate*: among deleted targets, $\Pr[\hat{y} \in \mathcal{H}]$ (reporting a pre-deletion value). Accuracy says *whether* a model fails; these say *how*: (iii) and (iv) are the two failure modes an entity-address prior specifically targets.

Shortcut and leakage controls. Attack seeds and state updates share the same event grammar as benign events (no marker tokens), so order must be tracked. T1's answer is independent of the tail (defeating recency); T3's single-axis ceilings are measured, not assumed; T2/T4 entity identifiers are sampled per sequence from a pool, so an identifier is not predictive of a label. Distractor timelines are *content-matched* to the target's—the same update grammar, the same update-count distribution, and the same lifecycle probability—so neither event types nor counts identify the target, and the no-identity floor is verified empirically with an entity-blind model. Labels are not uniform, however: the deleted class covers about 12% of instances and the other three about 29% each, so the constant-

predictor rate is 0.31 rather than the 0.25 of a uniform four-way task. The measured entity-blind floors (0.33–0.36) sit above that rate, so they are not a majority-class artifact.

Security-grammar suite: arms and training settings. Naju's gate parameterization does not touch the address path ECA modifies (rank-32 and full-rank gates sit at the same no-identity floor, 0.351–0.353), so we report a single Naju backbone. The mechanism arms: **A0** no entity information, **A1** entity-input, **A2/A3** write-/read-only keys, **A4 ECA-Hash** (main), and the ECA-Learned key-source ablation (Table A15); same-code injection-site controls **B1** (hash key to input only) and **B2** (input + addresses); all entity-aware arms receive identical input information, with λ=4 (the $\lambda \propto \sqrt{N}$ heuristic at N=64) unless a λ sweep is shown. Portability arms inject the same keys into official Mamba-1/2/3 and Gated DeltaNet's per-head q/k (Table 3), and a per-entity GRU dictionary reference completes the matrix. T1–T3: 50 epochs, three seeds (five for official Mamba-1/2/3), $512 \to 1024/2048$ extrapolation; the T2/T4 grid trains at length 1024 with 30-epoch screening and sweeps M.

Engineering note: generator correctness gate. Before any training run, the generators pass an independent replay-oracle gate that re-derives every label from the frozen caches. Active sets are redrawn until their codes are distinct; T4-C varies the overlap between keys of distinct codes and does not test exact code aliases.

D BACKBONE EQUATIONS FOR THE EVALUATED INJECTIONS

Correspondence and injection sites. Table A2 lists the address correspondence and injection site for each evaluated backbone.

Table A2: Address correspondence and injection site for each evaluated backbone.

Backbone	k_t/q_t correspondence	implementation notes
Mamba-1	selective B_t/C_t	slow path; N=64, $\lambda \propto \sqrt{N}$
Mamba-2	B_t/C_t address vectors	non-fused path; group broadcast
Mamba-3	non-rotary half of B_t/C_t	rotary half carries relative phase
Naju	B_t/C_t	host used for the ablation matrix
Gated DeltaNet	delta write key / read query	post-norm site (pre-norm ablated); the kernel re-applies its ℓ_2 normalization after the addition

Naju host. The native-discrete state-space recurrence maintains a state $X_n \in \mathbb{R}^{d_{\text{inner}} \times N}$ updated as

$$X_n = f_n[:,\text{None}] \odot X_{n-1} + (i_n \odot h_n)\, B_n^\top, \qquad y_n = \alpha\, X_n C_n + D \odot h_n, \tag{6}$$

with a fixed readout scale α=$1/\sqrt{N}$, a per-channel retain gate f_n, write gate i_n, content branch h_n, and write/read address vectors $B_n, C_n \in \mathbb{R}^N$. The gates are computed from h_n as $f_n = \sigma\big(W_f^{(2)} W_f^{(1)} h_n + \text{conv}_f(h)_n + b_f\big)$ and likewise for i_n, using the backbone's low-rank gate option with rank r=32 for each gate (a depthwise causal convolution of width 4; b_f=+5, b_i=−2 at initialization). All Naju results use this configuration (0.71M parameters without keys at d=128); full-rank gates would give 1.1M. ECA injects into B_n, C_n as in the main text. Mamba-1 injects into the selective B_t/C_t (slow path), Mamba-2 into the SSD B_t/C_t with group broadcast (non-fused path), Mamba-3 into the non-rotary coordinate block of B_t/C_t (the rotary half carries relative phase), and Gated DeltaNet adds the key to its per-head q/k after normalization (pre-norm ablated).

Ablation axes. The injection exposes three switches that we ablate independently (Appendix F): the key source (the main ECA-Hash method versus the materialized-table ECA-Learned ablation), the injection site (write-only λ_C=0, read-only λ_B=0, or both), and the scale λ (fixed per backbone from pilot sweeps on the controlled benchmarks, with a sweep in Table A11; λ=0 recovers the matched input-conditioned backbone without address injection—verified to floating-point precision for the Naju host and the official Mamba blocks; for Gated DeltaNet the λ=0 and keyed arms are the same module with the addition disabled). The added cost is one key lookup and two vector additions

Table A3: Full primary-benchmark model configurations (es-SET). "Inject" is the ECA site for the keyed arm of each backbone; input-only references have none. Parameter counts are from the training runs.

Backbone	impl./version	depth	d_{model}	state/head	params	inject (seeds)
Mamba-2	`mamba_ssm` 2.3.2.post1	4	256	N=64, d_h=32, 1 grp.	1.84M	B_t/C_t (5)
Gated DeltaNet	`fla` 0.5.1	4	128	d_h=64, 4 heads	1.10M	post-norm q/k (3)
Transformer	PyTorch, bidir.	4	256	4 heads	3.78M	— (3)
+ID-eq. bias	custom, causal	4	256	4 heads	3.78M	attn. logits (3)
keyed delta mem.	dedicated	1	128	N=64	0.06M	unit-key δ (3)
Entity-GRU dict.	per-entity GRU	1	128	$O(M)$ state	0.14M	— (3)

per layer and step; the recurrence and its chunk-parallel associative scan are untouched. The keys are not generated per step: each keyed layer holds a non-trainable float32 buffer with the keys of all H model-facing codes, regenerated from the seed at construction and excluded from checkpoints (arms with a hash-code input encoder hold one more at the input). It takes $4HN$ bytes per layer; at N=64 that is 1 MiB for H=4096 (es-SET, ACME, GCT; 4 MiB over four layers) and ≈64 KiB for H=257 (T2/T4 security grammar, CTU-13; H=513 in the expanded capacity-grid cells). It grows with H, not with the number of raw entities (trainable *key* parameters: none for ECA-Hash; $|\mathcal{E}| \cdot N$ per layer for ECA-Learned). The codes come from preprocessing: es-SET and ACME hash raw ids into 4095 nonzero codes (32-bit MD5 and BLAKE2b digests, respectively, modulo 4095, plus one; es-SET redraws any active set whose codes collide), CTU-13 hashes destination addresses into 256 nonzero codes (MD5), the T2/T4 security grammar draws tracked entities' codes without replacement from $\{1, \ldots, 256\}$, expanded to $\{1, \ldots, 512\}$ for the capacity-grid cells at M=257 and 385 (benign filler events reuse codes 2–65; T1–T3 always give the queried target code 1), and GCT numbers tasks by first appearance within each window. Distinct raw ids that share a code share its key, and GCT codes do not persist across windows. For fairness we distinguish four identity representations: a learned per-entity *input* embedding table, entity-*type*/event-type embeddings, the deterministic hash code, and a learned per-entity *key* table. Within each arm family the input-side representation is matched—the entity-aware security-suite A-arms all receive the same id/event input embeddings, and the primary ECA-Hash arm shares the hash-input arm's encoder—so any gap is attributable to the address injection, not to extra information. The ECA-Hash *keys* use no per-entity table in either family.

