Forensic Provenance for LLM Deployments: Real-Time Activation Watermarking for Legal Non-Repudiation

Jose Joaquín Vargas Altalaguerri
`jvargas8@xtec.cat` · Independent Researcher
June 2026

Working paper draft. All quantitative claims come from real runs on Qwen2.5-0.5B (RTX A5000), reproduced in the repository `ttzrs/neural-cdma` *(*`*_result.json`*). System:* `watermark_pkg/` *(*`cdma-watermark`*). License: CC-BY-4.0 (paper), MIT (code).*

Abstract

We present a white-box activation watermarking system that gives an LLM provider **real-time forensic provenance** over its own deployment: each generation is tagged, in the residual stream, with a per-session payload (session/user ID, timestamp) using fixed key-derived sign codes, at an amplitude low enough that the generated text is unchanged (KL ≈ 3×10^{-4}, token-agreement 1.0). From the provider's activation logs, the payload is recovered and a one-sided binomial test yields a legal-grade provenance statement: *this exact generation was produced by our model under session key K* ($p = 1.5\times10^{-5}$ at 16 bits), or — exculpatory — *this text carries no valid signature, so it was not produced by our deployment.* An attacker without the key reads only chance (≈0.5/bit). Crucially, the watermark is recoverable from **activations, not from text**: the final LayerNorm + unembedding scrambles the code before the logits (logit-side recovery stays at chance for every amplitude we tested). This non-linear collapse is not an implementation limit but a structural one, which we characterize across six superposition-based applications (KV-cache merging, knowledge ablation, multi-concept steering, text watermarking, parallel compute, modality multiplexing). The same physics that kills a *copyright* watermark (detectable from text) is what makes a *forensic* one (verifiable from the provider's own logs) sound. We ship the system as an installable package with an honest threat model and contribute, in the appendix, the capacity theory that bounds it: superposition lives in the linear channel and is destroyed by every untrained non-linear readout, and language's per-token demand collapses the multiplexing capacity to K≈2.

Contributions. (1) A white-box forensic activation-watermarking system for LLM non-repudiation, with an attribution *and* an exculpatory mode and a binomial-test provenance guarantee ($p = 1.5\times10^{-5}$ at 16 bits, text unchanged), shipped as an installable package. (2) An applied **negative-results map** of where engineered CDMA superposition collapses on *off-the-shelf* pretrained models, across six applications and five distinct non-linear readouts. (3) The **surprise → multiplexability** finding and the language K_cap≈2 capacity collapse. **Critical limitation:** verification is white-box (the provider must log activations) and version-bound — forced by the non-linearity, not a design choice (§5).

---

1. The problem: who is accountable for an AI generation?

An LLM provider serving a high-stakes domain (medicine, law, finance) faces a non-repudiation problem. Consider:

> A hospital uses the provider's LLM through an API. A clinician acts on an AI-generated diagnosis, a patient is harmed, and the provider is sued: *"the model told me to prescribe this."* Two questions decide liability, and today neither has a cryptographic answer: 1. **Attribution.** Did *our* model actually produce this exact text, in this session, at this time? 2. **Exculpation.** Or was the text altered, fabricated, or produced by a different model — in which case we are not liable?

Output-side text watermarking (Kirchenbauer-style logit biasing) cannot answer either question robustly: it is removable by paraphrase, it perturbs the output distribution, and it does not bind a generation to a session. What a provider *does* control is its own deployment — including the model's internal activations, which it can log. We exploit exactly this asymmetry.

2. Threat model (stated honestly)

- **Who embeds:** the provider, inside its own serving stack, with a secret key K and a per-session payload.
- **Who verifies:** the provider (or a court-appointed auditor) with (a) the model weights/version, (b) the key K, (c) the logged residual activations of the disputed generation, (d) the injection/read layer indices.
- **What it proves (inculpatory):** that a specific generation carries the provider's key-K signature encoding session metadata — to within a binomial p-value.
- **What it proves (exculpatory):** that a candidate text's logged activations do *not* carry the signature → not produced by this deployment.
- **What it does NOT do:** prove provenance from *text alone* (black-box). The signature is not recoverable from the output tokens/logits (§5). It also assumes the verifier has the **same model version**; a fine-tuned or quantized model changes activations and breaks verification (the provider knows its version).
- **Adversary:** cannot read or forge the mark without K (decoding with a wrong key returns chance, §4).

This is a forensic / audit-logging tool for the party that runs the model, not an anti-theft or anti-distillation copyright tool. The distinction is forced by the physics (§5), not a design preference.