E Detailed real-data tables

Supporting results for Sections 8.1, 8.2, and 8.3.

ACME corpus scope. Modelling uses the 12 hosts of the capture that carry at least 40,000 process instances; smaller hosts are dropped before any probe is built. The leakage audit in Table A7 is computed over all 31 hosts of the raw capture, and the host-held-out re-split holds out two of the twelve modelled hosts.

ACME split audit. Test probes share no process instance with the training and validation pools, and 0.06% of test context events precede the 2024-09-10 cut (0.5% of training context events follow it). For the results in Table 5 the validation pool is a random tenth of the training probes and is therefore not target-process-disjoint: 75% of its target processes also occur among training targets; it is used only for checkpoint and λ selection. Retraining all fifteen arms from scratch on a target-process-disjoint split, which moves every probe of a validation target out of the training set and leaves the test set unchanged, reproduces the table: fourteen arms change test accuracy by at most 0.008; the fifteenth, official Mamba-3 with entity input, improves by 0.025. Repeating the $\lambda \in \{0, 1, 4\}$ selection under this split again selects λ=4 on all three official Mamba blocks. In 11% of terminated probes the target logs trailing registry events up to about 2 s after `PROC_END`, so fewer than g events separate its last event from the query (never for alive probes). Excluding these probes moves seed-1 test accuracy from .706 to .694 (entity input) and from .973 to .972 (ECA), so they favour the input arm. In 10% of probes another process in the window shares the target's hash code, equally often for both labels.

Table A4: GCT: probing a task's scheduler state on the Google 2011 cluster trace under $\geq$16 intervening other-task events (three seeds, mean$\pm$sd). The single-event variant is decodable from one distinctive event (chance 0.333); the mutable-transition variant forces pending$\leftrightarrow$running tracking over content-identical SCHEDULE/EVICT/SUBMIT events (chance 0.500). Every no-entity arm sits at exact chance; transition-variant entity-input and ECA arms overlap (.82–.84 vs. .80–.86) with no consistent direction. Official-block pairs differ in batch size (input 64, keyed 32) and, on Mamba-1, in state size.

Probe / backbone	no entity	entity input	ECA-Hash
single-event (Naju)	.333	.746	.748
transition, Mamba-1 (official)	$.500 \pm .000$	$.842 \pm .005$	$.855 \pm .012$
transition, Mamba-2 (official)	$.500 \pm .000$	$.824 \pm .006$	$.804 \pm .008$
transition, Mamba-3 (official)	$.500 \pm .000$	$.830 \pm .003$	$.820 \pm .010$
transition, Naju	.500	$.820 \pm .019$	$.821 \pm .019$

ACME official Mamba blocks and Gated DeltaNet controls. Under the probe-level validation split used for Table 5, each keyed official Mamba block is trained at $\lambda \in \{0, 1, 4\}$ with an otherwise identical configuration (three seeds each; batch 16; Mamba-1 at N=64). For λ=0/1/4, Mamba-1 (3.25M): validation PR-AUC 0.803/0.806/0.981 and test accuracy $0.726 \pm .003$/$0.733 \pm .008$/$0.936 \pm .011$; Mamba-2 (2.94M): validation PR-AUC 0.796/0.991/0.997 and test accuracy $0.722 \pm .001$/$0.955 \pm .008$/$0.984 \pm .000$; Mamba-3 (3.11M): validation PR-AUC 0.795/0.802/0.995 and test accuracy $0.716 \pm .003$/$0.732 \pm .021$/$0.973 \pm .004$. Validation PR-AUC is the checkpoint-selection metric and selects λ=4 for all three. Per-seed paired gains (λ=4 minus λ=0, seeds 1/2/3): Mamba-1 $+.210/+.200/+.219$ ($+0.210 \pm .010$), Mamba-2 $+.261/+.261/+.264$ ($+0.262 \pm .001$), Mamba-3 $+.257/+.262/+.250$ ($+0.257 \pm .006$). The keyed Mamba-1 runs used a second host with `mamba_ssm` 2.2.5, whose `Mamba` constructor defaults match 2.3.2.post1. Mamba-1 and Mamba-3 run at their library defaults; the Mamba-2 block is width-matched at N=64 and head dimension 32. The library-default input arms (batch 64) give $0.730 \pm .001$ (Mamba-1, library default N=16), $0.720 \pm .003$ (Mamba-2), and $0.745 \pm .042$ (Mamba-3). For Gated DeltaNet, the entity-input arm is the keyed post-norm model at λ=0 (1.66M parameters, identical code path; paired gain $+0.260 \pm .003$, per seed $+.263/+.257/+.260$), the no-entity arm is the same configuration without identity inputs (1.13M), and the same learned entity input on the library-default Gated DeltaNet layer (fla default head dimension; 4.85M) gives $0.725 \pm .003$.

ACME attack-process detection (full). On **attack-process detection** (is this process red-team, queried at its last event), entity addressing provides no consistent gain. A0, entity input, ECA with λ=4, and ECA with λ=1 obtain PR-AUC 0.470 ± 0.089, 0.434 ± 0.053, 0.426 ± 0.073, and 0.487 ± 0.120, respectively (mean $\pm$ s.d., n=3); the differences are within seed variation. A plausible explanation is that the query targets the most recent process and is therefore largely content- and recency-decodable (official Mamba-1 trails at 0.28, consistent with the ceiling being the content/recency signal rather than architecture). Lowering the key scale from λ=4 to 1 increases the observed ECA mean, consistent with the hypothesis that a strong key can crowd out content information needed for this task. The no-entity model is also smaller, with 0.73M parameters versus 1.26M for the entity-input and ECA models. Task construction: the split uses the probe's criterion (test targets are processes first observed on or after 2024-09-10 UTC; context windows are not cut at that date), training negatives are subsampled to six per positive (7,309 windows, 1,056 positive), validation is 1,000 random windows of the training pool (131 positive), and the test split keeps every positive with capped negatives (150,000 windows, 2,501 positive, i.e. 1.7% prevalence), which is why PR-AUC rather than accuracy is reported.

GCT: FULL RESULTS (GOOGLE 2011 CLUSTER TRACE)

The GCT probe extends the evaluation to real *non-security* telemetry: entity = scheduler task, with a lifecycle probe under enforced interference (Table A4). On the single-event variant, identity-as-input already suffices (0.746 vs. ECA 0.748). On the mutable-transition variant, identity is genuinely required (every no-entity arm sits at exact chance)—but under its window-local identity encoding, the cross-architecture controls show that input identity can suffice: input arms on all four backbones

Table A5: CTU-13 host-window botnet classification, scenario-held-out (three seeds): the ECA on/off boundary measurement, on an entity edition of the standard windows (entity = destination endpoint hashed into the fixed key space; tokens, labels, and splits byte-identical to the standard encoding). Entity *information* provides no clear additional benefit; the keyed arms reduce held-out botnet recall, dose- and backbone-dependently, although in-distribution validation selects a keyed configuration on every backbone (see below).

Arm	Accuracy	Botnet recall
A0 Naju (no entity info)	$0.932 \pm .059$	$0.667 \pm .296$
A1 Naju + entity input	$0.942 \pm .035$	$0.731 \pm .171$
A4 Naju + ECA-Hash, λ=1	$0.909 \pm .049$	$0.551 \pm .246$
A4 Naju + ECA-Hash, λ=4	$0.840 \pm .008$	$0.200 \pm .042$
Mamba-1 (official) + entity input	$0.959 \pm .028$	$0.797 \pm .137$
keyed Mamba-1 (official), λ=4	$0.950 \pm .029$	$0.759 \pm .141$
Mamba-2 (official) + entity input	$0.968 \pm .007$	$0.845 \pm .031$
keyed Mamba-2 (official), λ=4	$0.897 \pm .026$	$0.491 \pm .123$

reach .820–.842, overlapping the keyed arms' .804–.855. ECA has no consistent edge here on any backbone. Nor is the overlap an artefact of the batch-size difference: the keyed models at λ=0, which share the keyed arms' batch size, reach $.832/.819/.828$ on official Mamba-1/2/3—the input level. A gap sweep (min. gap 16/32/64) likewise shows *no* clean dose–response, unlike ACME. The instructive contrast is with the ACME probe, where the *same* four backbones' input arms stay pinned at the 0.71–0.75 floor: in these two corpora the GCT transitions behave as identity-sufficient and the ACME probe as routing-bound, and ECA's cross-architecture advantage appears only in the latter (a regime characterized after the GCT experiments). The corpora also differ in identity encoding (GCT: window-local first-appearance ids, ≤ 254 per window; ACME: 4095 persistent hash buckets), so we evaluated alternative identity encodings for both probes on the Naju host. With persistent hash ids the GCT arms still overlap (ECA $.801 \pm .027$ vs. entity input $.782 \pm .020$, three seeds). For ACME we renumber the existing hash buckets by first appearance inside each window, which preserves any pre-existing aliases; ECA then reaches $.980 \pm .002$ against $.719 \pm .010$ for entity input, the same $+.261$ gap as under persistent ids. The contrast therefore survives the recodings we tested on this backbone.

Across the three corpora. Where ECA on/off is directly measured, the advantage tracks whether the task is *routing-bound*: decisive on the ACME probe (content-identical mutable state under interference, $+$26pt in the three-seed means on Naju (27pt for the checkpoint pair used in the bootstrap), and $+21$ to $+$26pt on the official Mamba blocks and GDN against the same model at λ=0), without a consistent gain where identity information alone suffices (both GCT probes, where input arms match ECA's level), and absent-to-harmful where recency and content suffice (the ACME detection task and CTU-13 flow windows, where keyed configurations, including the validation-selected λ=1, cost botnet recall)—a pattern consistent with the synthetic benchmarks. CTU-13 contributes the orthogonal check that every backbone is healthy on real traffic.

Beyond this study. Two extensions are left to future work: an *operational* attack-onset evaluation (detection delay, recall at fixed false-alerts-per-host-day), which requires a deployment-style protocol rather than the mechanism-isolating probes used here; and additional security corpora (CERT insider-threat, LANL authentication) for breadth beyond the three real corpora covered here.

Host-held-out generalization. Re-splitting the probe so a disjoint set of machines is held out (no shared process instances or per-host idiosyncrasies) reproduces the dissociation at similar magnitude (three seeds, 24k probes): A0 $0.721 \pm .002$, A1 $0.722 \pm .008$ (within seed variation of the floor), ECA-Hash $\mathbf{0.975 \pm .003}$. For one checkpoint pair, the per-probe gain was $+0.251$ over A0 (bootstrap 95% CI $[0.247, 0.255]$), compared with -0.009 $[-0.009, -0.008]$ for input identity. The latter difference is within the three-seed variation of A1 ($.722 \pm .008$ vs. A0 $.721 \pm .002$); these host-cluster bootstrap intervals do not capture seed-to-seed uncertainty. The held-out test spans only two machines, and per-host effects are not separately tabulated; the effect is not an artifact of memorized per-host structure.

CTU-13 windowing and class balance. A window holds 127 consecutive flows of one host. Within each split, windows of a botnet host that contain no botnet flow are dropped as ambiguous and normal windows are subsampled to four times the botnet windows; training is additionally capped at 20,000 windows and validation at 2,000 (stratified), while the test split is uncapped. Final counts are 1,134 botnet / 4,536 normal for training, 327 / 1,308 for validation and 2,022 / 8,088 for test. Accuracy and botnet recall are therefore reported at an evaluated prevalence of 20%, not at the 3.8% prevalence of the raw held-out scenarios.

CTU-13 validation-based selection. Validation is the chronologically last 10% of windows within each training scenario; test windows come from held-out scenarios. Validation accuracy saturates for every arm, so accuracy-based selection cannot detect the scenario-shift harm (Table A6).

Table A6: CTU-13 validation-based selection: validation accuracy, test accuracy, and held-out-scenario botnet recall (means over three seeds).

Backbone / arm	val acc	test acc	test botnet recall
Mamba-1 (official), entity input	.9994	.959	.797
Mamba-1 (official), ECA-Hash λ=4	1.0000	.950	.759
Mamba-2 (official), entity input	.9982	.968	.845
Mamba-2 (official), ECA-Hash λ=4	.9988	.897	.491
Naju, entity input	.9992	.942	.731
Naju, ECA-Hash λ=1	1.0000	.909	.551
Naju, ECA-Hash λ=4	.9993	.840	.200

E.1 Scenario-held-out validation diagnostic on CTU-13

This diagnostic was designed after the test results in Table A5 were known. Complete scenarios are withheld for validation (train $\{3, 4, 5, 7, 10, 11\}$, validation $\{12, 13\}$, test $\{1, 2, 6, 8, 9\}$), so model selection meets the same kind of distribution shift as deployment. Because scenarios 12 and 13 are removed from training, absolute recall is not comparable with Table A5; only comparisons among arms *within* this protocol are meaningful. Mamba-2 contributes its entity-input arm, two keyed scales, and the keyed model at λ=0 (same code path and parameter count as its keyed arms); Naju contributes its entity-input arm and two keyed scales. Mamba-1 is not evaluated under this protocol.