3. Mechanism

Let h ∈ ℝ^d be the residual stream at an injection layer ℓ_inj. A key K seeds a generator producing a code matrix C ∈ {±1}^{d×B} (B payload bits), C_{·,k}/√d. A payload b ∈ {±1}^B (session ID etc.) is injected as a low-amplitude perturbation during inference:

$$h' = h + \alpha \|h\| \cdot (C\, b), \quad \alpha \approx 0.01$$

applied via a forward pre-hook so the text is generated from h'. The provider logs the residual at a read layer ℓ_read. Recovery correlates the logged activation against the codes, averaged over token positions and calibrated against the model's own activation bias:

$$\hat{s} = C^{T} (\text{mean}_t\, h'_t - \text{calib}), \quad \hat{b} = \text{sign}(\hat{s})$$

Verification compares $\hat{b}$ to the claimed payload and reports the one-sided binomial tail p = P(≥ m matches | chance ½) over B bits. Multi-bit session IDs use a repetition code (ECC) to trade bits for reliability. The full two-party protocol (inject with owner key, decode the *injected delta* with owner vs attacker keys) is exposed as the `audit` command.

4. Evaluation (real, Qwen2.5-0.5B, fp32)

Injection layer 8, read layer 20, α = 0.01, averaged over 8 prompts × continuations.

Quantity	Result
Text fidelity (KL clean‖watermarked)	**3×10^{-4}**
Token-agreement (argmax unchanged)	**1.000** (text identical)
Owner recovery, B=16	bit-acc **1.000** (avg) / 0.92 (per-prompt), **$p = 1.5\times10^{-5}$** → CONFIRMED
Owner recovery, B=64 / B=256	bit-acc 0.77 / 0.64 (more bits → lower per-bit acc; use ECC)
Attacker (wrong key), B=16	10/16, **p = 0.23** → not confirmed (chance)
Logit / black-box recovery	**0.52–0.57** (chance) at *every* amplitude — text watermark impossible

Decoupled `audit` demo (one owner, two attacker keys), injected delta decoded by each:

```
OWNER  'acme-corp-2026': 16/16 bits  p=1.5e-05  → PROVENANCE CONFIRMED
ATTACK 'thief-key':      10/16 bits  p=2.3e-01  → chance (cannot read without the
secret)
ATTACK 'competitor-xyz':  9/16 bits  p=4.0e-01  → chance
```

System: installable `cdma-watermark` package (codec / embed-extract-verify / ECC / CLI `embed|verify|extract| audit`), 23/23 tests pass, real-model roundtrip included.

5. Why forensic and not copyright: the non-linear wall

The watermark is recoverable from the residual stream (a linear read) at KL≈0, but **not** from the logits. The final RMSNorm + unembedding W_U is a non-linear map; the low-amplitude code direction in residual space does not survive it as a linearly-detectable structure (logit-side bit-acc 0.52–0.57 across all α, even where residual recovery is 0.92). A provider who logs activations can verify; an outside party with only the text cannot. This is the structural reason the product is forensic (provider-side logs) rather than copyright (text-detectable): **the same non-linearity that destroys a text watermark is what we lean on for the white-box one.** The appendix shows this is one instance of a general law.

6. Limitations

1. White-box: needs logged activations; verification is provider-side or court-supervised.
2. Same-model-version: fine-tuning / quantization changes activations and breaks verification.
3. Payload size vs reliability: per-bit accuracy falls with B (0.92→0.77→0.64 at 16→64→256); session IDs need ECC (repetition) or hashing. Documented in the package.
4. Not a text watermark; not robust to the provider failing to log.
5. Demonstrated on a 0.5B model at one layer pair; scaling study pending.

7. Related work

Output/logit watermarking (Kirchenbauer et al.) is black-box but removable and distribution-perturbing. White-box model fingerprinting — SEAL (subspace-anchored), EverTracer (probabilistic fingerprint), activation/hidden-state signatures — targets *model ownership* (is this model mine?). Ours targets *generation non-*

repudiation (did this session produce this text?), a per-inference forensic-logging primitive with a session payload and an exculpatory mode, which is a different threat model. The watermarking mechanism is CDMA/spread-spectrum (key-derived sign codes); the scope boundary is explained by computation-in-superposition theory (appendix).

Appendix: The science that bounds the system

The system above is one usable corner of a larger empirical study of **engineered superposition (CDMA sign codes) on off-the-shelf pretrained LLMs**. The single law that emerged also explains exactly why the watermark is white-box-only.