Under this split the arms separate on validation (.848–.949; Table A9) instead of saturating, λ=4 ranks last on both backbones, and the validation ranking reproduces the test-recall ranking exactly within each backbone. Validation selects the entity-input arm for both backbones—that is, it rejects every keyed configuration here, which is the outcome one wants on a task where keys are harmful: deployment-aligned validation *can* expose a harmful address scale. With three seeds and botnet-recall standard deviations up to $\pm$.279, the ordering among the Mamba-2 entity-input, λ=0 and λ=1 arms in particular rests on differences far smaller than the seed spread.

F Mechanism and scaling diagnostics on the Naju ablation host

These experiments use the security-grammar task suite (T1–T4, Appendix C) and its arm naming (A0 no-ID, A1 entity input, A4 ECA-Hash; B1/B2 injection-site controls). They complement the primary placement comparison (Table 2) with length, capacity, intervention, and cost measurements. Naju is the designated ablation host; the length baselines (Table A10), the interference table (Table A11), and the cost table (Table A16) also include the official Mamba and Gated DeltaNet blocks, and the intervention diagnostics (Table A13) include official Mamba-1 and Mamba-2.

F.1 Baseline measurements: length extrapolation does not expose interference

Table A10 reports the completed T1–T3 baseline grid. Three observations shape the rest of the study. First, every model saturates in-distribution, so the tasks discriminate only under length extrapolation.

Table A7: Wintap ACME leakage audit: what the `unknown`-bucket exclusion removes. The excluded aggregation buckets are not process instances but absorb three quarters of red-team-*attributed* events; process-level labels (the task unit) are 99.86% retained. Event-level difficulty is therefore not assumed to be preserved. Counts cover all 31 hosts of the raw capture.

	before	excluded	after
hosts	31	0	31
process instances	1,773,405	56 buckets	1,773,349
red-team *process* labels	3,693	5	3,688 (99.86%)
attributed events (all tables)	17.53M	14.4%	15.0M
red-team-attributed *events*	1.75M	**74.6%**	0.44M

Table A8: The interference gap on real data (Naju; test accuracy, mean of three seeds). Each column regenerates the probes with minimum gap g: alive queries fall at least g events after the target's last prior event, terminated queries g–$2g$ events after its `PROC_END`. This is a selection filter, not inserted events: from g=8 to 64 the median terminated-query distance in the test probes grows from 12 to 96 events (9 to 60 distinct processes; 22 s to 460 s), alive queries stay far from the target's last event (median 555–681 events), and the probe set is resampled, so distinct-entity load is not isolated. The A4−A0 accuracy gap grows from +22 to +29 points.

min. gap	8	16	32	64
A0 no-entity	0.767	0.710	0.662	0.638
A1 entity-input	0.781	0.714	0.654	0.636
A4 ECA-Hash	**0.985**	**0.975**	**0.956**	**0.927**

Second, under 4× extrapolation the Transformer degrades on all three tasks (position generalization), Naju degrades on retention-bound tasks, and official Mamba-1, Mamba-3, and Gated DeltaNet show the same high-variance seed lottery on T1/T3 (0.80±.16/0.71±.22, 0.49±.24/0.57±.15, and 0.45±.34/0.66±.29—the newest Mamba does not inherit Mamba-2's length robustness, and the delta rule does not confer it either)—but **official Mamba-2 stays at ceiling on all three tasks** (1.00/0.97/0.97): with a single queried entity and benign distractors, the strongest selective SSM shows no clear interference problem at these lengths. Third, the stale-selection rate mirrors this: all three official Mamba blocks make near-zero stale errors at 2048 ($\approx$ 0.01) while the Transformer's stale rate (0.41) explains most of its $T2_0$ loss. The conclusion is methodological: *sequence length alone does not surface cross-entity interference*. The T2/T4 grid therefore varies the active-entity count M directly, with content-identical distractor updates.

F.2 Interference at M=17: routing, not information

Table A11 carries three findings. **(i) Information is not the bottleneck; routing is.** Supplying entity/event identifiers as input embeddings moves neither the Naju backbone (A1) nor Mamba off the entity-blind floor. This is not an expressivity gap: the addresses B_n, C_n are functions of an input stream that now contains the entity id, so an entity-dependent addressing solution exists in the hypothesis class. It is a learning gap (of optimization or generalization)—with content-identical distractor updates, standard training did not discover a well-separated, transferable routing geometry from input-level identity, and the selectivity that let Mamba saturate T1–T3 did not find one either. ECA supplies exactly that geometry as an explicit inductive bias. **(ii) With a stronger address prior, routing emerges earlier and generalizes better.** At λ=0.25 the routing solution is found only partially (0.69 ± .02); from there accuracy, cross-entity selection, and recency pull improve monotonically up to a ≈0.96 plateau at λ=8—on this task an 8× key did not reduce accuracy, plausibly because *values* ride the content stream h_n rather than the address (on content-decodable real tasks larger keys do hurt; Appendix E). In the per-epoch logs, validation accuracy first reaches 0.6 in epoch 1 for $\lambda \geq 2$, in epochs 2 and 4 for λ=1 and 0.5, and in epochs 12–14 for λ=0.25, while final online training accuracy is $\geq$ 0.93 at every λ: the small-λ deficit is mainly one of generalization; because rescaling learned weights absorbs λ exactly on this host (Appendix H), the benefit of a larger λ is a training effect. **(iii) Address injection suppresses the targeted interference errors.** Cross-entity selection falls by more than an order of magnitude (0.65→0.05 at λ=4) and the re-

Table A9: Post-hoc scenario-held-out validation diagnostic on CTU-13 (three seeds, mean ± s.d.). Whole scenarios $\{12, 13\}$ form the validation split, so absolute values are not comparable with Table A5; only within-protocol arm comparisons are valid. Bold marks the arm selected by validation accuracy within each backbone. Within each backbone the validation ordering and the test-recall ordering coincide, and λ=4 is last in both.

Backbone	Arm	Validation accuracy	Test botnet recall
Mamba-2 (official)	**entity input**	$\mathbf{.949 \pm .029}$	$\mathbf{.485 \pm .279}$
	keyed, λ=0	$.927 \pm .011$	$.343 \pm .140$
	ECA-Hash, λ=1	$.942 \pm .030$	$.423 \pm .209$
	ECA-Hash, λ=4	$.848 \pm .045$	$.120 \pm .096$
Naju	**entity input**	$\mathbf{.945 \pm .012}$	$\mathbf{.218 \pm .025}$
	ECA-Hash, λ=1	$.926 \pm .012$	$.124 \pm .026$
	ECA-Hash, λ=4	$.885 \pm .052$	$.089 \pm .012$

Table A10: T1–T3 baselines, three seeds (five for official Mamba-1/2/3; mean±sd), in-distribution (512) and 4×-length extrapolation (2048). T1: seed-retention accuracy; $T2_0$: current-privilege accuracy and stale-selection rate in the *single-writer* variant (no competing updates—no entity binding required; see Appendix C); T3: joint stage accuracy (single-axis ceiling 0.53 measured, chance 0.25). Mamba-1/2/3 are the official `mamba_ssm` blocks (five seeds each) and Gated DeltaNet the official `fla` layer (three seeds). All models saturate in-distribution; official Mamba-2 remains at ceiling even at 4× length, while Mamba-1, Mamba-3, and Gated DeltaNet degrade with high seed variance—length robustness is neither monotone in Mamba version nor conferred by delta-rule mechanics. Contrast with Table A11: the same Mamba-2 that scores 0.97 on $T2_0$ falls to the 0.35 floor once 16 competing writers (M=17) require entity binding.

	T1 retention		$T2_0$ state		T3 joint	
Model	acc_{512}	acc_{2048}	acc_{2048}	stale_{2048}	acc_{512}	acc_{2048}
Transformer	1.00	0.33±.13	0.51±.01	0.41±.01	0.99	0.64±.01
Mamba-1	1.00	0.80±.16	0.99±.01	0.01	1.00	0.71±.22
Mamba-2	1.00	**1.00±.00**	**0.97±.04**	**0.01**	1.00	**0.97±.04**
Mamba-3	1.00	0.49±.24	0.98±.03	0.01	1.00	0.57±.15
Naju	1.00	0.54±.22	0.89±.01	0.06	1.00	0.47±.04
Gated DeltaNet	1.00	0.45±.34	0.92±.04	0.04	1.00	0.66±.29

cency pull more than tenfold, exactly the failure modes the entity key targets. At λ=1 the M-sweep reads 0.97 (M=5), 0.88 (M=17), and 0.62 (M=65—65 active entities in a 64-dimensional address space, length 2048), while the input-only controls stay at the floor throughout (0.44/0.36/0.33): near-ceiling operation well below capacity, and graceful degradation at the onset of the $M \approx N$ boundary, which the full capacity grid below maps. On T4-B (a burst of 64 distractor updates inserted after the target's last event), ECA at λ=1 scores $0.86 \pm .04$ with a recency-pick rate of 0.06 (controls: 0.29–0.33 accuracy, recency-pick ≈0.30)—the adversarial burst does not pull the read off the target's address. At the plateau scale λ=4 every operating point improves further and stabilizes: the M-sweep reads 0.99/0.95/0.78 (M=5/17/65; five seeds at M=17: $0.951 \pm .006$ vs. five-seed floors 0.351/0.364), T4-B reaches 0.96, and unseen-entity generalization shows no clear loss (0.947 held-out vs. 0.951 seen, five seeds each)—the stronger prior even pushes back the $M \approx N$ capacity onset (0.62 → 0.78 at M=65).

The capacity grid. Figure 2 reports 22 update-controlled cells ($M \leq 257$) and one budget-limited stress cell at M=385. The per-lifecycle update cap is 1 throughout; for $M \leq 257$ the queried entity receives a mean of 1.12–1.14 writes. At M=385 the fallback that shrinks an instance until it fits the 510 event slots also scales the lifecycle probability down, which lowers that mean to 1.07 and changes the lifecycle distribution. We therefore mark that cell separately and keep it out of the update-controlled comparison. Within the update-controlled subset three readings matter. First, the predicted monotonic trends hold: accuracy rises with N at fixed M, and falls with M at fixed N (at N=64: 0.998/0.993/0.910/0.811/0.598 for M=5/17/65/129/257). Second, the M/N ratio

Table A11: Interference on T2 (M=17 concurrently tracked entities, length 1024, three seeds unless noted, 30-epoch screening): four-way current-state accuracy, wrong answers matching another entity's current state, and recency-pick rate. Entity identifiers as *input* do not beat the entity-blind floor (row 1) on any backbone —the official Mamba-1/2/3 blocks, Naju, a Transformer, or Gated DeltaNet; the same identifiers injected as ECA *address keys* solve the task, monotonically in the key scale λ. *Caveat*: at M=17, 89% of test samples have every wrong class present as some other entity's current state, so the second column is close to the overall error rate by construction; the paired intervention tests of Table A13 (query-swap, distractor perturbation) are the mechanism-bearing diagnostics.

Model	acc	cross-entity	recency-pick
A0 Naju host, no entity info	0.35 ± .02	0.65	0.49
A1 + entity input	0.36 ± .01	0.64	0.38
Mamba-1 (official) + entity input	0.35 ± .01	0.65	0.49
Mamba-2 (official) + entity input	0.34 ± .01	0.66	0.48
Mamba-3 (official) + entity input	0.35 ± .00	0.65	0.46
Transformer + entity input	0.33 ± .01	0.67	—
Gated DeltaNet + entity input	0.34 ± .01	0.66	0.54
A4 ECA-Hash, λ=0.25	0.69 ± .02	0.31 ± .02	0.15
A4 ECA-Hash, λ=0.5	0.75 ± .02	0.25	0.13
A4 ECA-Hash, λ=1	0.876 ± .027	0.12	0.06
A4 ECA-Hash, λ=2	0.921 ± .019	0.08	0.04
A4 ECA-Hash, λ=4 (5 seeds)	0.951 ± .006	0.05	0.03
A4 ECA-Hash, λ=8	**0.962 ± .002**	**0.04**	**0.02**
Entity-GRU dictionary (reference)	1.000 ± .000	0.00	0.00

organizes the data to first order: plotted against load, the cells form a single band falling from 1.000 (M/N=0.04) to 0.425 (M/N=8), and cells matched in load but built from different (M, N) agree to within 0.10: 0.71–0.81 at M/N≈2, 0.57–0.61 at 4, and 0.50–0.53 at 6. Third, the collapse is *approximate*: matched-load pairs agree closely at low load (0.960 at M/N=0.53 versus 0.968 at 0.51) but separate near capacity, where the narrowest state falls clearly below the rest while the three wider ones stay within 0.02 of one another (0.851/0.917/0.910/0.929 at $M/N \approx 1$ for N=16/32/64/128). Final-epoch online training accuracy, averaged over seeds, is ≥0.98 in every update-controlled cell (lowest single run 0.968), so test accuracy falls with load while the training set is still fitted closely. The stress cell reaches 0.862 in training against 0.562 on test, low enough that fitting limitations may contribute there. Identifiers are drawn from a pool of 256, expanded to 512 for the M=257 and 385 cells, which M itself forces. The static superposition analysis (Section 4) thus earns its role as a first-order guide—and its stated limits: the learned recurrence's gating and the $\lambda \propto \sqrt{N}$ coupling add width-dependent structure the idealization does not capture.