A. Linear / non-linear channel view

A transformer alternates **linear communication** (W_{q,k,v,o}, MLP projections, residual, embeddings, unembedding) and **non-linear processing** (softmax attention, GELU, LayerNorm, softmax logits). Cost lives in the linear part and is compressible (BitNet/Mamba/Haar/CDMA/sparsity); *capacity* lives in the non-linear part, scales as C_nl $\propto$ d, and is consumed at a per-token rate D_task. Efficiency $\leq$ 1/max(linear_cost, D_task): linear tricks help only down to D_task. The lever is the *heterogeneity* of D_task, not compressing the linear channel.

B. The one robust law

CDMA superposition survives only in the linear channel (linear/white-box read or a *trained* de-multiplexer); every untrained non-linear readout collapses it. Confirmed independently at five non-linear points:

Non-linear point	Application probe	Result
Unembedding (logits)	text/black-box watermark	recovery → chance (§5)
Attention softmax	KV-cache CDMA merge	perplexity 8.57 → **172,904** (catastrophic); plain eviction gives 1.5× context at +0.29
MLP non-linearity	knowledge-ablation "neurosurgery"	concept localizes (116 neurons drop a fact 0.40→0.00) but ppl +50–266% (not surgical); multiplexed scan non-separable (cosine −0.16)
Downstream stack	multi-concept eigen-steering	orthogonalizing codes cuts cross-talk 2.2% but halves on-target retention; interference is non-linear, not geometric
(output)	parallel-compute multiplex	logits collapse (prior `multithread.py`)

Positive controls in the *linear/trained* regime work: residual watermark (§4); trained masked-multiplex rides to K=32 (App. C).

C. Capacity law and the language collapse

Trained masked-multiplex on a small transformer follows **$K_{cap} \approx 0.37 \cdot d/\log_2 V$** (toy). But on real causal Qwen-0.5B, both LoRA and full fine-tuning cap at **$K_{cap} = 2$** — language's $D_{task} \gg \log_2 V$, so the law over-predicts ~10×. The bottleneck is the language, not the method.

D. Surprise predicts multiplexability (the most distinctive finding)

Per-token **surprise predicts CDMA-multiplexability**, near-binary: low-surprise ("easy") tokens recover under superposition, high-surprise ("hard") tokens collapse even at K=1. Stratified by surprise tercile (Qwen-0.5B): easy recovery 0.60 at K=1 (vs a prior-only floor 0.124), hard 0.000 at every K; easy-token retention vs K=1 is 0.85 (K=2), 0.70 (K=4), 0.44 (K=8). Multiplexing *easy* tokens is cheap-but-not-free at low K.

E. Modality and token design are not levers

- **Modality:** in the masked toy, text and code ride to K_cap=32; **images collapse at K_cap=4** regardless of tokenizer (crude k-means *and* a trained VQ-VAE). The gap is a modality effect (pixel-derived tokens are low-rank/correlated, non-orthogonalizable), not token redundancy — and it is *not* predicted by token autocorrelation.
- **Code design:** under training, the binding code is irrelevant — perfectly orthogonal (Hadamard, coherence 0.000), random (0.266) and near-collinear (0.969) codes all reach K_cap=32, because the learned embedding absorbs any code. The lever is *training*; the ceiling is *D_task*. Designing tokens/codes adds nothing.

F. Honest novelty

Mechanism (trained MIMO via orthogonal keys) is **MIMONets** (NeurIPS'23); the linear-readout floor is formalized in **arXiv:2605.01192** (theory); natural task-superposition in pretrained LLMs is **arXiv:2410.05603**; parallel streams are **PARSCALE**; entropy-guided caching exists (**EntropyCache**). Our contributions are (i) the **forensic activation-watermarking system** and its threat model; (ii) the **applied negative-results map** on *off-the-shelf* pretrained models (where engineered superposition dies, per application); and (iii) the **surprise → multiplexability** link with the K_cap=2 language collapse. Scope: empirical / systems note.

Reproducibility

All numbers from this repo: `watermark_provenance.py`, `watermark_pkg/` (§4–5); `kv_selective.py`, `cdma_neurosurgery.py`, `eigen_steering.py`, `multithread.py` (App. B); `capacity_law.py`, `validate_llm*.py` (App. C); `prototype_selective.py` (App. D); `vision_kcap*.py`, `code_kcap.py`, `vqvae_kcap.py`, `coded_vocab_kcap.py` (App. E). Provenance saga: SIGIL → lextrace → neural-cdma.