Consistent with that width-dependence, the key scale itself is a lever *at* the capacity edge: at the grid's hardest ECA-favorable point (M=129, N=64), sweeping λ from 2 to 4 to 8 moves accuracy 0.629 → 0.818 → 0.862 (three seeds each, unmatched caches)—a stronger prior buys back part of the capacity loss, echoing the λ-monotonicity observed at M=17. Meanwhile the entity-input controls show no systematic gain with N at fixed M (cell means across N: 0.537–0.610, non-monotone, at M=5; 0.396–0.419 at M=17; 0.343–0.349 at M=65; 0.301–0.309 at M=129; three seeds per cell, 23 cells): more state does not substitute for routing. The *length* axis likewise has little effect: holding M=17 and sweeping the sequence length 8× (512→1024→4096, three seeds each), ECA declines only gently (0.965±.001 → 0.951±.006 → 0.947±.011) while entity-input never leaves the floor (0.368/0.364/0.374)—over this range the horizon matters far less than entity load (the per-entity update cap is 8 at 1024–4096 tokens and 5 at 512), consistent with the benchmark's premise that scaling L alone neither creates nor cures the failure.

Update-count control. A companion sweep (Figure A1) uses the generator's default update budget, which varies with M: to fit the 510 event slots of a 2048-token instance, the per-entity update cap is $\min(8, \lfloor 510/M \rfloor - 2)$, i.e. 8/8/5/1 for M=5/17/65/129, so at M=129, 88% of targets re-

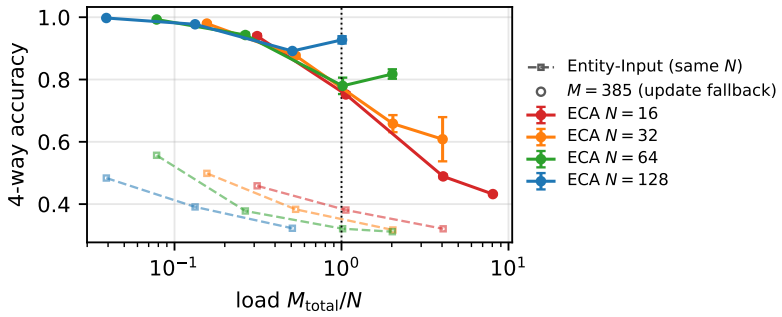

Figure A1: Unmatched capacity grid: the sweep of Figure 2 on caches whose per-entity update cap falls with M (8/8/5/1 for M=5/17/65/129). Accuracy falls through M=65 and reverses at M=129 for N=64 and N=128; matching the update count across M removes the reversal. Three seeds per ECA cell; dashed lines are the entity-input controls at the same N (two seeds, 13 of the 16 cells).

Table A12: Same-code injection-site control: the identical deterministic hash key, supplied to the input stream only (B1) or to the input and the addresses (B2); the code is held fixed and only the injection site (and, in the amplified row, the input-injection scale) varies. Five seeds, three for the amplified row. References: learned-embedding input A1 $0.364 \pm .012$, ECA-Hash A4 $0.951 \pm .006$.

Injection site (same hash code)	acc
B1 input only (learned projection)	$0.366 \pm .025$
B1 input only, amplified $16\times$	$0.688 \pm .029$
B2 input + address (λ=4)	$0.957 \pm .005$

ceive a single `SETPRIV` (mean 1.1 versus 3.3 at M=65) and an instance carries fewer keyed events ($\approx$193 versus 236). There, accuracy falls through M=65 but reverses at M=129 for the two largest N ($0.780 \to 0.818$ at N=64, $0.891 \to 0.927$ at N=128), by more than the seed variation, and cells matched in M/N but differing in M carry the same confound. Matching the update count across M removes the reversal ($0.910 \to 0.811$ and $0.968 \to 0.929$ in Figure 2) while leaving the M=129 cells, which already used a cap of 1, essentially unchanged ($0.818 \to 0.811$, $0.927 \to 0.929$): the reversal was a property of the generator's update budget, not of entity load.

Identity-grouping intervention. To vary the number of distinct identities without changing what is written or when, we relabel the 16 distractors of each M=17 test instance into k groups that share one identity; tokens, positions, event types, the target's timeline and the label are unchanged, and the trained checkpoints of Table A13 are evaluated as is. ECA-Hash accuracy falls as the same writes are concentrated on fewer identities: .941/.934/.905/.837/.705 for k=16/8/4/2/1 (three checkpoints), while the entity-blind, entity-input and hash-input arms stay at .35–.37 for every k. The number of distinct identities alone therefore does not set the difficulty; per-identity write load matters too (and k<16 lies outside the training distribution).

Same-code injection-site control. The comparison above still leaves one confound: A1 uses a *learned* input embedding while A4 adds a *fixed hash* to the addresses, so the gap could in principle reflect the identity representation rather than the injection site. We therefore hold the code itself fixed: the same deterministic Rademacher key (same width N, same Philox seed) is provided (B1) to the input stream only, through a learned projection $x_n \mathrel{+}= P\,\mathbf{e}_n$, or (B2) to the input *and* the addresses. With identical identity code, data, backbone, and budget (30 epochs, five seeds, M=17), the site alone decides the outcome (Table A12). At its natural scale the input-only variant is within seed variation of the learned-embedding baseline, and the address variant of the headline ECA. Amplifying the input-only injection $16\times$ recovers part of the signal (0.69, three seeds) but stays roughly 0.27 below the address variant under the same code, data, and budget: *where* the identity code enters the model dominates how strongly it is injected.

Table A13: Intervention diagnostics: query-swap (does the model respond to *which* entity is queried?) and distractor perturbation (does it falsely read distractor content?). Under the query swap, A1 is nearly insensitive; B1 exhibits seed-dependent sensitivity, but only a small fraction of its changes correct the prediction. Both input-injection variants are perturbation-sensitive; ECA responds to the query and ignores distractor resampling. The bottom block repeats the diagnostics on the *official* Mamba blocks (three trained checkpoints each): the same signature reproduces—official Mamba-1 with entity input shows only a small response to the query swap ($.025 \pm .009$), while the keyed official blocks respond (.86/.93).

	Query swap		Distractor perturb.	
Model	respond	both correct	flip	both correct
No entity info (A0, 1 seed)	0.000	0.000	0.536	0.195
Entity-Input (A1)	$.038 \pm .021$	$.005 \pm .004$	$.618 \pm .011$	$.169 \pm .005$
Hash-Input (B1)	$.156 \pm .179$	$.033 \pm .040$	$.563 \pm .066$	$.191 \pm .030$
ECA-Hash (A4)	$\mathbf{.952 \pm .006}$	$\mathbf{.889 \pm .011}$	$\mathbf{.078 \pm .011}$	$\mathbf{.909 \pm .011}$
Mamba-1 (official) + entity input	$.025 \pm .009$	$.003 \pm .001$	$.562 \pm .073$	$.181 \pm .018$
keyed Mamba-1 (official)	$.857 \pm .014$	$.661 \pm .034$	$.216 \pm .020$	$.725 \pm .022$
keyed Mamba-2 (official)	$\mathbf{.933 \pm .010}$	$\mathbf{.842 \pm .024}$	$\mathbf{.114 \pm .021}$	$\mathbf{.868 \pm .024}$

Table A14: One-sided keying (T2, M=17, length 1024, three seeds; each cell λ=1 / λ=4). Keying only the write address stays at the floor at either tested scale, while a strong read-only key partially recovers—the fixed key acts as a *target* that the learnable content-write path (which receives entity inputs) can align to, but not vice versa, consistent with alignment being the mechanism, with the write side the learnable one.

Ablation	acc (λ=1 / λ=4)
A2 write-only key	0.36 / 0.36
A3 read-only key	0.35 / 0.58
A4 ECA-Hash, write + read	0.88 / 0.95

Paired intervention diagnostics. Because the set-membership error metrics degenerate at large M (Table A11 caveat), we measure the binding mechanism directly with two paired tests on 2,000 pairs each. *Query swap*: two examples share an identical event stream (identical tokens and positions; only the query's entity id differs) with different correct answers—a model that actually reads the queried entity's state must change its prediction. *Distractor perturbation*: the target's timeline, the query, and the label are held fixed while the *values* of all distractor updates are resampled (label-independent; target label verified invariant by an oracle re-derivation)—a model that reads the target's state should not move. Derived pair members are not independent samples; n is the pair count, and each cell aggregates three independently trained checkpoints (seeds 1–3; the entity-blind A0 reference is one seed—it responds 0.000 by construction). Table A13 reports both diagnostics. The entity-blind reference never responds to the query (by construction it cannot). Under the query swap, A1 changes its prediction on fewer than 4% of pairs. B1 responds more frequently on average ($15.6\% \pm 17.9\%$), but this rate is highly seed-dependent ($6.9\%/36.1\%/3.7\%$) and is driven primarily by one seed; moreover, B1 answers both queries correctly on only 3% of pairs. Thus, A1 is nearly insensitive to the queried entity, whereas B1 exhibits unstable rather than reliably corrective sensitivity. Resampling unrelated entities' values flips both on ≈60% of pairs. ECA answers both queries correctly on 89% of query-swap pairs, is correct on both members of 91% of perturbation pairs, and keeps its prediction on 92% of them—it is reading *the queried entity's* state, not a global summary.

Address alignment. In the three ECA-Hash (A4) λ=4 checkpoints of Table A13 (T2, M=17, seeds 1–3) the key dominates the written address: with $\mathbf{e}_n$ the key of the entity written at position n, $\cos(\widetilde{B}_n, \mathbf{e}_n) = 0.931$–$0.952$ across layers (per-layer means over seeds) against -0.01 ± 0.12 for other entities' keys (read address 0.904–0.928), with $\|\lambda \mathbf{e}_n\|/\|B_n\| \approx 3$. The learned content-address norm therefore does not follow the $\sqrt{N}$ scaling the idealized $\lambda \propto \sqrt{N}$ argument assumes, which is why that scaling is treated as a heuristic to verify per architecture.

Table A14 completes the mechanism picture. **Joint write/read keying reaches higher accuracy than either one-sided variant**: write-only keying (A2) remains at the floor at both tested scales, while read-only keying (A3) partially recovers at the larger scale (0.58 at λ=4)—consistent with the learnable content-write path aligning to a fixed read target, but not vice versa. These results support a benefit from joint write/read alignment. Finally, the pairwise worst case of Section 4 is measured directly (T4-C): evaluation sets force the target and one active distractor to be a key pair from a fixed $|\cos|$ bucket (label-independent selection; the maximum overlap among the 256 keys at N=64 is 0.53). Accuracy degrades gracefully and monotonically with pair overlap—0.956/0.946/0.923/0.911 at mean $|\cos| = 0.02/0.18/0.33/0.42$ (three checkpoint seeds)—so near-collisions are a real but modest failure mode: the worst 58 of ∼33k pairs cost about five points, far from a collapse. **The two key sources trade seen-entity accuracy against deployment properties** (Table A15). A well-spread Gaussian-initialized ECA-Learned table reaches 0.973 ± .005

Table A15: Key sources on the security-grammar interference task (λ=4): accuracy on seen entities and on the T4-D split, whose test-time entity identifiers are disjoint from training, and key storage (five seeds for ECA-Hash and for the Gaussian-initialized table's seen split, three otherwise).

Key source	seen	unseen (T4-D)	unseen handling	key storage
ECA-Hash	.951 ± .006	.947 ± .006	procedural	$O(1)$
Learned, zero init	.804 ± .044	.362 ± .003	untrained zero rows	$O(\|\mathcal{E}\|N)$
Learned, Gaussian init	.973 ± .005	.975 ± .004	allocate+init row	$O(\|\mathcal{E}\|N)$

on seen entities, slightly above ECA-Hash (0.951 ± .006). Initialization is essential for the learned table: zero-initialized, it reaches only 0.80 and collapses to the floor on unseen identifiers, whose rows never receive updates, whereas well-spread rows keep their geometry. ECA-Hash's advantage is deployment: it stores no per-entity key table, needs no enumerated vocabulary or allocation policy, and supports unseen identifiers by construction. Both realize the same principle: entity geometry supplied to the write/read addresses. Two further architecture controls (Table A11) are informative: a Transformer with entity inputs sits at the floor, and so does *Gated DeltaNet* (0.34), whose delta rule implements exactly the same-key-overwrite mechanics this task rewards; the mechanics of keyed updating are not the bottleneck; the *source* of the key is. That interpretation is supported by the keyed Gated DeltaNet control: injecting the same deterministic entity key into Gated DeltaNet's own per-head q/k (respecting its L2 normalization; both before- and after-norm injection sites) lifts the architecture from its input floor to 0.952 ± .007 / 0.956 ± .004 (three seeds per site). The comparison is size-controlled: at the keyed model's own 1.1M configuration, the no-entity arm reaches 0.337±.010 and the entity-input arm—the keyed model itself at λ=0, a byte-identical architecture—0.339 ± .006, so within the identical model only the key injection separates floor from solved; the 4× larger `fla`-default baseline floors as well (0.335 ± .010). Both keyed Gated DeltaNet sites are within seed variation of ECA on the Naju backbone (0.951 ± .006; the full five-backbone consolidation is Table 3). The principle is portable across update-rule families. At the other end, an explicit per-entity GRU dictionary tops the task at 1.00 with entity-linear memory; ECA recovers 95% of that reference accuracy in a fixed-size state, locating it on the accuracy–memory spectrum the paper argues for.

Measured cost of the prior. We measure the accuracy–memory–throughput positions directly (RTX 4090, bf16, batched full-sequence forward at L=1024, B=32, median of 10 timed iterations; persistent state = tensors a streaming deployment retains between events, weights excluded); see Table A16. The hash-derived key itself has no trainable parameters; the +0.03M between the A0 and A4 rows is the entity/event *input* embedding tables that every entity-aware arm shares (A1-information matching, verified by parameter breakdown), not a key-table cost. ECA leaves the persistent state size unchanged, and on the fused Naju host A4 (entity/event inputs plus ECA) has 3.5% higher forward latency than A0 (paired measurement, 95% CI [3.1%, 4.0%]); the injection alone, timed by switching it off in the same weights (λ=0 vs. 4), adds 2.5% (95% CI [2.0%, 2.9%]). The seed-regenerable keys are cached as a float32 257×64 buffer in each layer (257 KiB in total), which the state column, like the weights, excludes. For the official blocks, the throughput change also includes the switch to the non-fused path (Table A16). The dictionary's state grows linearly in retained entities (≈0.45 GB if all 1.77M ACME process instances were kept, which a deployment need not do; at M_{pool}=257 the state is below Naju's), and its per-token sequential update runs 5×

Table A16: Measured cost of the prior: parameters, persistent state bytes, and throughput (all rows re-measured together on one RTX 4090, bf16, B=32, L=1024, median of 10). On the fused Naju host, A4 has 3.5% higher forward latency than A0 (15 alternating paired repetitions; normal-approximation 95% CI $[3.1\%, 4.0\%]$), which includes the identity-input embeddings as well as the injection; state bytes are computed from the architecture, not measured, and are unchanged by ECA; every keyed row keeps state constant in both the entity count and the horizon, unlike the per-entity dictionary ($O(M)$) and the Transformer's input window ($O(T)$). ‡The keyed official variants run each block's *non-fused reference* forward (the code path that exposes B/C), so their 8–40% slow-down mixes two factors—leaving the fused kernel and the injection itself—that these measurements do not decompose; attributing the cost requires a same-path incremental measurement (reference with vs. without keys). The Transformer is a bidirectional encoder that re-encodes its full window, so it has no reusable KV cache and no state-byte entry.

Model	params	state (bytes)	tokens/s	state scaling
Mamba-1 (official)	2.15M	274 K	5.46×10^6	$O(1)$ in M, T
+ keys‡	2.22M	274 K	4.08×10^6	$O(1)$ in M, T
Mamba-2 (official)	1.84M	278 K	8.45×10^6	$O(1)$ in M, T
+ keys‡	1.91M	278 K	5.11×10^6	$O(1)$ in M, T
Mamba-3 (official)	2.01M	526 K	5.19×10^6	$O(1)$ in M, T
+ keys‡	2.07M	526 K	4.80×10^6	$O(1)$ in M, T
Naju host (A0)	0.71M	137 K	3.80×10^6	$O(1)$ in M, T
+ ECA-Hash (A4)	0.74M	137 K	3.66×10^6	$O(1)$ in M, T
Gated DeltaNet	1.11M	287 K	7.59×10^6	$O(1)$ in M, T
+ keys‡	1.14M	287 K	6.16×10^6	$O(1)$ in M, T
Entity-GRU dict.	0.15M	66 K at M_{pool}=257	0.70×10^6	$O(M)$
Transformer	4.31M	—	6.59×10^6	$O(T)$ window

slower here. ECA retains constant persistent state with respect to both the entity count and the sequence horizon, while the explicit dictionary and the full-window Transformer incur entity- or horizon-dependent storage, respectively.

G Extended security-detection literature

Classical detection consumes flows and windows: feature-engineered classifiers on CTU-13 (García et al., 2014) and CIC-IDS2017 (Sharafaldin et al., 2018), LSTM log models (Du et al., 2017), autoencoder ensembles (Mirsky et al., 2018), and provenance/graph systems (SLEUTH (Hossain et al., 2017), HOLMES (Milajerdi et al., 2019), Unicorn (Han et al., 2020), ThreaTrace (Wang et al., 2022), Kairos (Cheng et al., 2024)); Transformer families dominate traffic and log anomaly work (ET-BERT (Lin et al., 2022), YaTC (Zhao et al., 2023), FlowTransformer (Manocchio et al., 2024), NetGPT (Meng et al., 2023), LogBERT (Guo et al., 2021)), while entity-keyed chronological models are rare (ATLAS (Alsaheel et al., 2021), tiered RNN LMs on LANL (Tuor et al., 2018), user-sequenced encoders on CERT (Elbasheer & Akinfaderin, 2025)). The recent Mamba-security wave (NetMamba (Wang et al., 2024), MambaITD (Kong et al., 2025b), Log2Sig (Kong et al., 2025a), MIDS (Liu et al., 2026), byte-level classification (Kulatilleke et al., 2026), traffic generation (Chu et al., 2024), graph-aware source detection (Cheng et al., 2025)) adds representation engineering, decomposition, or hybrid structure *around* the SSM on short windows; in the systems we surveyed, none keys the recurrent state to a semantic security entity. ECA fills that gap, and the security corpora serve as real-world tests of a general memory principle.

H Crosstalk lemma: pairwise worst case and caveats

Beyond the average $O(M/N)$ result of Eq. 5, a pairwise worst-case bound follows from sub-Gaussian concentration of Rademacher inner products: $\Pr(|e_i^\top e_j| > \epsilon) \le 2\exp(-cN\epsilon^2)$, so a union bound puts the maximum pairwise overlap among M entities at $O(\sqrt{\log M/N})$. This bounds *individual* overlaps only—it is not a bound on the full value-weighted retrieval error—so rather

than argue about the error tail from the pair bound we stress-test high-overlap key pairs directly in the collision ablation (Appendix F). The idealization also omits three learned effects present in the full model: retain-gate decay, content-dependent addresses, and mutable-state overwriting. Exactly storing M independent K-state variables needs $M \log_2 K$ bits regardless of mechanism, so the constant-state operating point is approximate by construction. Two further caveats: the deterministic hash realizes *one* fixed key draw, so per-seed geometry can deviate from the expectation (we report the seed and test unseen ids); and Eq. 5 is *static*—the learned recurrence adds gated decay, content-dependent addresses, and overwriting, and a key-only readout scales same-entity signal and crosstalk alike with λ (the monotone benefit of larger λ in Table A11 is a training effect: on the Naju host, rescaling learned weights absorbs λ exactly). Further predictions: (M, N) curves should approximately collapse on M/N if the static term dominates; $\lambda \propto \sqrt{N}$ is a heuristic (if $\|B\| \sim O(\sqrt{N})$ the relative prior strength scales as $\lambda/\sqrt{N}$), consistent with a two-point Mamba-1 check at $N{=}64$; and the explicit per-entity dictionary serves as a measured no-crosstalk reference, not a derived bound.

I Address prior versus isolated entity memory

ECA is an address prior, not an isolated entity memory. The retain gate still acts on the shared recurrent state, so retention remains global rather than per-entity; additive writes do not guarantee exact replacement of a mutable value; different entity keys can collide (hash keys with probability concentrating as N grows, learned keys with no bound); and the effective number of entities the state can serve depends jointly on the state width N, key similarity, and the update dynamics—not on a hard slot count. State separation is therefore measured rather than assumed: by the T4 tests (Appendix C), the paired interventions (Table A13), and an explicit per-entity memory (Entity-GRU dictionary) as an accuracy reference.